Guide de l’utilisateur

AWS Modernisation du mainframe

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Modernisation du mainframe: Guide de l’utilisateur

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques commerciales et la présentation commerciale d’Amazon ne peuvent pas être utilisées en relation avec un produit ou un service extérieur à Amazon, d’une manière susceptible d’entraîner une confusion chez les clients, ou d’une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce que la modernisation AWS du mainframe 1
Caractéristiques de la modernisation des AWS ordinateurs centraux 2
Modèles 3
Comment démarrer avec la modernisation du AWS mainframe 4
Services connexes 4
Accès à la AWS modernisation du mainframe 5
Utilisez-vous la modernisation du AWS mainframe pour la première fois ? 5
Tarification de la modernisation AWS du mainframe 5
Configuration pour la modernisation du AWS mainframe 7
Inscrivez-vous pour un Compte AWS 7
Création d'un utilisateur doté d'un accès administratif 8
Concepts 10
Application 10
Définition de l'application 11
Tâche par lots 11
Configuration 12
Ensemble de données 12
Environnement 12
Modernisation de l'ordinateur central 12
Parcours migratoire 12
Point de montage 13
Refactorisation automatisée 13
Replateforme 13
Ressource 13
Moteur d'exécution 13
Approche de modernisation 14
Phase d'évaluation 14
Phase de mobilisation 15
Phase de migration et de modernisation 15
Phase d'exploitation et d'optimisation 16
Mise en route 17
Tutoriel : Configuration d'un environnement d'exécution géré pour AWS Blu Age 17
Prérequis 18
Étape 1 : Téléchargez l'application de démonstration 18

Étape 2 : Création de la définition de l'application 18
Étape 3 : Création d'un environnement d'exécution 19
Étape 4 : Création d'une application 24
Étape 5 : Déploiement d'une application 26
Étape 6 : démarrer une application 28
Étape 7 : Accédez à l'application 28
Étape 8 : Tester l'application 29
Nettoyage des ressources 31
Tutoriel : Configuration d'un environnement d'exécution géré pour Rocket Software 31
Prérequis 32
Étape 1 : créer et charger un compartiment Amazon S3 32
Étape 2 : Création et configuration d'une base de données 34
Étape 3 : Création et configuration d'un AWS KMS key 36
Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données 37
Étape 5 : ajouter le SSLMode au secret 38
Étape 6 : Création d'un environnement d'exécution 39
Étape 7 : Création d'une application 44
Étape 8 : Déploiement d'une application 50
Étape 9 : Importer des ensembles de données 52
Étape 10 : démarrer une application 58
Étape 11 : Connectez-vous à l'application CardDemo CICS 59
Nettoyage des ressources 67
Étapes suivantes 68
Cycle de vie des composants 69
Aperçu du cycle de vie des composants 69
Mise à niveau de version 71
AWS Présentation de la version de Mainframe Modernisation Refactor with AWS Blu Age 71
Applications gérées 73
Création de AWS ressources pour une application migrée 74
Autorisations requises 74
Compartiment Amazon S3 75
Base de données 75
AWS Key Management Service clé 76
AWS Secrets Manager secret 76
Création d'une application 77

Création d'une application 77
Déployer une application 78
Déployer une application 79
Mise à jour d'une application 80
Mise à jour d'une application 80
Supprimer une application 81
Supprimer une application 81
Soumettre des tâches par lots pour les candidatures 82
Soumettre une tâche par lots 82
Redémarrer un traitement par lots 83
Annuler les tâches par lots pour les applications 84
Annuler une tâche par lots 85
Importer des ensembles de données pour les applications 85
Importer un ensemble de données 86
Exporter des ensembles de données pour les applications 87
Exporter un ensemble de données 87
Gérez les transactions pour les applications 88
Gérez les transactions pour les applications 88
Configuration de l'application gérée par Rocket Software 89
Intégrations tierces prises en charge pour Rocket Software 90
Configuration de l'application gérée AWS Blu Age 92
Structure des applications gérées par AWS Blu Age 92
Configuration de l'accès aux utilitaires pour les applications gérées 94
Configurer des propriétés supplémentaires pour l'application gérée 106
Référence de définition de l'application 127
Section d'en-tête générale 128
Présentation de la section consacrée aux définitions 130
AWS Exemple de définition d'application Blu Age 130
AWS Détails de la définition de Blu Age 131
Définition de l'application Rocket Software 137
Détails de la définition du logiciel Rocket 139
Référence de définition de l'ensemble de données 150
Propriétés communes 151
Exemple de format de demande d'ensemble de données pour VSAM 153
Exemple de format de demande d'ensemble de données pour la base GDG 155

Exemple de format de demande d'ensemble de données pour les générations PS ou GDG 156
Exemple de format de demande d'ensemble de données pour PO 157
Environnements d'exécution gérés 159
Création d'un environnement d'exécution 160
Création d'un environnement d'exécution 160
Mettre à jour un environnement d'exécution 164
Mettre à jour un environnement d'exécution 164
Fenêtre de maintenance 165
Arrêter un environnement d'exécution 165
Arrêter un environnement d'exécution 165
Redémarrer un environnement d'exécution 167
Redémarrer un environnement d'exécution 167
Supprimer un environnement d'exécution 168
Supprimer un environnement d'exécution 168
Tests d'application 169
Qu'est-ce que le test d'applications 169
Utilisez-vous les tests d'applications pour la première fois ? 170
Avantages des tests d'applications 171
Intégration avec AWS CloudFormation 171
Comment fonctionnent les tests d'applications 172
Services connexes 4
Accès aux tests d'applications 174
Tarification des tests d'applications 174
Concepts de test d'applications 174
Cas de test 175
Suite de tests 175
Configuration de l'environnement de test 176
Charger 176
Relire 176
Compare 176
Comparaison de bases de données 177
Comparaisons de jeux de 177
État de la comparaison 178
Règles d'équivalence 178
Comparaison des ensembles de données sur l'état final 179

Comparaisons de bases de données sur l'état 179
Equivalence fonctionnelle (FE) 179
Comparaisons d'écrans 3270 en ligne 179
Rejouer les données 180
Données de référence 180
Téléchargez, rejouez et comparez 180
Différences 181
Équivalences 181
Application source 182
Application cible 182
Prérequis pour les tests d'applications 182
Workflows de console dans le cadre des tests d'applications 182
Création de cas de test dans Application Testing 183
Création de suites de tests dans le cadre des tests d'applications 186
Création de configurations d'environnement de test dans Application Testing 188
Tutoriel : Configuration de CardDemo l'application dans le cadre des tests d'applications 191
Prérequis 191
Étape 1 : Préparation de la configuration CardDemo 191
Étape 2 : créer toutes les ressources nécessaires 192
Étape 3 : Déployer et démarrer l'application 193
Étape 4 : Importer les données initiales 193
Étape 5 : Connectez-vous à l' CardDemoapplication 194
Tutoriel : Rejouez et comparez sur AWS Blu Age en utilisant CardDemo 195
Étape 1 : Obtenir une image de machine Amazon EC2 Amazon AWS Blu Age (AMI) 195
Étape 2 : démarrer une EC2 instance Amazon à l'aide de l'AMI AWS Blu Age 196
Étape 3 : télécharger les fichiers CardDemo dépendants sur S3 197
Étape 4 : Chargement des bases de données et initialisation de l'application CardDemo 197
Étape 5 : Lancez le moteur d'exécution AWS Blu Age CloudFormation 200
Étape 6 : Test de l' EC2 instance Amazon AWS Blu Age 203
Étape 7 : Valider que les étapes précédentes ont été effectuées correctement 204
Étape 8 : Création du scénario de test 204
Étape 9 : Création d'une suite de tests 205
Étape 10 : Création d'une configuration d'environnement de test 205
Étape 11 : Téléchargez vos données d'entrée dans la suite de tests 206
Étape 12 : Rejouer et comparer 207
Pages de code d'ensembles de données prises en charge dans les tests d'applications 207

Protection des données dans le cadre des tests d'applications 219
Données collectées lors des tests de l'application de modernisation AWS du mainframe 220
Chiffrement des données au repos pour les tests d'applications de modernisation AWS du mainframe 221
Création d'une clé gérée par le client 222
Spécification d'une clé gérée par le client pour les tests AWS d'applications de modernisation du mainframe 223
AWS Modernisation du mainframe | Test des applications | Contexte de chiffrement 224
Surveillance de vos clés de chiffrement 225
Chiffrement en transit 225
Comment les tests d'applications fonctionnent avec IAM 225
Politiques basées sur l'identité 226
Politiques basées sur les ressources 227
Actions de politique 228
Ressources de politique 229
Clés de condition de politique 231
ACLs 232
ABAC 232
Informations d'identification temporaires 233
Transmission des sessions d'accès 233
Rôles de service 234
Rôles liés à un service 234
AWS Refactorisation de Blu Age 235
AWS Sorties de Blu Age 236
AWS Versionnage de Blu Age 236
AWS Options d'exécution de Blu Age 237
AWS Notes de publication de Blu Age 241
AWS Failles de sécurité dans Blu Age 352
Mise à niveau de AWS Blue Age 353
AWS Cycle de vie de Blu Age 356
AWS Concepts de Blu Age Runtime 357
Architecture de haut niveau 357
Structure d'application modernisée 362
Comprendre les simplificateurs de données 399
AWS Fard à joues Blue Age 407
Programmes disponibles dans l'application Web utilitaire 431

Console d'administration Blusam 433
AWS Configuration de Blu Age Runtime 474
Principes de base de configuration des applications 475
Priorité des applications 477
JNDI pour bases de données 477
AWS Les secrets de Blu Age Runtime 478
Autres fichiers (groovy, sql, etc.) 490
Application Web supplémentaire 491
Activer les propriétés 491
Propriétés du cache Redis disponibles 579
Configuration de la sécurité pour les applications Gapwalk 596
AWS Blue Age Runtime APIs 612
Points de terminaison pour la construction URLs 612
Points de terminaison pour l'application Gapwalk 613
Points de terminaison REST de la console d'applications Blusam 634
Gérer la console d'applications JICS 657
Structures de données 678
Configurer AWS Blu Age Runtime (non géré) 688
AWS Prérequis pour Blu Age Runtime 688
Intégration à AWS Blue Age Runtime 689
Exigences de configuration de l'infrastructure 694
AWS Artefacts de Blu Age Runtime 701
Déployez AWS Blu Age Runtime sur Amazon EC2 704
Déployez AWS Blue Age Runtime sur Amazon ECS et Amazon EKS 715
Testez l' PlanetsDemo application 723
Modifiez le code source avec Blu Age Developer IDE 727
Tutoriel : Configuration de la AppStream version 2.0 pour AWS Blu Age Developer IDE 727
Tutoriel : Utiliser AWS Blu Age Developer sur AppStream 2.0 733
AWS FAQ sur Blu Age 750
Général 750
AWS Blue Age Runtime 753
Données 762
Transformation 764
Déploiement 765
Replateforme du logiciel Rocket 768
Configurer Rocket Software (sur Amazon EC2) 768

Prérequis pour Rocket Software (sur Amazon EC2) 769
Création du point de terminaison Amazon VPC pour Amazon S3 769
Demander la mise à jour de la liste d'autorisation pour le compte 772
Créez le AWS Identity and Access Management rôle 773
Accordez à License Manager les autorisations requises 780
Abonnez-vous aux Amazon Machine Images 781
Lancer une instance de Rocket Software 785
Sous-réseau ou VPC sans accès à Internet 792
Configurer l'automatisation AppStream 2.0 798
Configurer l'automatisation au début de la session 799
Configurer l'automatisation à la fin de la session 799
Afficher les ensembles de données sous forme de tables dans Enterprise Developer 800
Prérequis 800
Étape 1 : configurer la connexion ODBC à la banque de données Rocket Software (base de données Amazon RDS) 801
Étape 2 : Création du fichier MFDBFH.cfg 803
Étape 3 : Création d'un fichier de structure (STR) pour la mise en page de votre cahier 804
Étape 4 : Création d'une vue de base de données à l'aide du fichier de structure (STR) 807
Étape 5 : Afficher les ensembles de données de Rocket Software (anciennement Micro Focus) sous forme de tableaux et de colonnes 807
Modifier des ensembles de données à l'aide des outils de fichiers de données dans Enterprise Developer 808
Prérequis 809
Outils de fichiers de données Launch Rocket Software (anciennement Micro Focus) 809
Modifier les ensembles de données VSAM stockés dans la base de données MFDBFH 810
Modifier les ensembles de données non VSAM stockés dans la base de données MFDBFH 814
Modifier les ensembles de données VSAM et non-VSAM stockés dans le système de fichiers (EFS/FSx) 816
Tutoriels pour Rocket Software 817
Tutoriel : Configuration de la version pour l' BankDemoexemple d'application 817
Tutoriel : Configuration CI/CD pipeline avec Rocket Enterprise Developer 828
Tutoriel : Configuration de la AppStream version 2.0 pour Enterprise Analyzer et Enterprise Developer 853
Tutoriel : Utiliser des modèles avec Rocket Enterprise Developer 862
Tutoriel : Configuration d'Enterprise Analyzer 873

Tutoriel : Configuration d'Enterprise Developer 884
Utilitaires Batch 890
Emplacement binaire 891
Utilitaire batch M2SFTP 891
Utilitaire batch M2WAIT 898
TXT2Utilitaire PDF par lots 900
Utilitaire de traitement par lots M2DFUTIL 906
Utilitaire batch M2RUNCMD 913
Transfert de fichiers 918
Qu'est-ce que le transfert de fichiers 918
Avantages du transfert de fichiers lié à la modernisation du mainframe AWS 919
Comment fonctionne le transfert de fichiers pour la modernisation du mainframe AWS 919
Installation d'un agent de transfert de fichiers 920
Étape 1 : créer un ensemble de données ZFS pour l'agent M2 921
Étape 2 : Formater l'ensemble de données au format ZFS 921
Étape 3 : monter le système de fichiers 921
Étape 4 : vérifier le support 922
Étape 5 : Entrez OMVS 922
Étape 6 : définir la variable d'environnement du répertoire d'installation de l'agent 922
Étape 7 : définir la variable d'environnement du répertoire de travail 922
Étape 8 : Création du répertoire de travail 922
Étape 9 : Copiez le fichier tar de l'agent et copiez le répertoire de travail 923
Étape 10 : terminer l'installation de l'agent 923
Configuration d'un agent de transfert de fichiers 924
Étape 1 : configurer les autorisations et démarrer le contrôle des tâches (STC) 924
Étape 2 : créer des compartiments Amazon S3 926
Étape 3 : Création d'une clé gérée par AWS KMS le client pour le chiffrement 926
Étape 4 : Création d'un AWS Secrets Manager secret pour les informations d'identification du mainframe 927
Étape 5 : Création d'une politique IAM 928
Étape 6 : Création d'un utilisateur IAM avec des informations d'identification d'accès à long terme 930
Étape 7 : Création d'un rôle IAM que l'agent devra assumer 931
Étape 8 : Configuration de l'agent 932
Création de points de terminaison de transfert de données 935
Création de points de terminaison de transfert de données 935

Création de tâches de transfert ... 937
Création de tâches de transfert ... 938
Afficher les tâches de transfert ... 941
Tutoriel : Débuter avec le transfert de fichiers ... 942
Présentation ... 942
Étape 1 : transférer le package tar des fichiers binaires de l'agent AWS vers la partition logique du mainframe ... 943
Étape 2 : Configuration de l'agent de transfert de fichiers sur le mainframe source ... 943
Étape 3 : Création d'un point de terminaison de transfert de données ... 943
Étape 4 : Création d'une tâche de transfert ... 943
Étape 5 : Afficher la progression de la tâche de transfert ... 943
Pages de code source et cible prises en charge ... 944
Types d'ensembles de données du mainframe ... 944
Pages de code prises en charge ... 944
AWS Transform pour ordinateur central ... 946
Principaux avantages ... 946
Procédure pas à pas de la transformation de la console d'applications du mainframe ... 947
Réplication des données avec Precision ... 948
Prérequis ... 948
Abonnez-vous à l'Amazon Machine Image ... 948
Lancez la réplication des données de modernisation du AWS mainframe avec Precisely ... 949
Créer une politique IAM ... 950
Créer un rôle IAM ... 951
Attachez le rôle IAM à l'instance Amazon EC2 ... 951
Conversion d'un assembleur avec mLogica ... 953
Qu'est-ce que la conversion en assembleur avec MLogica ... 953
Compilateurs de conversion de code ... 954
Architecture de conversion de code ... 954
Approche d'automatisation ... 955
Sécurité ... 955
Ressources supplémentaires ... 956
Comprendre la facturation par conversion de code ... 956
Conversion de code, facturation et champ d'application ... 956
Concepts de conversion de code ... 958
Gestion des macros ... 959
Pages de codes (EBCDIC ou ASCII) ... 959

CodeBuild 959
Comprendre les composants et les processus 959
AWS Mainframe Modernization contenant 960
compartiment de projet S3 961
Emplacement des fichiers journaux 961
Présentation du processus 961
Tutoriel : Convertir le code d'Assembler en COBOL 962
Prérequis 963
Étape 1 : Partagez les actifs de construction avec Compte AWS 963
Étape 2 : créer des compartiments Amazon S3 964
Étape 3 : Création d'une politique IAM 964
Étape 4 : Création d'un rôle IAM 966
Étape 5 : associer la politique IAM au rôle IAM 967
Étape 6 : Création du CodeBuild projet 967
Étape 7 : Définition du projet et téléchargement du code source 974
Étape 8 : Exécuter l'analyse et comprendre les rapports 975
Étape 9 : Exécuter la conversion du code 977
Étape 10 : Vérifiez la conversion du code 981
Étape 11 : Téléchargez le code converti 982
Nettoyage des ressources 982
Intégration de Charon 984
Présentation de Charon-SSP 984
Systèmes d'exploitation clients pris en charge 986
Conditions préalables à l'instance cloud Charon-SSP 987
Prérequis pour l'instance 988
Création et configuration d'une instance AWS cloud pour Charon (nouvelle interface graphique) 990
Prérequis généraux 990
Utilisation du AWS Management Console pour lancer une nouvelle instance 991
Replateforme avec NTT DATA 997
Prérequis 997
Abonnez-vous à l'Amazon Machine Image 997
Lancez la replateforme de modernisation du AWS mainframe avec l'instance NTT DATA 998
Commencer à utiliser NTT Data 998
Tutoriel : Déployer CardDemo une application sur NTT DATA 1000
Schéma du flux de déploiement 1000

Prérequis 1001
Étape 1 : Préparation de l'environnement 1002
Étape 2 : Création d'une région TPE 1002
Étape 3 : Création du nœud et du sous-système BPE 1003
Étape 4 : Compiler et déployer CardDemo l'application 1012
Étape 5 : Importer le catalogue BPE et TPE 1014
Étape 6 : démarrer et connecter le TPE au BPE 1014
Étape 7 : Exécutez l' CardDemoapplication 1015
Résolution des problèmes 1021
Sécurité 1023
Protection des données 1024
Données collectées par AWS Mainframe Modernization 1025
Chiffrement des données arrêté pour le service de modernisation des AWS mainframes ... 1027
Comment la modernisation AWS du mainframe utilise les subventions dans AWS KMS 1029
Création d'une clé gérée par le client 1031
Spécification d'une clé gérée par le client pour la modernisation AWS du mainframe 1033
AWS Contexte de chiffrement de la modernisation du mainframe 1034
Surveillance de vos clés de chiffrement 1035
En savoir plus 1051
Chiffrement en transit 1051
Gestion de l'identité et des accès 1052
Public ciblé 1052
Authentification par des identités 1053
Gestion des accès à l'aide de politiques 1057
Comment fonctionne la modernisation AWS du mainframe avec IAM 1060
Exemples de politiques basées sur l'identité 1075
Résolution des problèmes 1079
Utilisation des rôles liés aux services 1080
Validation de conformité 1084
Résilience 1085
Sécurité de l'infrastructure 1085
AWS PrivateLink 1086
Considérations 1087
Création d'un point de terminaison d'interface 1087
Création d'une politique de point de terminaison 1087
Surveillance 1089

Surveillance avec CloudWatch 1089
Métriques de l'environnement d'exécution 1090
Métriques d'application 1091
Dimensions 1097
Journalisation des appels d'API CloudTrail avec 1097
AWS Informations sur la modernisation du mainframe dans CloudTrail 1098
Comprendre les AWS entrées des fichiers journaux de modernisation des mainframes 1099
Résolution des problèmes dans M2 1101
Erreur de résolution des problèmes : expiration du délai d'attente pendant que le nom de l'ensemble de données soit déverrouillé 1101
Cause courante 1102
Résolution 1102
Forcer le verrou à le relâcher 1102
Configurer le mécanisme de réparation auto Blusam 1103
Gestionnaire de serrures Blusam 1104
Erreur de résolution des problèmes : Impossible d'accéder à l'URL d'une application 1105
Cause courante 1105
Résolution 1105
Résolution des problèmes : AWS Blu Insights ne s'ouvre pas depuis la console 1106
Cause courante 1107
Résolution 1107
Erreur de résolution des problèmes : environnement malsain 1107
Cause courante 1108
Résolution 1108
Résolution des problèmes de licence pour Rocket Software 1109
Vérifiez que l' EC2 instance Amazon possède le rôle de licence IAM 1109
Utiliser l'analyseur d'accessibilité 1110
Exécutez le daemon de licence 1110
Problèmes de licence avec Enterprise Server ou Enterprise Build Tools sous Linux après l'application de correctifs au système d'exploitation 1111
Historique de la documentation 1113
..... mcxx

Qu'est-ce que la modernisation AWS du mainframe ?

AWS La modernisation du mainframe vous aide à moderniser vos applications mainframe pour les adapter à des environnements d'exécution AWS gérés. Il fournit des outils et des ressources pour vous aider à planifier et à implémenter la migration et la modernisation. Vous pouvez analyser vos applications mainframe existantes, les développer ou les mettre à jour à l'aide de COBOL ou PL/I implémenter un pipeline automatisé pour une intégration et une livraison continues (CI/CD) des applications. Vous pouvez choisir entre des modèles de refactoring automatique ou de replateforme, en fonction des besoins de vos clients. Si vous êtes consultant et que vous aidez un client à migrer ses charges de travail sur le mainframe, vous pouvez utiliser les outils de modernisation du AWS mainframe pour toutes les phases du processus de migration et de modernisation, de la planification initiale aux opérations cloud après la migration.

Vous pouvez utiliser la modernisation du AWS mainframe pour créer et gérer efficacement l'environnement d'exécution de vos applications mainframe, ainsi que AWS pour gérer et surveiller vos applications modernisées.

Rubriques

- Caractéristiques de la modernisation des AWS ordinateurs centraux
- Modèles
- Comment démarrer avec la modernisation du AWS mainframe
- Services connexes
- Accès à la AWS modernisation du mainframe
- Utilisez-vous la modernisation du AWS mainframe pour la première fois ?
- Tarification de la modernisation AWS du mainframe

> **Note**
>
> Avez-vous fait appel à des partenaires compétents en matière de migration de AWS mainframe ou à des services AWS professionnels pour votre projet de modernisation de mainframe ? Si ce n'est pas le cas, nous vous recommandons vivement de faire appel à des experts pour votre projet.
>
> - AWS Partenaires compétents en matière de modernisation des ordinateurs centraux

- AWS Professional Services

Les fonctionnalités et les cas d'utilisation de la modernisation des AWS mainframes soutiennent une approche de modernisation évolutive, qui offre des avantages à court terme en améliorant l'agilité et en offrant de nombreuses opportunités d'optimisation et d'innovation par la suite. Pour de plus amples informations, veuillez consulter Approche de modernisation.

Caractéristiques de la modernisation des AWS ordinateurs centraux

AWS Les fonctionnalités de modernisation du mainframe prennent en charge les cas d'utilisation suivants :

- Évaluation : La fonctionnalité d'évaluation de AWS Mainframe Modernization peut vous aider à évaluer, définir et planifier un projet de migration et de modernisation.
- Refactor : grâce à AWS Blu Age, vous pouvez utiliser le refactoring pour convertir les anciens langages de programmation d'applications, créer des macroservices ou des microservices et moderniser les interfaces utilisateur (UIs) et les piles de logiciels d'application.

  AWS Blu Insights est désormais disponible AWS Management Console via l'authentification unique. Vous n'avez plus besoin de gérer des informations d'identification AWS Blu Insights distinctes. Vous pouvez accéder aux fonctionnalités de la base de code AWS AWS Blu Age et du centre de transformation directement depuis le AWS Management Console.
- Replateforme : grâce à la solution Micro Focus Enterprise, vous pouvez porter l'application sur laquelle une grande partie du code source de l'application est recompilée sans modification.
- IDE pour développeurs : AWS Mainframe Modernization propose un environnement de développement intégré (IDE) à la demande qui permet aux développeurs d'écrire du code plus rapidement grâce à une édition et un débogage intelligents, à une compilation instantanée du code et à des tests unitaires.
- Temps d'exécution géré : l'environnement d'exécution géré AWS Mainframe Modernization surveille en permanence vos clusters afin de garantir le bon fonctionnement des charges de travail de l'entreprise grâce à des calculs autoréparants et à une mise à l'échelle automatisée.
- Intégration et livraison continues (CI/CD) : la CI/CD fonctionnalité de modernisation du AWS mainframe aide les équipes de développement d'applications à apporter des modifications de

code plus fréquemment et de manière plus fiable, ce qui accélère la vitesse de migration, améliore la qualité et contribue à réduire le nombre time-to-market de nouvelles fonctions commerciales disponibles.

- Intégrations avec d'autres AWS services : la modernisation AWS du mainframe permet AWS CloudFormation AWS PrivateLink, et AWS Key Management Service pour un déploiement reproductible et une sécurité et une conformité accrues.
- Disponibilité étendue : la modernisation des AWS ordinateurs centraux est désormais disponible dans l'est des États-Unis (Ohio), l'ouest des États-Unis (Californie du Nord), l'Asie-Pacifique (Mumbai), l'Asie-Pacifique (Séoul), l'Asie-Pacifique (Singapour), l'Asie-Pacifique (Tokyo), l'Europe (Londres) et l'Europe (Paris).

Pour plus d'informations, consultez la section Fonctionnalités de modernisation AWS du mainframe.

Modèles

Le modèle de refactorisation automatisée, développé par AWS Blu Age, vise à accélérer la modernisation en convertissant l'ensemble des applications existantes et sa couche de données en une application Java moderne tout en préservant l'équivalence fonctionnelle. Au cours de cette transformation automatisée, il crée une application à plusieurs niveaux avec un front-end basé sur Angular, un backend Java compatible avec les API et une couche de données accédant aux magasins de données modernes. Le processus de refactorisation fournit des fonctionnalités équivalentes à celles de l'ancienne solution afin d'accroître l'automatisation des projets, ce qui se traduit par une rapidité, une qualité et une réduction des coûts, ce qui permet d'obtenir des avantages commerciaux plus rapidement. Pour plus d'informations, consultez la section AWS Mainframe Modernization Automated Refactor.

Le modèle de replateforme, basé sur la suite Rocket Software (anciennement Micro Focus) Enterprise, vise à préserver le langage, le code et les artefacts de l'application afin de minimiser l'impact sur les actifs et les équipes de l'application. Il aide les clients à maintenir leurs connaissances et leurs compétences en matière d'applications. Bien que les modifications apportées aux applications soient limitées, ce modèle facilite également la modernisation de l'infrastructure et des processus. L'infrastructure est remplacée par un service géré moderne basé sur le cloud, tandis que les processus sont modifiés pour suivre les meilleures pratiques en matière de développement d'applications et d'opérations informatiques. Pour plus d'informations, consultez la section AWS Mainframe Modernisation Replatform.

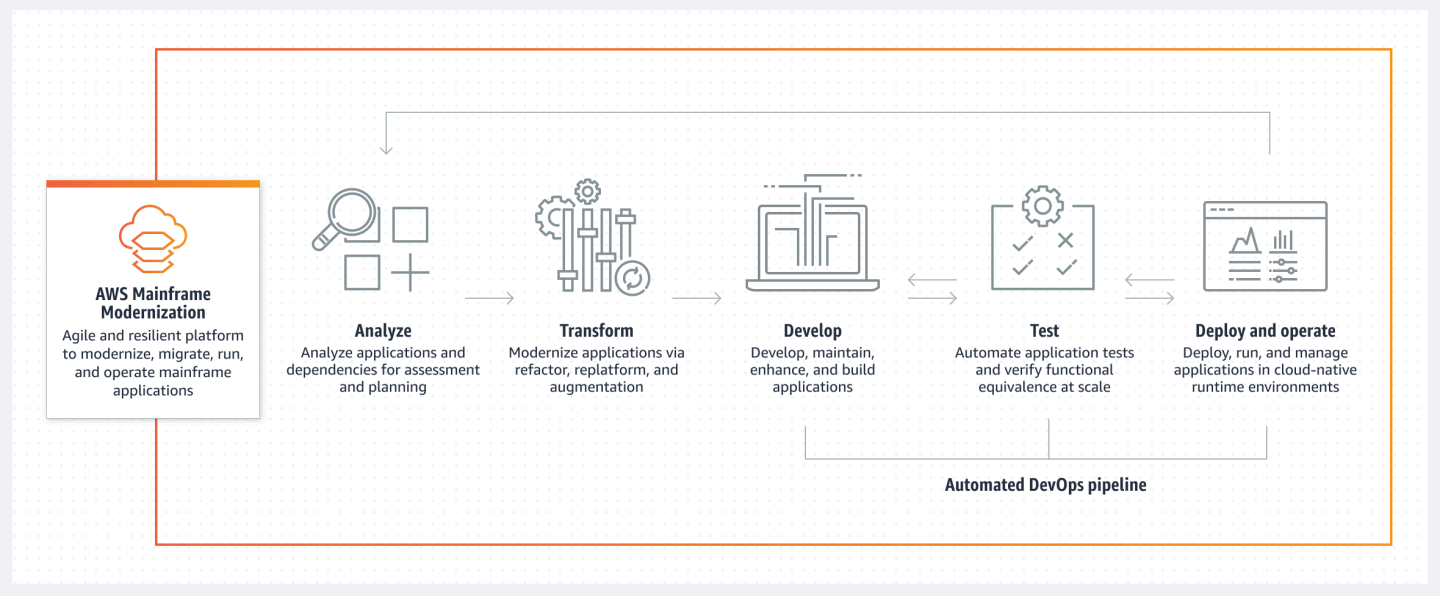
Comment démarrer avec la modernisation du AWS mainframe

À vous d'essayer ! Nous proposons des didacticiels et des exemples d'applications pour vous aider à vous faire une idée de ce qu'offre la modernisation des AWS mainframes. Choisissez le Tutoriel : Configuration d'un environnement d'exécution géré pour AWS Blu Age ou le Tutoriel : Configuration du runtime géré pour Rocket Software (anciennement Micro Focus) pour un step-by-step didacticiel complet.

Si vous êtes intéressé par le refactoring automatique, consultez les outils AWS Blu Age sur. BluInsights Vous pouvez également configurer la AppStream version 2.0 pour accéder à l'IDE AWS Blu Age Developer ou aux outils Rocket Enterprise Analyzer (anciennement Micro Focus Enterprise Analyzer) et Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer).

Les didacticiels et les exemples d'applications ne vous donnent qu'une idée des avantages de la modernisation des AWS mainframes. Lorsque vous êtes prêt à démarrer un projet de modernisation, consultez Approche de modernisation pour plus de détails sur les étapes et les tâches d'un projet de modernisation.

Le schéma suivant montre le flux de travail du service de modernisation du AWS mainframe pour analyser, transformer, développer, tester, déployer et exploiter des applications mainframe.



Services connexes

Outre Blu Insights pour le refactoring automatisé, vous pouvez utiliser les AWS services suivants avec AWS Mainframe Modernization.

- Amazon RDS pour héberger vos bases de données migrées
- Amazon S3 pour le stockage des fichiers binaires et des fichiers de définition des applications
- Amazon FSx ou Amazon EFS pour le stockage des données d'application
- Amazon AppStream pour accéder aux outils Rocket Enterprise Analyzer et Rocket Enterprise Developer
- AWS CloudFormation pour le DevOps pipeline automatisé que vous pouvez utiliser CI/CD pour configurer vos applications migrées
- AWS Migration Hub
- AWS DMS pour la migration de vos bases de données

Accès à la AWS modernisation du mainframe

Actuellement, vous pouvez accéder à la modernisation AWS du mainframe via la console à https://console.aws.amazon.com/m2/l'adresse. Pour obtenir la liste des régions dans lesquelles la modernisation des AWS mainframes est disponible, consultez la section Points de terminaison et quotas de modernisation des AWS mainframes dans le. Référence générale d'Amazon Web Services

Utilisez-vous la modernisation du AWS mainframe pour la première fois ?

Si vous utilisez AWS Mainframe Modernization pour la première fois, nous vous recommandons de commencer par lire les sections suivantes :

- Commencez à moderniser votre AWS mainframe
- Configuration pour la modernisation du AWS mainframe

Tarification de la modernisation AWS du mainframe

AWS La modernisation du mainframe facture l'utilisation d'instances prenant en charge les environnements d'exécution gérés. En outre, la modernisation AWS du mainframe propose certains outils sans frais supplémentaires. Vous êtes responsable des frais engagés pour les autres AWS services que vous utilisez dans le cadre de la modernisation du AWS mainframe. AWS fournira un préavis de 30 jours avant que toute modification tarifaire ne prenne effet pour l'utilisation de la

modernisation du AWS mainframe. Pour plus d'informations, consultez la section Modernisation du mainframe avec AWS.

Avec AWS Blu Insights, vous payez pour l'utilisation du centre de transformation. Pour plus d'informations, consultez la section Tarification de la modernisation des AWS mainframes.

Configuration pour la modernisation du AWS mainframe

Avant de commencer à utiliser AWS Mainframe Modernization, vous ou votre administrateur devez vous inscrire à un compte Compte AWS, créer un utilisateur avec des paramètres administratifs et sécuriser vos utilisateurs IAM.

Rubriques

- Inscrivez-vous pour un Compte AWS
- Création d'un utilisateur doté d'un accès administratif

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Une partie de la procédure d'inscription consiste à recevoir un appel téléphonique ou un message texte et à saisir un code de vérification sur le clavier du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d’un accès à d’autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d’autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

   Pour obtenir des instructions, consultez Création d’un ensemble d’autorisations dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l’utilisateur AWS IAM Identity Center .

AWS Concepts de modernisation des ordinateurs centraux

AWS La modernisation du mainframe fournit des outils et des ressources pour vous aider à migrer, à moderniser et à exécuter les charges de travail du mainframe sur celui-ci. AWS Vous pouvez utiliser cette page pour comprendre les différents concepts de la modernisation des AWS mainframes, notamment les applications, la modernisation, les environnements, le replatforming, le refactoring et les moteurs d'exécution.

Rubriques

- Application
- Définition de l'application
- Tâche par lots
- Configuration
- Ensemble de données
- Environnement
- Modernisation de l'ordinateur central
- Parcours migratoire
- Point de montage
- Refactorisation automatisée
- Replateforme
- Ressource
- Moteur d'exécution

Application

Une charge de travail récurrente dans le cadre de la modernisation du AWS mainframe. Un ensemble de tâches par lots, de transactions interactives (CICS ou IMS) ou d'autres composants constitue une application. Vous définissez le champ d'application. Vous devez définir et spécifier tous les composants ou ressources nécessaires à la charge de travail, tels que les transactions CICS ou les tâches par lots.

Définition de l'application

Définition ou spécification des composants et des ressources nécessaires à une application (charge de travail du mainframe) exécutée dans le cadre de la modernisation du AWS mainframe. Il est important de séparer la définition de l'application elle-même, car il est possible de réutiliser la même définition pour plusieurs étapes (pré-production, production), représentées par différents environnements d'exécution.

Tâche par lots

Programme planifié configuré pour s'exécuter sans intervention de l'utilisateur. Dans AWS Mainframe Modernization, vous devez stocker à la fois les fichiers JCL et les fichiers binaires des tâches par lots dans un compartiment Amazon S3, et indiquer leur emplacement dans le fichier de définition de l'application. Lorsque vous exécutez une tâche par lots, AWS Mainframe Modernization indique les valeurs d'état suivantes :

Soumission

Le traitement par lots est en cours de soumission.

En attente

Le traitement par lots est en attente.

Expédition

Le traitement par lots est en cours d'expédition.

En cours d'exécution

Le traitement par lots est actuellement en cours d'exécution.

Annulation

Le traitement par lots est en cours d'annulation.

Annulée

Le traitement par lots est annulé.

Réussi

Le traitement par lots s'est terminé correctement.

Échec

Le traitement par lots a échoué.

Réussite avec avertissement

Le traitement par lots s'est terminé correctement et une erreur mineure a été signalée. Le code de condition de travail renvoyé dans le cadre de la GetBatchJobExecution réponse indique la cause de l'erreur.

Configuration

Caractéristiques d'un environnement ou d'une application. Les configurations d'environnement comprennent le type de moteur, la version du moteur, les modèles de disponibilité, les configurations de système de fichiers facultatives, etc.

Les configurations d'application peuvent être statiques ou dynamiques. Les configurations statiques ne changent que lorsque vous mettez à jour une application en déployant une nouvelle version. Les configurations dynamiques, qui constituent généralement une activité opérationnelle telle que l'activation ou la désactivation du suivi, changent dès que vous les mettez à jour.

Ensemble de données

Fichier contenant des données destinées à être utilisées par les applications.

Environnement

Combinaison nommée de ressources de AWS calcul, d'un moteur d'exécution et de détails de configuration créée pour héberger une ou plusieurs applications.

Modernisation de l'ordinateur central

Processus de migration des applications d'un environnement mainframe existant vers. AWS

Parcours migratoire

Le end-to-end processus de migration et de modernisation des applications existantes comprend généralement les phases suivantes : évaluation, mobilisation, migration et modernisation, et exploitation et optimisation.

Point de montage

Répertoire d'un système de fichiers qui permet d'accéder aux fichiers stockés dans ce système.

Refactorisation automatisée

Processus de modernisation des artefacts d'applications existants pour les exécuter dans un environnement cloud moderne. Cela peut inclure la conversion de code et de données. Pour plus d'informations, consultez la section AWS Mainframe Modernization Automated Refactor.

Replateforme

Processus de déplacement d'une application et de ses artefacts d'une plateforme informatique vers une autre. Pour plus d'informations, consultez la section AWS Mainframe Modernisation Replatform.

Ressource

Composant physique ou virtuel d'un système informatique.

Moteur d'exécution

Logiciel qui facilite le fonctionnement d'une application.

Approche de modernisation

La migration est complexe et comporte de nombreuses variables. AWS La modernisation du mainframe propose une approche évolutive qui offre des avantages à court terme en améliorant l'agilité et en offrant de nombreuses opportunités d'optimisation et d'innovation par la suite. En outre, la modernisation du AWS mainframe permet de simplifier le parcours tout en respectant les spécificités de l'entreprise et des activités de votre client. Les deux principales approches prises en charge par la modernisation des AWS mainframes sont le refactoring automatisé ou le replatforming. Le choix dépend de la situation de votre client.

Le refactoring automatisé utilise les outils AWS Blu Age pour convertir automatiquement le code, les données et les dépendances en langage moderne, en banque de données et en frameworks, tout en garantissant l'équivalence fonctionnelle avec les mêmes fonctions métier.

Le replatforming utilise les outils Rocket Software (anciennement Micro Focus) pour transformer les charges de travail du mainframe en services agiles sur. AWS

Vous pouvez envisager le processus de modernisation par étapes. La première étape comprend trois phases : évaluation, mobilisation, migration et modernisation. L'étape suivante comprend la phase d'exploitation et d'optimisation, au cours de laquelle vous pourrez identifier d'autres opportunités d'innovation.

Rubriques

- Phase d'évaluation
- Phase de mobilisation
- Phase de migration et de modernisation
- Phase d'exploitation et d'optimisation

Phase d'évaluation

Au niveau le plus élevé, la phase d'évaluation vise à déterminer si vous êtes prêt à effectuer la migration. Vous définissez une analyse de rentabilisation, puis vous formez votre équipe grâce à des ateliers et à une journée d'immersion (démonstrations et laboratoires) proposés par AWS. Les ateliers et les journées d'immersion abordent différents sujets. Ces tâches sont effectuées en dehors de la modernisation du AWS mainframe.

Phase de mobilisation

Dans la phase de mobilisation, vous démarrez votre projet par un lancement, puis vous exécutez un processus de découverte qui extrait les données de vos applications mainframe et les intègre dans un outil de migration. Vous identifiez les applications que vous souhaitez migrer et vous sélectionnez quelques applications à piloter. Vous affinez votre analyse de rentabilisation, rédigez votre plan de migration et décidez de la manière dont vous souhaitez gérer la sécurité et la conformité, la gouvernance des comptes et votre modèle opérationnel. Vous mettez en place un centre d'excellence dans le cloud avec les bonnes personnes de votre équipe. Vous exécutez les projets pilotes et vous documentez ce que vous avez appris. Vous affinez votre plan de migration et votre analyse de rentabilisation. La plupart de ces tâches sont effectuées en dehors de la modernisation du AWS mainframe.

Phase de migration et de modernisation

La phase de migration et de modernisation s'applique à chaque application et comprend plusieurs tâches, notamment l'affectation de personnes, l'exécution d'une découverte approfondie, la détermination de l'architecture d'application appropriée, la configuration des environnements d'exécution des applications AWS, le replatformage ou la refactorisation de votre code, l'intégration à d'autres systèmes et, bien sûr, les tests. À la fin de la phase, vous déployez les applications replateformes ou refactorisées en production et vous passez au nouveau système sur AWS. La plupart ou la totalité de ces tâches sont effectuées dans le cadre de la modernisation du AWS mainframe, dans un autre AWS service ou dans un outil auquel la modernisation du AWS mainframe donne accès.

Si vous souhaitez utiliser le refactoring automatique, consultez Blu Insights. AWS Blu Insights est désormais disponible AWS Management Console via l'authentification unique. Vous n'avez plus besoin de gérer des informations d'identification AWS Blu Insights distinctes. Vous pouvez accéder aux fonctionnalités de la base de code AWS AWS Blu Age et du centre de transformation directement depuis le AWS Management Console.

Pour migrer des données du mainframe vers AWS, nous recommandons le AWS SCT et le. AWS Database Migration Service Pour plus d'informations, consultez Qu'est-ce que l'outil AWS Schema Conversion Tool ? dans le guide de l'utilisateur du AWS Schema Conversion Tool et qu'est-ce qu'AWS Database Migration Service ? dans le guide de AWS Database Migration Service l'utilisateur.

Phase d'exploitation et d'optimisation

Dans la phase d'exploitation et d'optimisation, vous vous concentrez sur la surveillance de vos applications déployées, la gestion des ressources et la mise à jour de la sécurité et de la conformité. Vous évaluez également les opportunités d'optimisation des charges de travail migrées.

Commencez à moderniser votre AWS mainframe

Vous pouvez commencer à moderniser le AWS mainframe en suivant des didacticiels qui vous présentent le service et chaque moteur d'exécution.

Rubriques

- Tutoriel : Configuration d'un environnement d'exécution géré pour AWS Blu Age
- Tutoriel : Configuration du runtime géré pour Rocket Software (anciennement Micro Focus)

Pour continuer à apprendre, consultez les didacticiels suivants.

- Tutoriel : Configuration de la version Rocket Software (anciennement Micro Focus) pour l' BankDemo exemple d'application
- Tutoriel : Configuration d'un CI/CD pipeline à utiliser avec Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)

Tutoriel : Configuration d'un environnement d'exécution géré pour AWS Blu Age

Vous pouvez déployer une application modernisée AWS Blu Age dans un environnement d'exécution AWS Mainframe Modernization à l'aide d'une application de démonstration spécifiée dans ce didacticiel.

Rubriques

- Prérequis
- Étape 1 : Téléchargez l'application de démonstration
- Étape 2 : Création de la définition de l'application
- Étape 3 : Création d'un environnement d'exécution
- Étape 4 : Création d'une application
- Étape 5 : Déploiement d'une application
- Étape 6 : démarrer une application
- Étape 7 : Accédez à l'application
- Étape 8 : Tester l'application

- Nettoyage des ressources

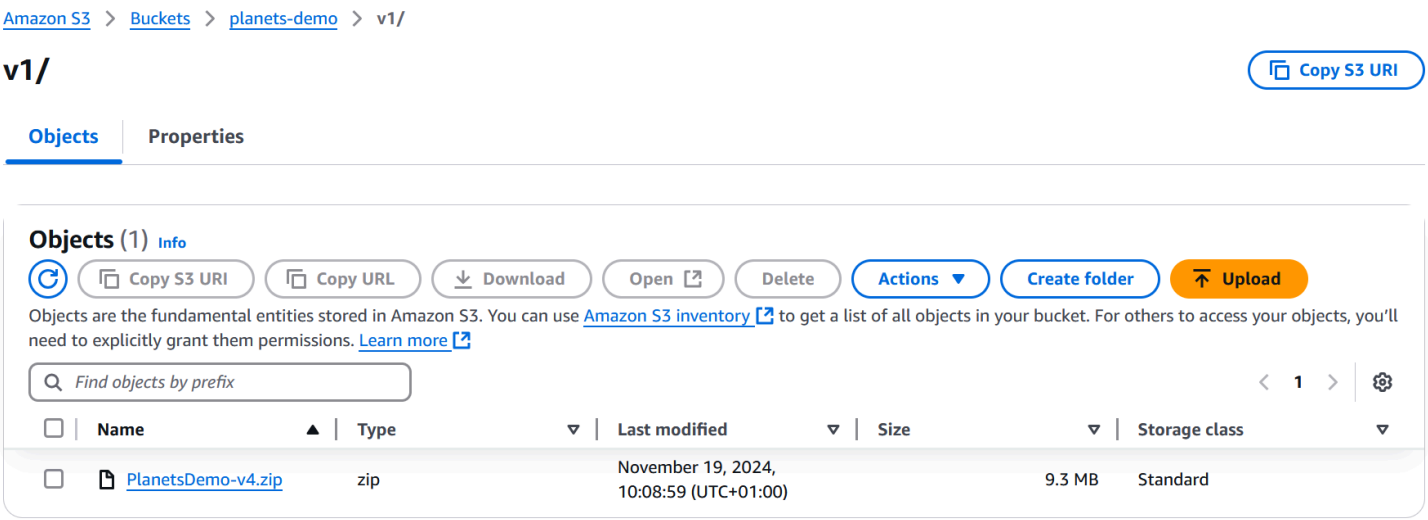
Prérequis

Pour terminer ce didacticiel, téléchargez l'archive PlanetsDemo-v4.zip de l'application de démonstration.

L'application de démonstration en cours d'exécution nécessite un navigateur moderne pour y accéder. Le fait que vous exécutiez ce navigateur depuis votre bureau ou depuis une instance Amazon Elastic Compute Cloud, par exemple au sein du VPC, détermine vos paramètres de sécurité.

Étape 1 : Téléchargez l'application de démonstration

Téléchargez l'application de démonstration dans un compartiment Amazon S3. Assurez-vous que ce compartiment se trouve dans le même compartiment que celui dans Région AWS lequel vous allez déployer l'application. L'exemple suivant montre un bucket nommé planets-demo, avec un préfixe de clé, ou dossier, nommé v1 et une archive nommée. `planetsdemo-v4.zip`



> **Note**
>
> Le dossier du compartiment est obligatoire.

Étape 2 : Création de la définition de l'application

Pour déployer une application sur le runtime géré, vous avez besoin d'une définition de l'application AWS Mainframe Modernization. Cette définition est un fichier JSON qui décrit l'emplacement et les

paramètres de l'application. L'exemple suivant est une telle définition d'application pour l'application de démonstration :

```
{
    "template-version": "2.0",
    "source-locations": [{
        "source-id": "s3-source",
        "source-type": "s3",
        "properties": {
            "s3-bucket": "planets-demo",
            "s3-key-prefix": "v1"
        }
    }],
    "definition": {
        "listeners": [{
            "port": 8196,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/PlanetsDemo-v4.zip"
        }
    }
}
```

Remplacez l'`s3-bucket`entrée par le nom du fichier zip de l'exemple d'application (par exemple,`planets-demo`) et l'`app-location`entrée par le chemin S3 dans lequel vous avez stocké le fichier zip de l'exemple d'application (par exemple,`${s3-source}/PlanetsDemo-v4.zip`).

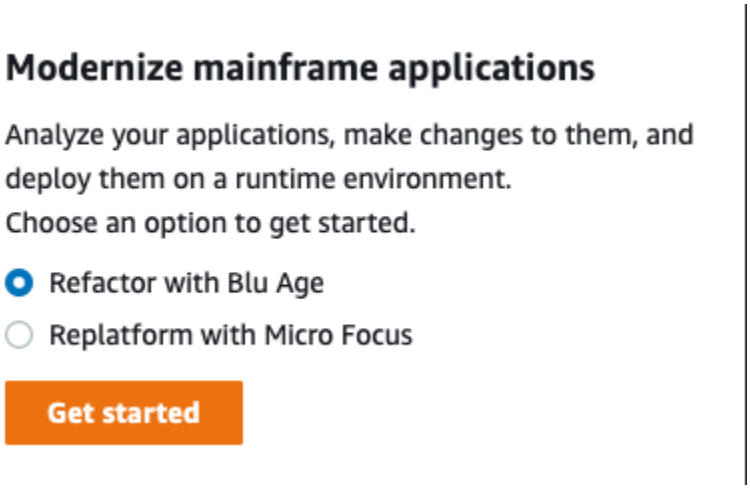
> **Note**
>
> Assurez-vous de créer le fichier de définition de l'application sur votre ordinateur local sous forme de fichier texte.

Pour plus d'informations sur la définition de l'application, consultezAWS Exemple de définition d'application Blu Age.

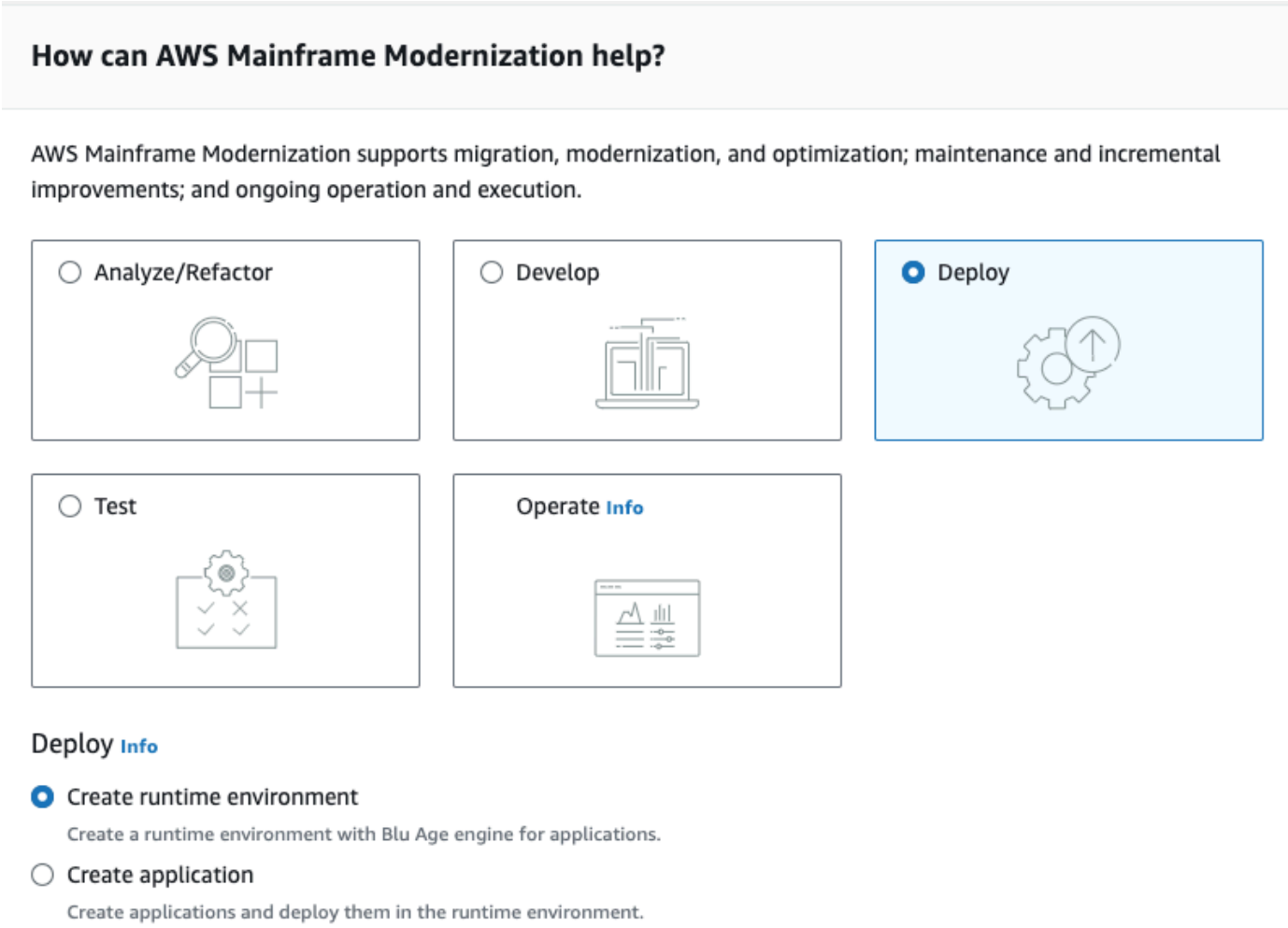
Étape 3 : Création d'un environnement d'exécution

Pour créer l'environnement d'exécution AWS Mainframe Modernization, effectuez les opérations suivantes :

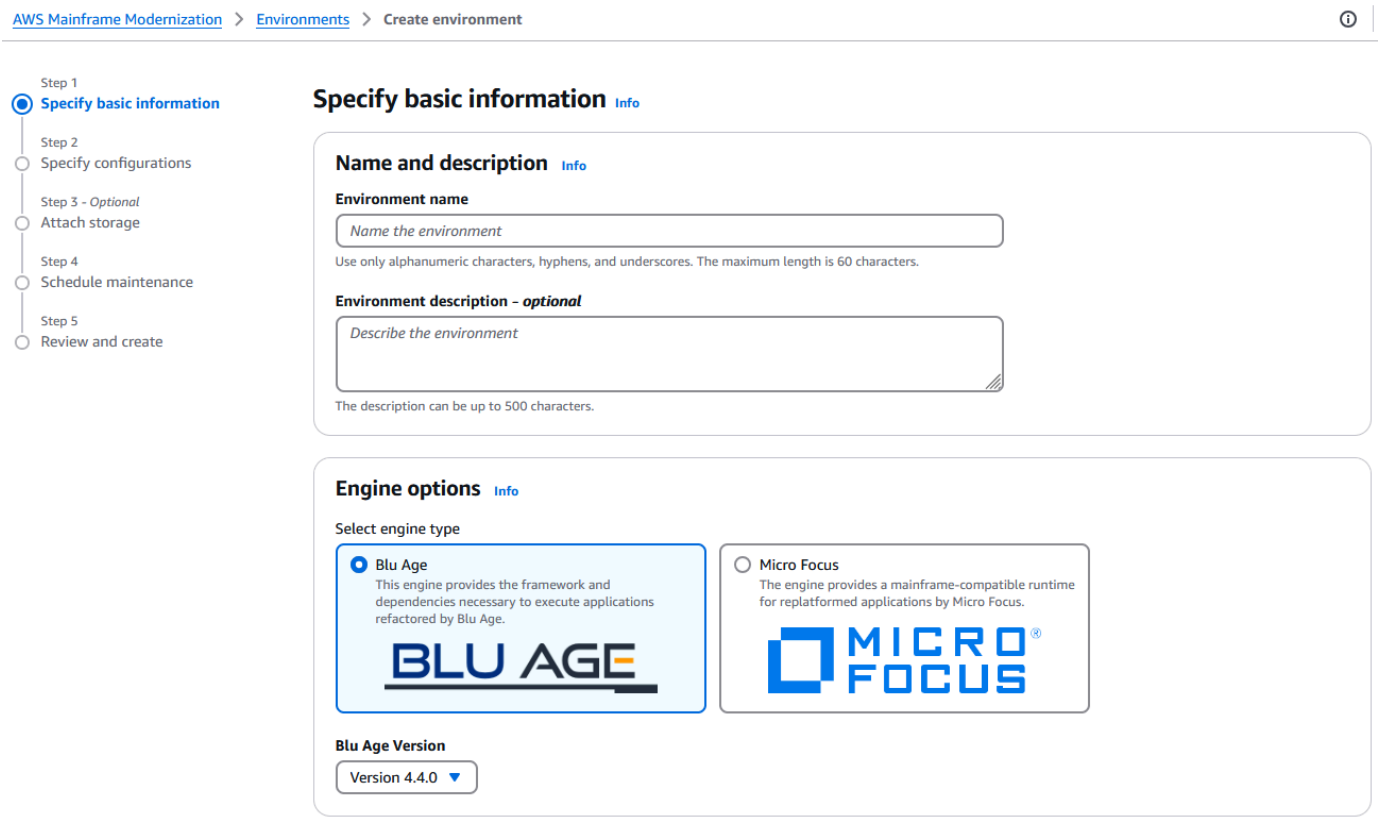
1. Ouvrez la console de modernisation AWS du mainframe.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'environnement. Cela Région AWS doit correspondre à la région dans laquelle vous avez créé le compartiment S3Étape 1 : Téléchargez l'application de démonstration.
3. Sous Moderniser les applications mainframe, choisissez Refactor with Blu Age, puis choisissez Get started.



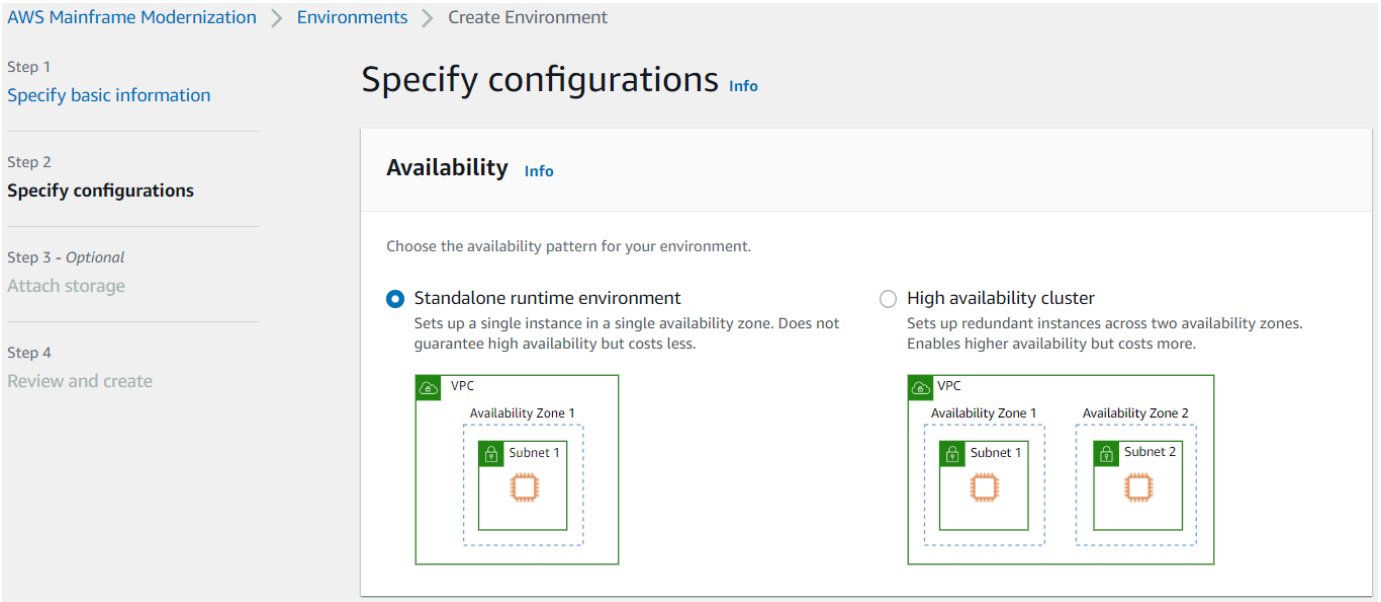
4. Dans la section Comment la modernisation du mainframe AWS peut-elle vous aider, choisissez Déployer et créer un environnement d'exécution.



5. Dans le volet de navigation de gauche, choisissez Environments, puis Create environment. Sur la page Spécifier les informations de base, entrez le nom et la description de votre environnement, puis assurez-vous que le moteur AWS Blu Age est sélectionné. Vous pouvez éventuellement ajouter des balises à la ressource créée. Ensuite, sélectionnez Suivant.



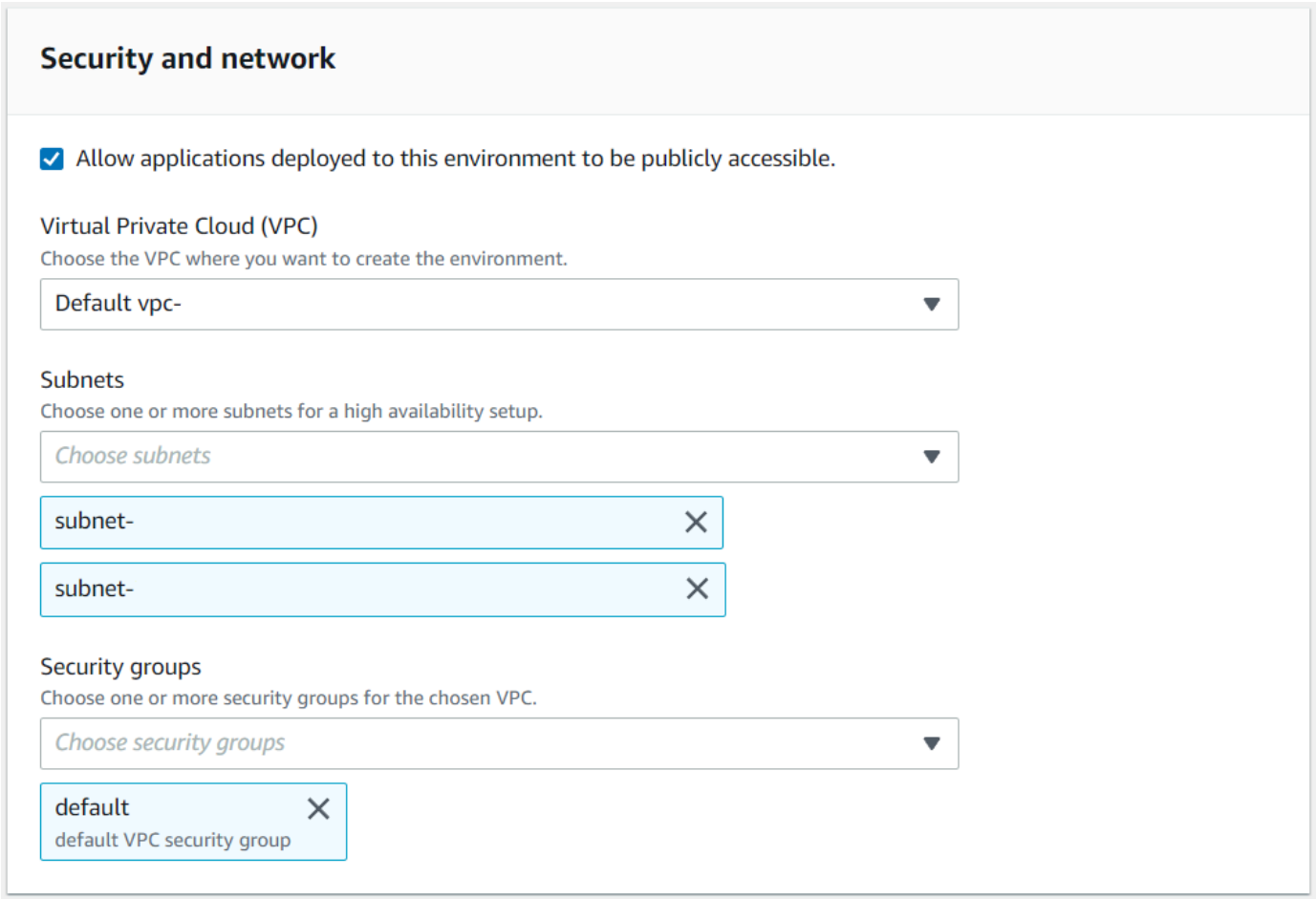
6. Sur la page Spécifier les configurations, sélectionnez Environnement d'exécution autonome.



7. Sous Sécurité et réseau, apportez les modifications suivantes :

   - Choisissez Autoriser les applications déployées dans cet environnement à être accessibles au public. Cette option attribue une adresse IP publique à l'application afin que vous puissiez y accéder depuis votre bureau.

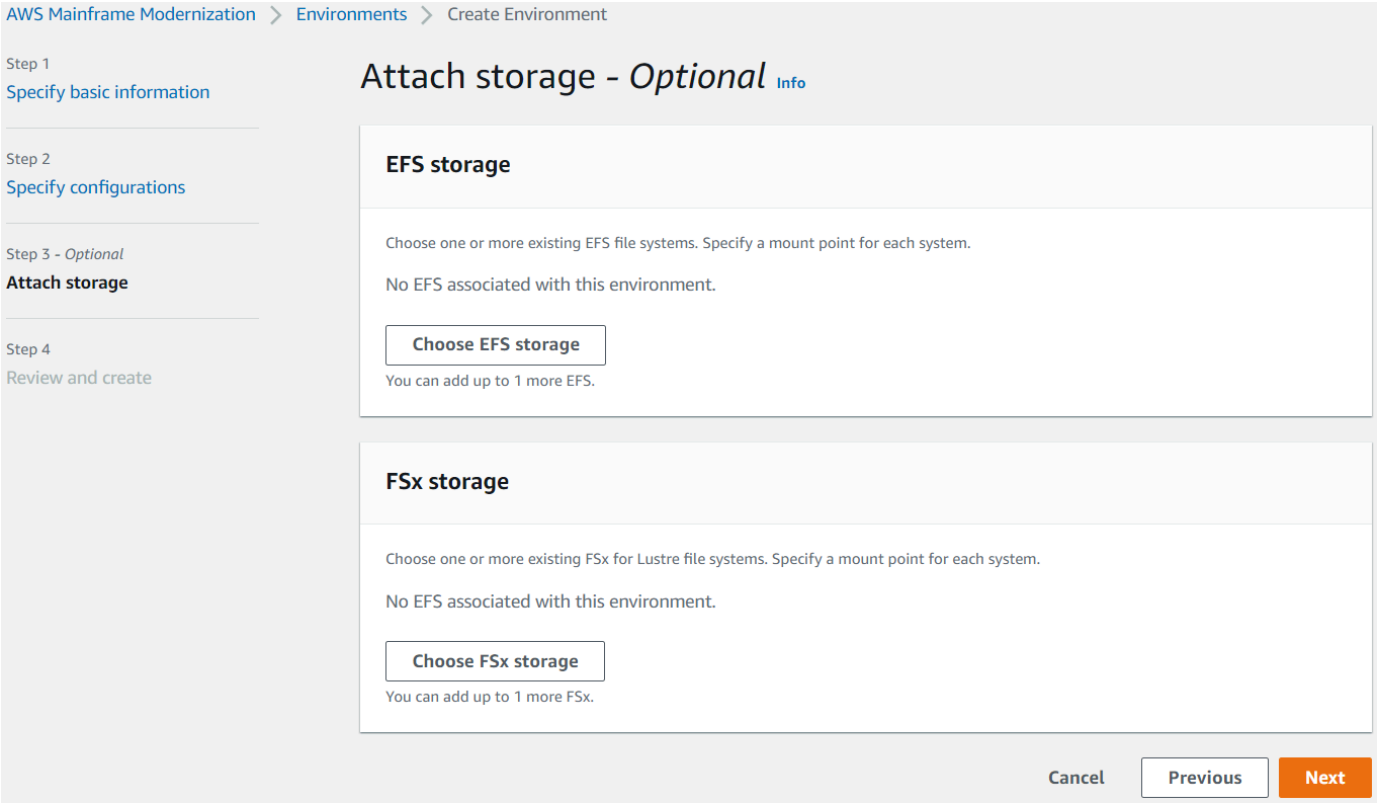
- Choisissez un VPC. Vous pouvez utiliser la valeur par défaut.
- Choisissez deux sous-réseaux. Assurez-vous que les sous-réseaux autorisent l'attribution d'adresses IP publiques.
- Sélectionnez un groupe de sécurité. Vous pouvez utiliser la valeur par défaut. Assurez-vous que le groupe de sécurité que vous choisissez autorise l'accès depuis l'adresse IP du navigateur au port que vous avez spécifié dans la `listener` propriété de la définition de l'application. Pour de plus amples informations, veuillez consulter Étape 2 : Création de la définition de l'application.



Si vous souhaitez accéder à l'application depuis l'extérieur du VPC que vous avez choisi, assurez-vous que les règles de trafic entrant pour ce VPC sont correctement configurées. Pour de plus amples informations, veuillez consulter Erreur de résolution des problèmes : Impossible d'accéder à l'URL d'une application.

8. Choisissez Next (Suivant).

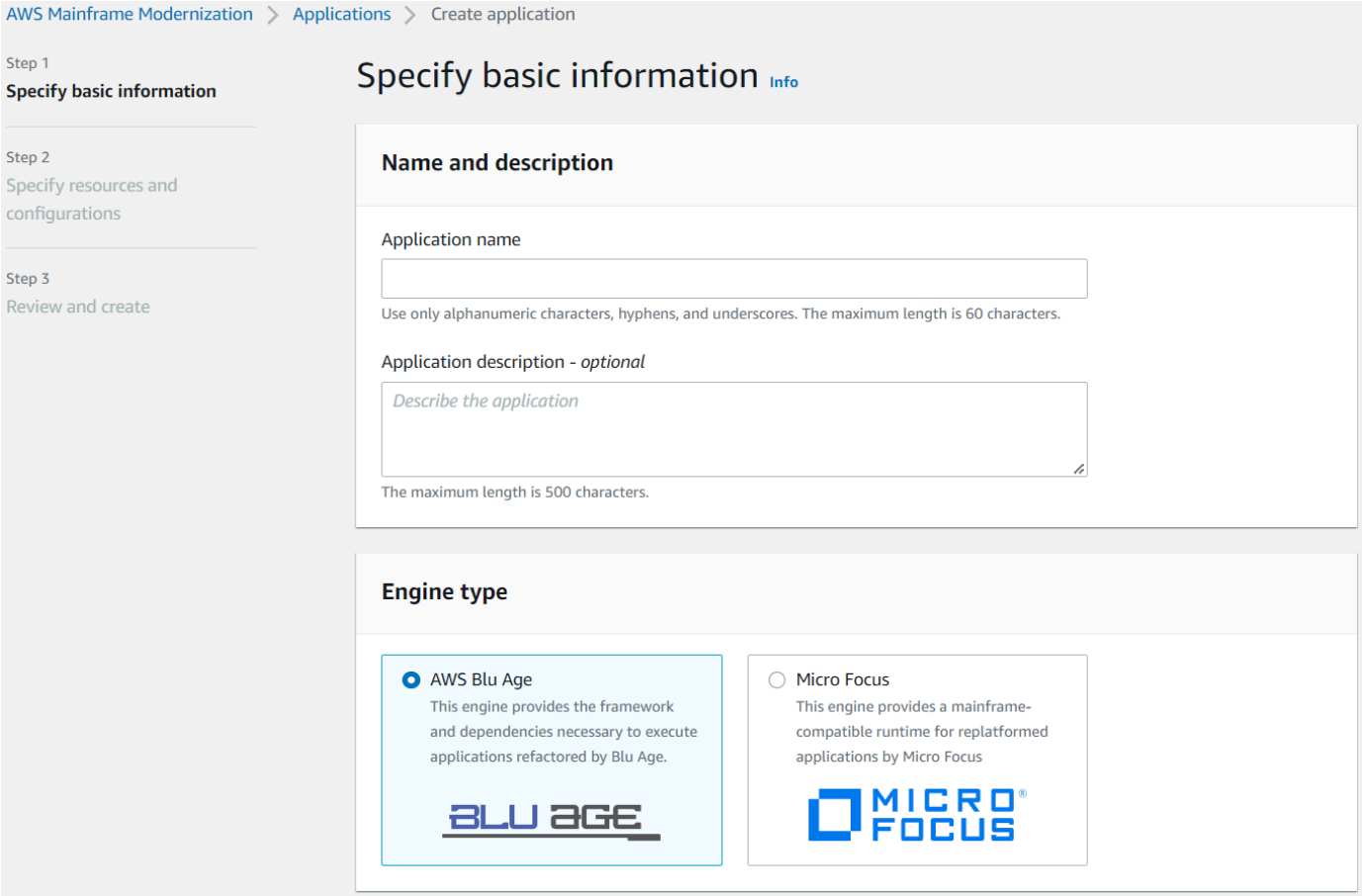
9. Dans Joindre un espace de stockage - Facultatif, laissez les sélections par défaut et choisissez Suivant.



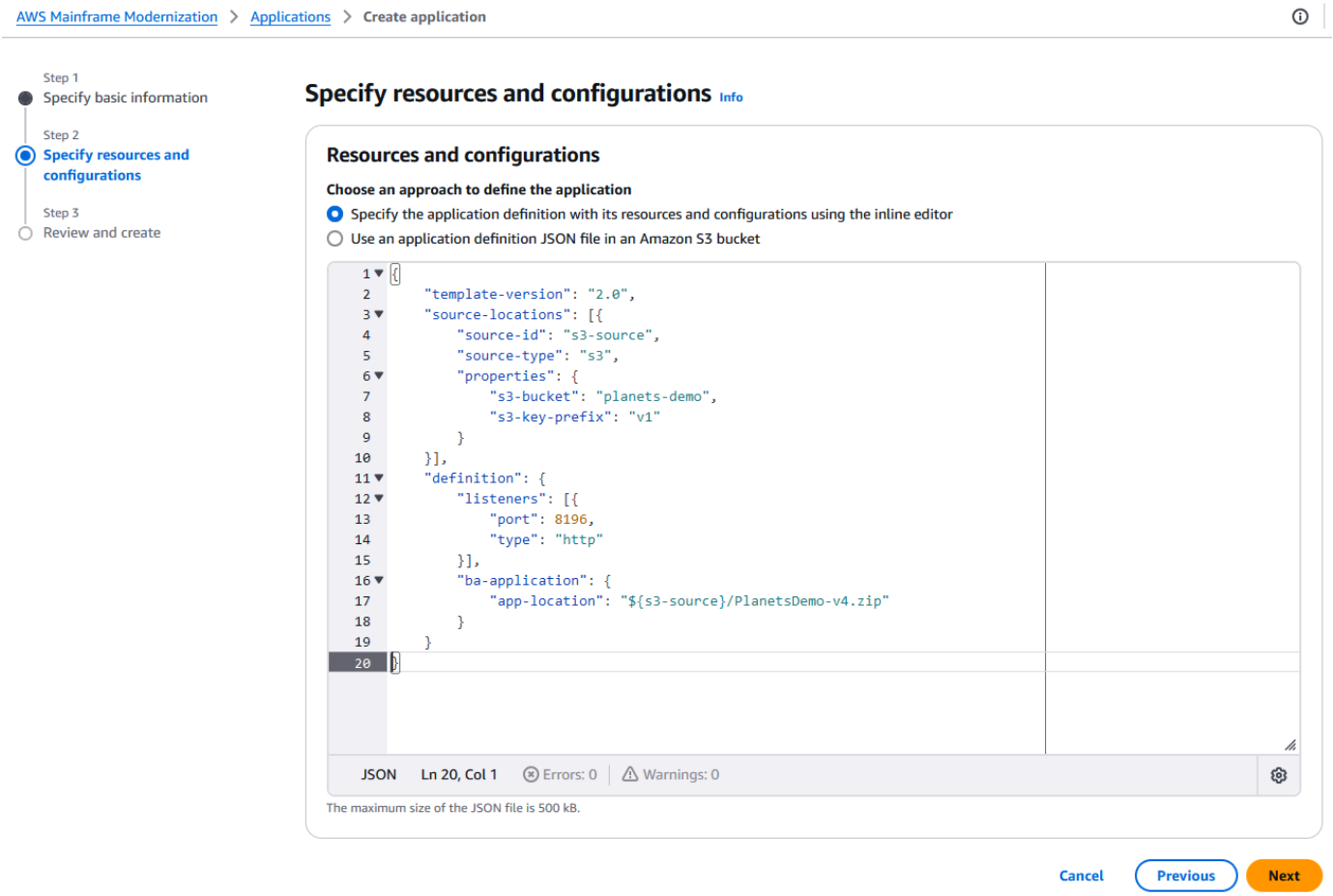
10. Dans Planifier la maintenance, choisissez Aucune préférence, puis cliquez sur Suivant.
11. Dans Révision et création, passez en revue les informations, puis choisissez Créer un environnement.

Étape 4 : Création d'une application

1. Accédez à la modernisation du mainframe AWS dans le AWS Management Console.
2. Dans le volet de navigation, choisissez Applications (Applications), puis Create a new application (Créer une nouvelle application). Sur la page Spécifier les informations de base, entrez le nom et la description de l'application et assurez-vous que le moteur AWS Blu Age est sélectionné. Ensuite, sélectionnez Suivant.



3. Sur la page Spécifier les ressources et les configurations, copiez et collez le JSON de définition d'application mis à jour dans lequel vous l'avez crééethe section called "Étape 2 : Création de la définition de l'application".



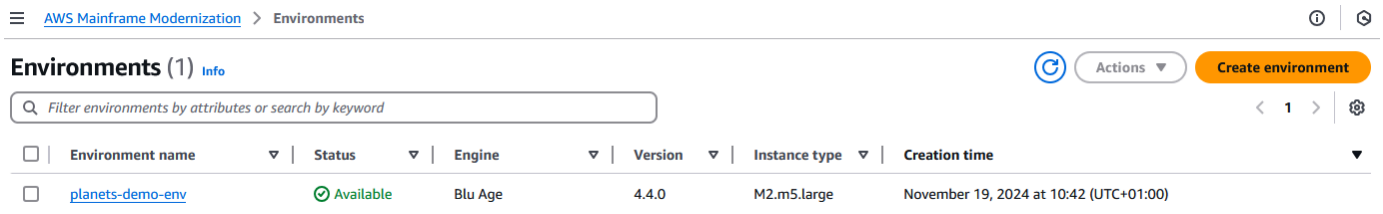
4. Dans Révision et création, passez en revue vos choix, puis choisissez Créer une application.

> **Note**
>
> Si la création de votre application échoue, vérifiez le chemin S3 que vous avez saisi car il distingue les majuscules et minuscules.

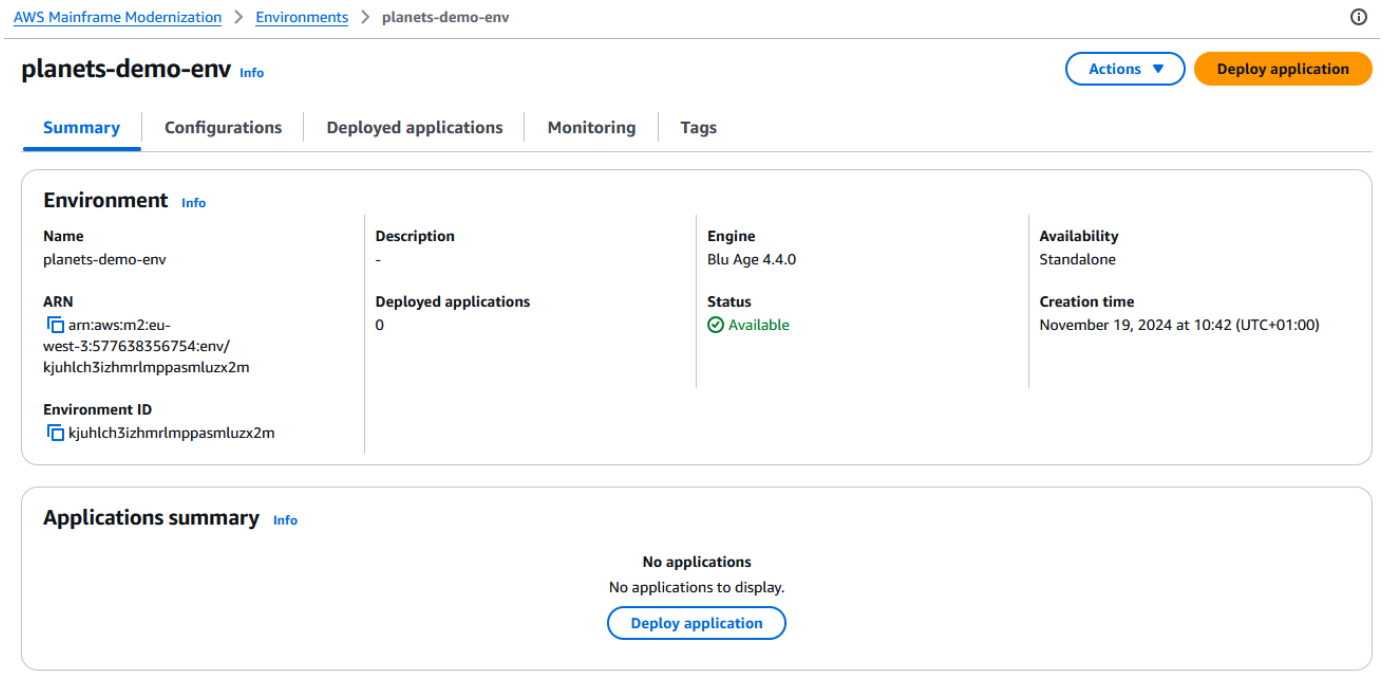
Étape 5 : Déploiement d'une application

Une fois que vous avez créé avec succès l'environnement d'exécution et l'application AWS Mainframe Modernization, et que les deux sont dans l'état Disponible, vous pouvez déployer l'application dans l'environnement d'exécution. Pour y arriver, exécutez les étapes suivantes.

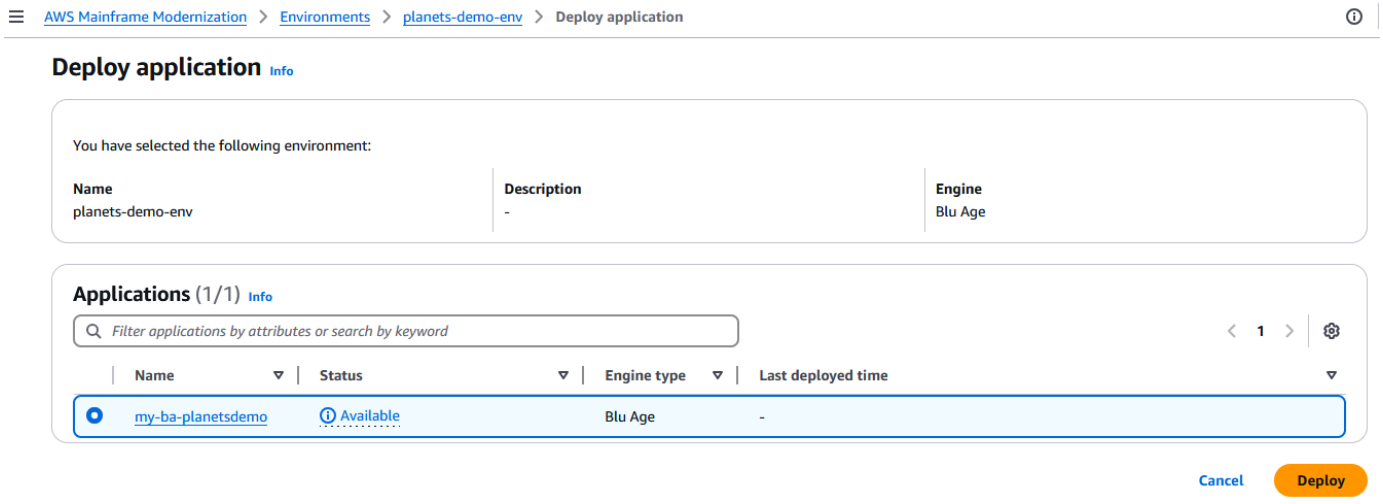
1. Accédez à la modernisation du mainframe AWS dans la console AWS de gestion. Dans le panneau de navigation, choisissez Environments (Environnements). La page de liste des environnements s'affiche.



2. Choisissez l'environnement d'exécution créé précédemment. La page des détails de l'environnement s'affiche.
3. Choisissez Déployer l'application.



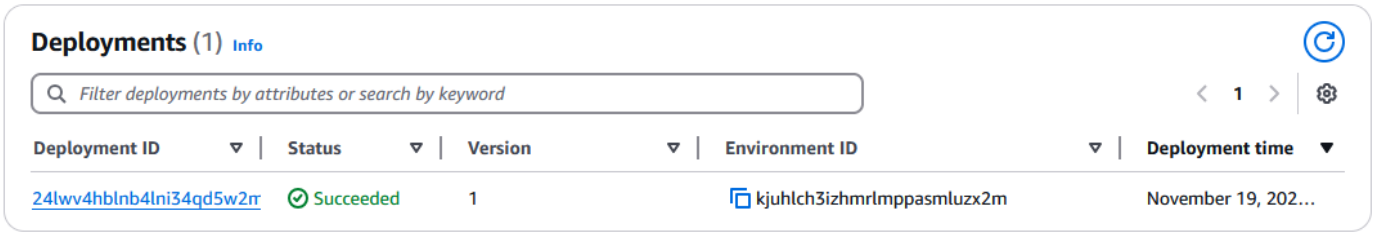
4. Choisissez l'application créée précédemment, puis la version vers laquelle vous souhaitez déployer votre application. Choisissez ensuite Deploy (Déployer).



5. Patientez jusqu'à ce que le déploiement de l'application soit terminé. Vous verrez une bannière avec le message L'application a été déployée avec succès.

Étape 6 : démarrer une application

1. Accédez à AWS Mainframe Modernization dans le AWS Management Console et choisissez Applications.
2. Choisissez votre application, puis accédez à Déploiements. Le statut de la demande doit être Réussi.



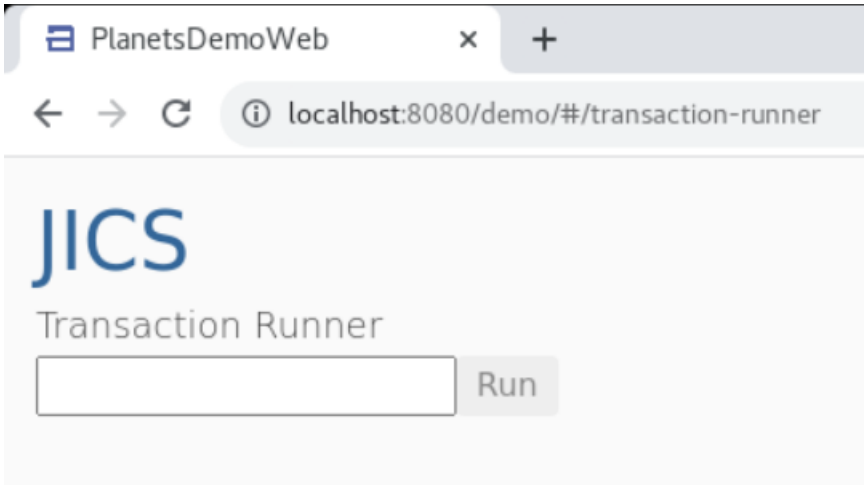
3. Choisissez Actions, puis sélectionnez Démarrer l'application.

Étape 7 : Accédez à l'application

1. Attendez que l'application soit en cours d'exécution. Vous verrez une bannière avec le message L'application a été démarrée avec succès.
2. Copiez le nom d'hôte DNS de l'application. Vous trouverez ce nom d'hôte dans la section Informations sur l'application de l'application.
3. Dans un navigateur, naviguez jusqu'à`http://{hostname}:{portname}/PlanetsDemo-web-1.0.0/`, où :

   - `hostname`est le nom d'hôte DNS copié précédemment.
   - `portname`est le port Tomcat défini dans la définition de l'application que vous avez créée dansÉtape 2 : Création de la définition de l'application.

   L'écran JICS apparaît.



Si vous ne parvenez pas à accéder à l'application, consultezErreur de résolution des problèmes : Impossible d'accéder à l'URL d'une application.

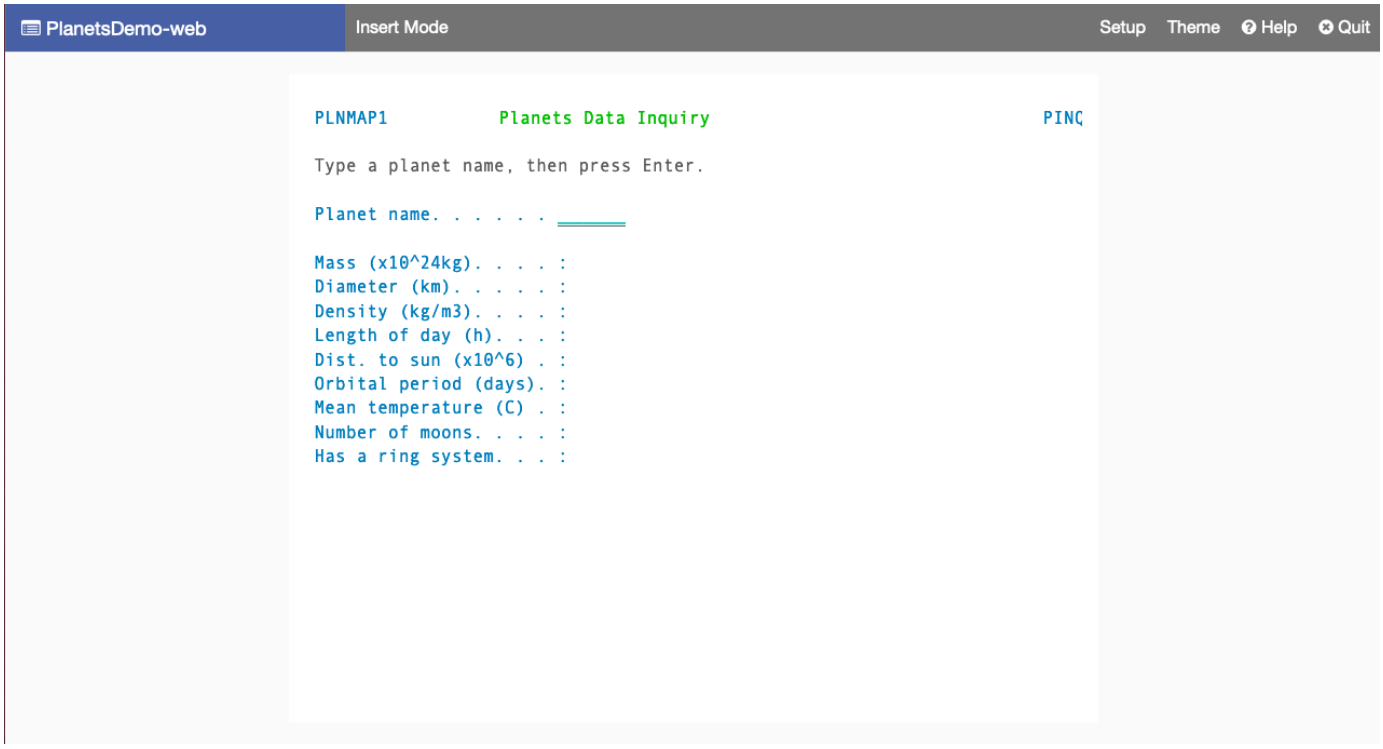
> **Note**
>
> Si l'application n'est pas accessible et que la règle entrante sur le groupe de sécurité indique que « Mon adresse IP » est sélectionnée sur le port 8196, spécifiez la règle pour autoriser le trafic depuis LB i/p sur le port 8196.

Étape 8 : Tester l'application

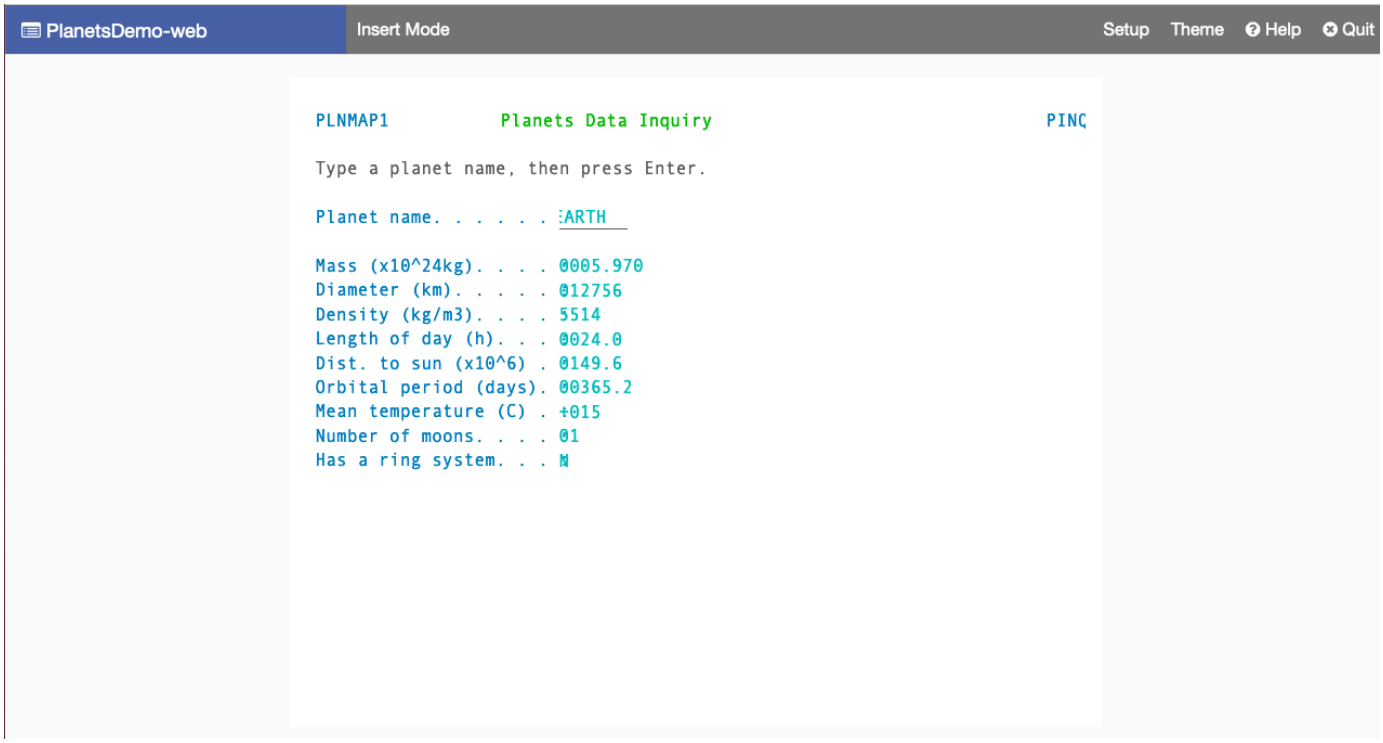
Au cours de cette étape, vous exécutez une transaction dans l'application migrée.

1. Sur l'écran JICS, entrez `PINQ` dans le champ de saisie et choisissez Exécuter (ou appuyez sur Entrée) pour démarrer la transaction de l'application.

   L'écran de l'application de démonstration devrait apparaître.



2. Tapez le nom d'une planète dans le champ correspondant et appuyez sur Entrée.



Vous devriez voir des informations sur la planète.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Si l'application AWS Mainframe Modernization est toujours en cours d'exécution, arrêtez-la.
- Supprimez l'application . Pour de plus amples informations, veuillez consulter Supprimer une AWS Mainframe Modernization application.
- Supprimez l'environnement d'exécution. Pour de plus amples informations, veuillez consulter Supprimer un environnement d'exécution de modernisation du AWS mainframe.

Tutoriel : Configuration du runtime géré pour Rocket Software (anciennement Micro Focus)

Vous pouvez déployer et exécuter une application dans un environnement d'exécution géré AWS Mainframe Modernization avec le moteur d'exécution Rocket Software. Ce didacticiel explique comment déployer et exécuter l' CardDemo exemple d'application dans un environnement d'exécution géré AWS Mainframe Modernization avec le moteur d'exécution Rocket Software. L' CardDemo exemple d'application est une application de carte de crédit simplifiée développée pour tester AWS et présenter des technologies partenaires dans le cadre de cas d'utilisation de la modernisation des ordinateurs centraux.

Dans le didacticiel, vous créez des ressources dans d'autres Services AWS. Il s'agit notamment d'Amazon Simple Storage Service, d'Amazon Relational Database Service AWS Key Management Service et. AWS Secrets Manager

Rubriques

- Prérequis
- Étape 1 : créer et charger un compartiment Amazon S3
- Étape 2 : Création et configuration d'une base de données
- Étape 3 : Création et configuration d'un AWS KMS key
- Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données
- Étape 5 : ajouter le SSLMode au secret
- Étape 6 : Création d'un environnement d'exécution
- Étape 7 : Création d'une application

- Étape 8 : Déploiement d'une application
- Étape 9 : Importer des ensembles de données
- Étape 10 : démarrer une application
- Étape 11 : Connectez-vous à l'application CardDemo CICS
- Nettoyage des ressources
- Étapes suivantes

Prérequis

- Assurez-vous d'avoir accès à un émulateur 3270 pour utiliser la connexion CICS. Des émulateurs 3270 gratuits et d'essai sont disponibles sur des sites Web tiers. Vous pouvez également démarrer une instance du logiciel Rocket AWS Mainframe Modernization AppStream 2.0 et utiliser l'émulateur Rumba 3270 (non disponible gratuitement).

  Pour plus d'informations sur la AppStream version 2.0, consultezthe section called "Tutoriel : Configuration de la AppStream version 2.0 pour Enterprise Analyzer et Enterprise Developer".

  > **Note**
  >
  > Lors de la création de la pile, choisissez l'option Enterprise Developer (ED) et non Enterprise Analyzer (EA).

- Téléchargez l'CardDemo exemple d'application et décompressez le fichier téléchargé dans n'importe quel répertoire local. Ce répertoire contiendra un sous-répertoire intitulé`CardDemo_runtime`.
- Identifiez un VPC dans votre compte dans lequel vous pourrez définir les ressources créées dans ce didacticiel. Le VPC aura besoin de sous-réseaux dans au moins deux zones de disponibilité. Pour plus d'informations sur Amazon VPC, consultez Comment fonctionne Amazon VPC.

Étape 1 : créer et charger un compartiment Amazon S3

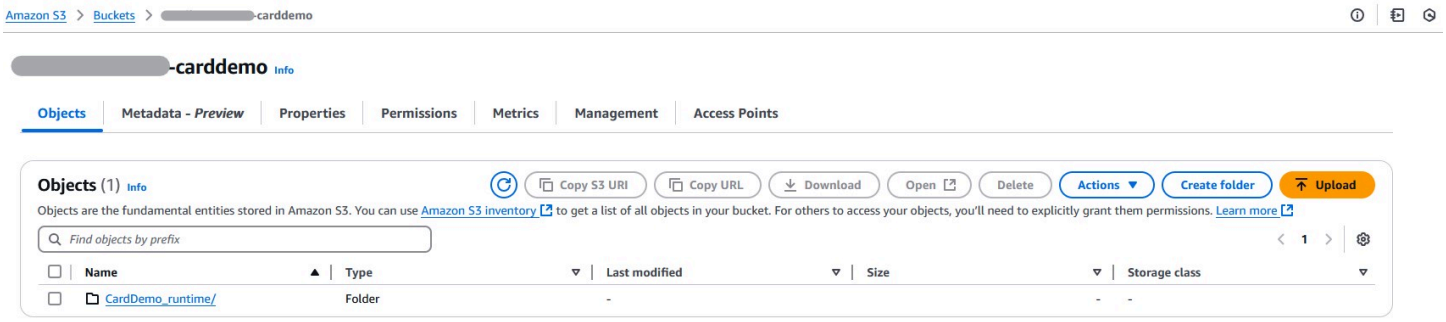
Au cours de cette étape, vous créez un compartiment Amazon S3 et chargez CardDemo des fichiers dans ce compartiment. Plus loin dans ce didacticiel, vous utiliserez ces fichiers pour déployer et exécuter l' CardDemo exemple d'application dans un environnement d'exécution géré par le logiciel AWS Mainframe Modernization Rocket.

> **Note**
>
> Il n'est pas nécessaire de créer un nouveau compartiment S3, mais le compartiment que vous choisissez doit se trouver dans la même région que les autres ressources utilisées dans ce didacticiel.

Pour créer un compartiment Amazon S3

1. Ouvrez la console Amazon S3 et choisissez Create bucket.
2. Dans Configuration générale, choisissez la région AWS dans laquelle vous souhaitez créer le AWS Mainframe Modernization Rocket Software Managed Runtime.
3. Entrez un nom de compartiment, par exemple,`yourname-aws-region-carddemo`. Conservez les paramètres par défaut et choisissez Create bucket. Vous pouvez également copier les paramètres d'un compartiment Amazon S3 existant, puis choisir Create bucket.
4. Choisissez le bucket que vous venez de créer, puis choisissez Upload.
5. Dans la section Télécharger, choisissez Ajouter un dossier, puis accédez au `CardDemo_runtime` répertoire depuis votre ordinateur local.
6. Choisissez Upload pour démarrer le processus de téléchargement. Les temps de téléchargement varient en fonction de la vitesse de votre connexion.
7. Lorsque le téléchargement est terminé, vérifiez que tous les fichiers ont été correctement chargés, puis choisissez Fermer.

Votre compartiment Amazon S3 contient désormais le `CardDemo_runtime` dossier.



Pour plus d'informations sur les compartiments S3, consultez Création, configuration et utilisation des compartiments Amazon S3.

Étape 2 : Création et configuration d'une base de données

Au cours de cette étape, vous allez créer une base de données PostgreSQL dans Amazon Relational Database Service (Amazon RDS). Pour le didacticiel, cette base de données contient les ensembles de données que l' CardDemo exemple d'application utilise pour les tâches des clients concernant les transactions par carte de crédit.

Pour créer une base de données dans Amazon RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez la région AWS dans laquelle vous souhaitez créer l'instance de base de données.
3. Dans le volet de navigation, sélectionnez Databases (Bases de données).
4. Choisissez Créer une base de données, puis sélectionnez Création standard.
5. Pour Type de moteur, choisissez PostgreSQL.
6. Choisissez une version du moteur 15 ou supérieure.

   > **Note**
   >
   > Enregistrez la version du moteur car vous en aurez besoin plus tard dans ce didacticiel.

7. Dans Modèles, choisissez Offre gratuite.
8. Remplacez l'identifiant de l'instance de base de données par quelque chose de significatif, par exemple`MicroFocus-Tutorial`.
9. Abstenez-vous de gérer les informations d'identification principales dans AWS Secrets Manager. Entrez plutôt un mot de passe principal et confirmez-le.

   > **Note**
   >
   > Enregistrez le nom d'utilisateur et le mot de passe que vous utilisez pour la base de données. Vous les stockerez en toute sécurité dans les prochaines étapes de ce didacticiel.

10. Sous Connectivité, choisissez le VPC sur lequel vous souhaitez créer l'environnement d'exécution géré AWS Mainframe Modernization.
11. Choisissez Créer une base de données.

Pour créer un groupe de paramètres personnalisé dans Amazon RDS

1. Dans le volet de navigation de la console Amazon RDS, sélectionnez Groupes de paramètres, puis sélectionnez Créer un groupe de paramètres.
2. Dans la fenêtre Créer un groupe de paramètres, pour Famille de groupes de paramètres, sélectionnez l'option Postgres correspondant à la version de votre base de données.

   > **Note**
   >
   > Certaines versions de Postgres nécessitent un type. Sélectionnez le groupe de paramètres de base de données si nécessaire. Entrez un nom de groupe et une description pour le groupe de paramètres.

3. Sélectionnez Créer.

Pour configurer le groupe de paramètres personnalisé

1. Choisissez le groupe de paramètres nouvellement créé.
2. Sélectionnez Actions, puis Edit (Modifier).
3. Filtrez `max_prepared_transactions` et remplacez la valeur du paramètre par 100.
4. Choisissez Save Changes (Enregistrer les modifications).

Pour associer le groupe de paramètres personnalisés à la base de données

1. Dans le volet de navigation de la console Amazon RDS, choisissez Databases, puis choisissez l'instance de base de données que vous souhaitez modifier.
2. Sélectionnez Modifier. La page Modifier l'instance de base de données s'affiche.

   > **Note**
   >
   > L'option Modifier n'est pas disponible tant que la création et la sauvegarde de la base de données ne sont pas terminées, ce qui peut prendre plusieurs minutes.

3. Sur la page Modifier une instance de base de données, accédez à Configuration supplémentaire et remplacez le groupe de paramètres de base de données par votre groupe de paramètres. Si votre groupe de paramètres n'est pas disponible dans la liste, vérifiez s'il a été créé avec la bonne version de base de données.

4. Choisissez Continuer, puis consultez le résumé des modifications.
5. Choisissez Appliquer immédiatement pour appliquer les modifications instantanément.
6. Choisissez Modifier l'instance de base de données pour enregistrer vos modifications.

Pour plus d'informations sur les groupes de paramètres, consultez la section Utilisation des groupes de paramètres.

> **Note**
>
> Vous pouvez également utiliser une base de données Amazon Aurora PostgreSQL AWS avec Mainframe Modernization, mais il n'existe pas d'option de niveau gratuit. Pour plus d'informations, consultez la section Utilisation d'Amazon Aurora PostgreSQL.

Étape 3 : Création et configuration d'un AWS KMS key

Pour stocker les informations d'identification de manière sécurisée pour l'instance Amazon RDS, créez d'abord un AWS KMS key.

Pour créer un AWS KMS key

1. Ouvrez la console du service de gestion des clés.
2. Choisissez Create key (Créer une clé).
3. Conservez les valeurs par défaut Symetric pour le type de clé et Chiffrer et déchiffrer pour l'utilisation des clés.
4. Choisissez Next (Suivant).
5. Donnez à la clé un alias `MicroFocus-Tutorial-RDS-Key` et une description facultative.
6. Choisissez Next (Suivant).
7. Attribuez un administrateur clé en cochant la case à côté de votre utilisateur ou de votre rôle.
8. Choisissez Next (Suivant).
9. Attribuez une autorisation d'utilisation clé en cochant la case à côté de votre utilisateur ou de votre rôle.
10. Choisissez Next (Suivant).
11. Sur l'écran de révision, modifiez la politique clé, puis entrez ce qui suit dans le tableau « Déclaration » existant :

```
{
     "Sid" : "Allow access for Mainframe Modernization Service",
     "Effect" : "Allow",
        "Principal" : {
           "Service" : "m2.amazonaws.com"
                 },
      "Action" : "kms:Decrypt",
      "Resource" : "*"
},
```

Cette politique accorde à AWS Mainframe Modernization des autorisations de déchiffrement à l'aide de cette politique clé spécifique.

12. Choisissez Terminer pour créer la clé.

Pour plus d'informations, consultez la section Création de clés dans le guide du AWS Key Management Service développeur.

Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données

Stockez maintenant les informations d'identification de la base de données en toute sécurité à l'aide du AWS Secrets Manager et AWS KMS key.

Pour créer et configurer un secret AWS Secrets Manager de base de données

1. Ouvrez la console Secrets Manager.
2. Dans le volet de navigation, sélectionnez Secrets.
3. Dans Secrets, choisissez Enregistrer un nouveau secret.
4. Définissez le type de secret sur Identifiants pour la base de données Amazon RDS.
5. Entrez les informations d'identification que vous avez spécifiées lors de la création de la base de données.
6. Sous Clé de chiffrement, sélectionnez la clé que vous avez créée à l'étape 3.
7. Dans la section Base de données, sélectionnez la base de données que vous avez créée pour ce didacticiel, puis choisissez Next.
8. Sous Nom secret, entrez un nom tel qu'`MicroFocus-Tutorial-RDS-Secret`une description facultative.

9. Dans la section Autorisations relatives aux ressources, choisissez Modifier les autorisations et remplacez le contenu par la politique suivante :

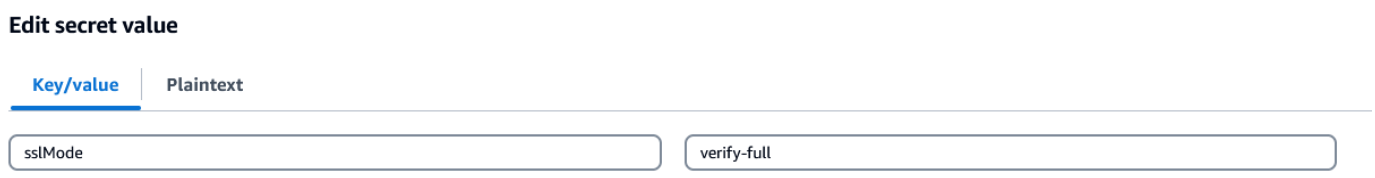
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Principal" : {
                "Service" : "m2.amazonaws.com"
            },
            "Action" : "secretsmanager:GetSecretValue",
            "Resource" : "*"
        }
    ]
}
```

10. Choisissez Enregistrer.
11. Choisissez Next pour les écrans suivants, puis sélectionnez Store.

Étape 5 : ajouter le SSLMode au secret

Pour ajouter le SSLMode au secret

1. Actualisez la liste des secrets pour voir le nouveau secret.
2. Choisissez le secret nouvellement créé à l'étape 4 et notez-le `Secret ARN` car vous en aurez besoin plus tard dans le didacticiel.
3. Dans l'onglet Vue d'ensemble du secret, choisissez Extraire la valeur du secret.
4. Choisissez Modifier, puis Ajouter une ligne.
5. Ajoutez une clé pour `sslMode` avec une valeur de `verify-full` :

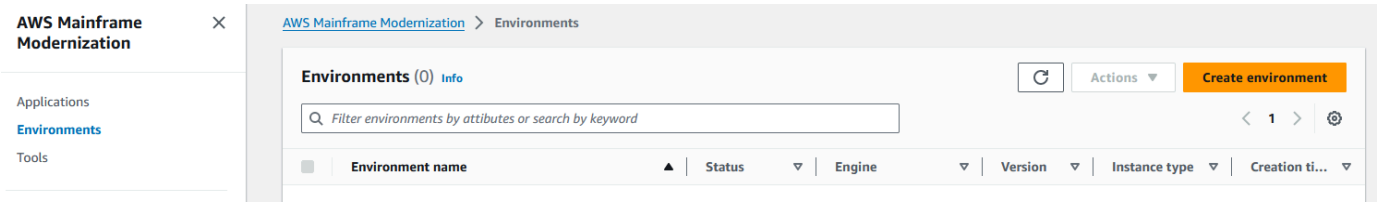


6. Choisissez Enregistrer.

Étape 6 : Création d'un environnement d'exécution

Pour créer un environnement d'exécution

1. Ouvrez la console de modernisation AWS du mainframe.
2. Dans le panneau de navigation, choisissez Environments (Environnements). Choisissez ensuite Create environment.



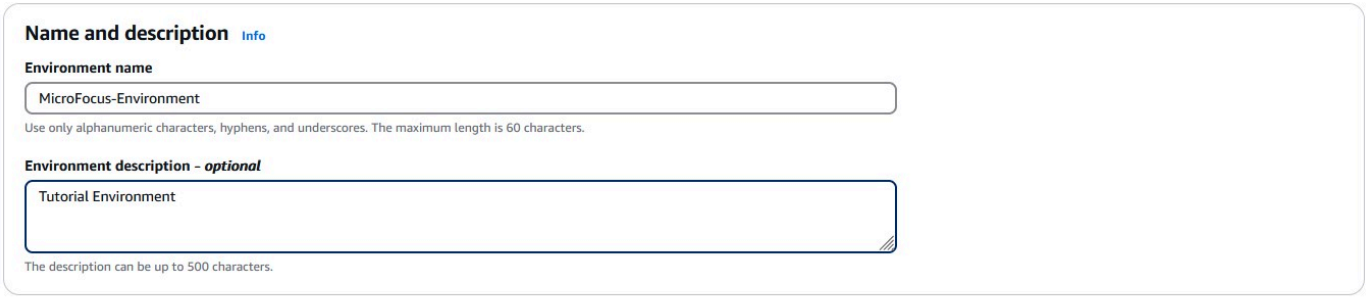
3. Sous Spécifier les informations de base,

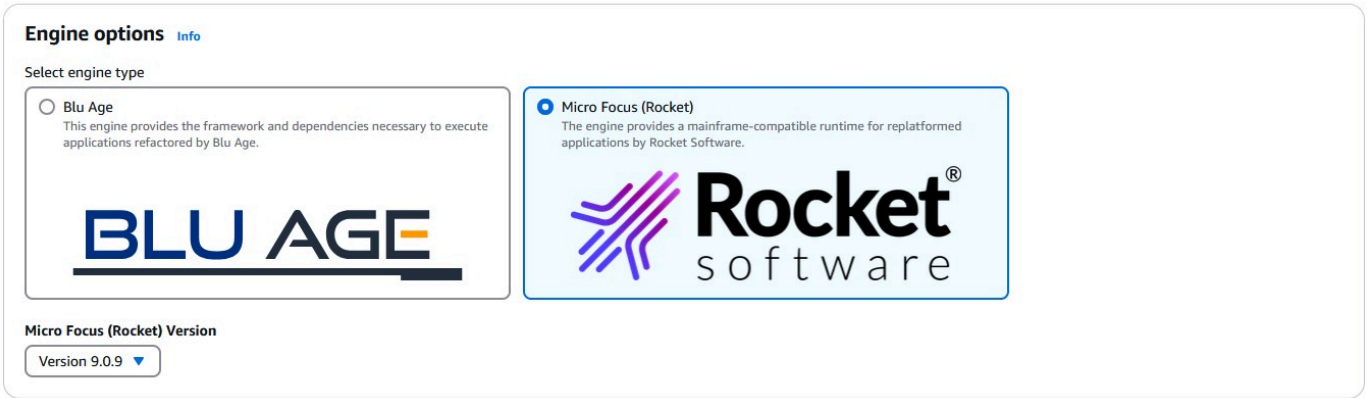
   a. Entrez `MicroFocus-Environment` le nom de l'environnement.

   b. Dans les options du moteur, assurez-vous que Micro Focus (Rocket) est sélectionné.

   c. Choisissez la dernière version de Micro Focus (Rocket).

   d. Choisissez Next (Suivant).





4. Configuration de l'environnement

a. Sous Disponibilité, choisissez High Availability Cluster.

b. Sous Ressources, choisissez M2.c5.large ou M2.m5.large pour le type d'instance et le nombre d'instances que vous souhaitez. Spécifiez jusqu'à deux instances.

c. Sous Sécurité et réseau, choisissez Autoriser les applications déployées dans cet environnement à être accessibles au public et choisissez au moins deux sous-réseaux publics.

d. Choisissez Next (Suivant).

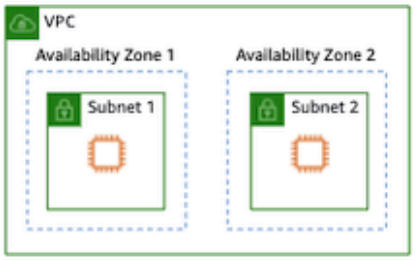
Specify configurations Info

Availability Info

Choose the availability pattern for your environment.

- Standalone runtime environment
  Sets up a single instance in a single availability zone. Does not guarantee high availability but costs less.

- High availability cluster
  Sets up redundant instances across two availability zones. Enables higher availability but costs more.



Resources

Instance type
Choose the instance type for your high availability cluster.

M2.m5.large

Desired capacity
Specify the desired number of instances.

2

Security and network

Allow applications deployed to this environment to be publicly accessible.

Virtual Private Cloud (VPC)
Choose the VPC where you want to create the environment.

Default vpc-15

Subnets
Choose one or more subnets for a high availability setup.

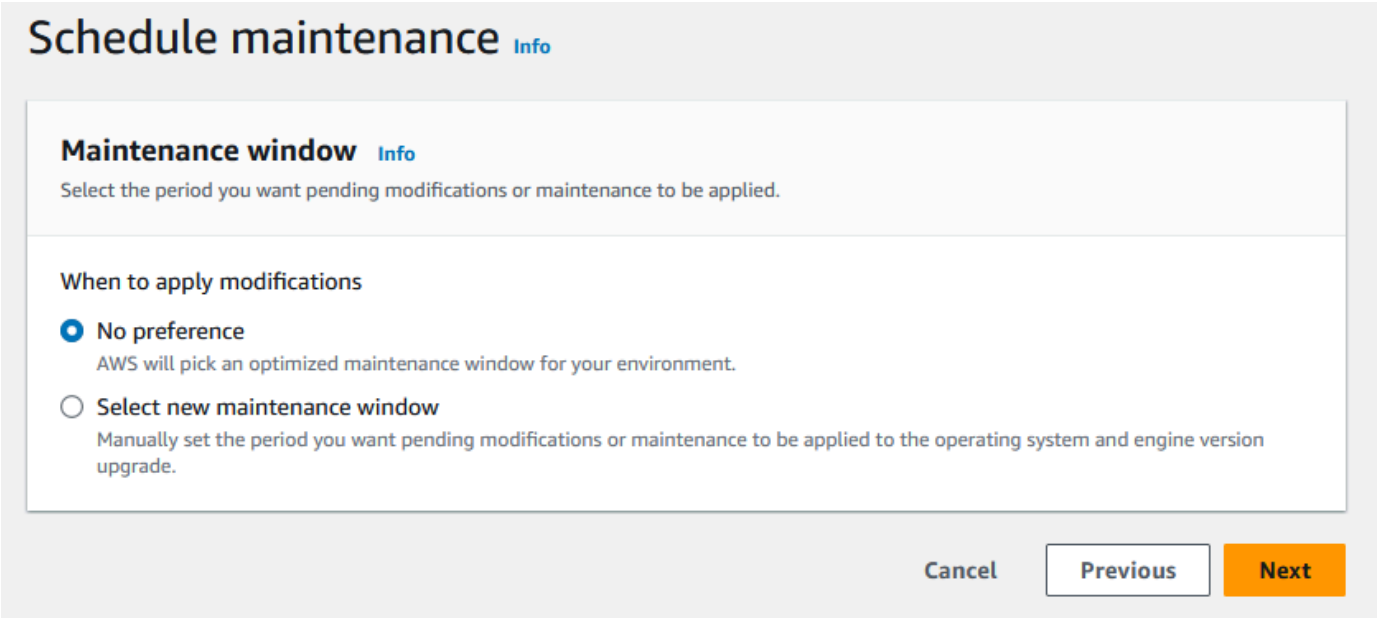
Choose subnets

subnet-56f1a | us-west-2a

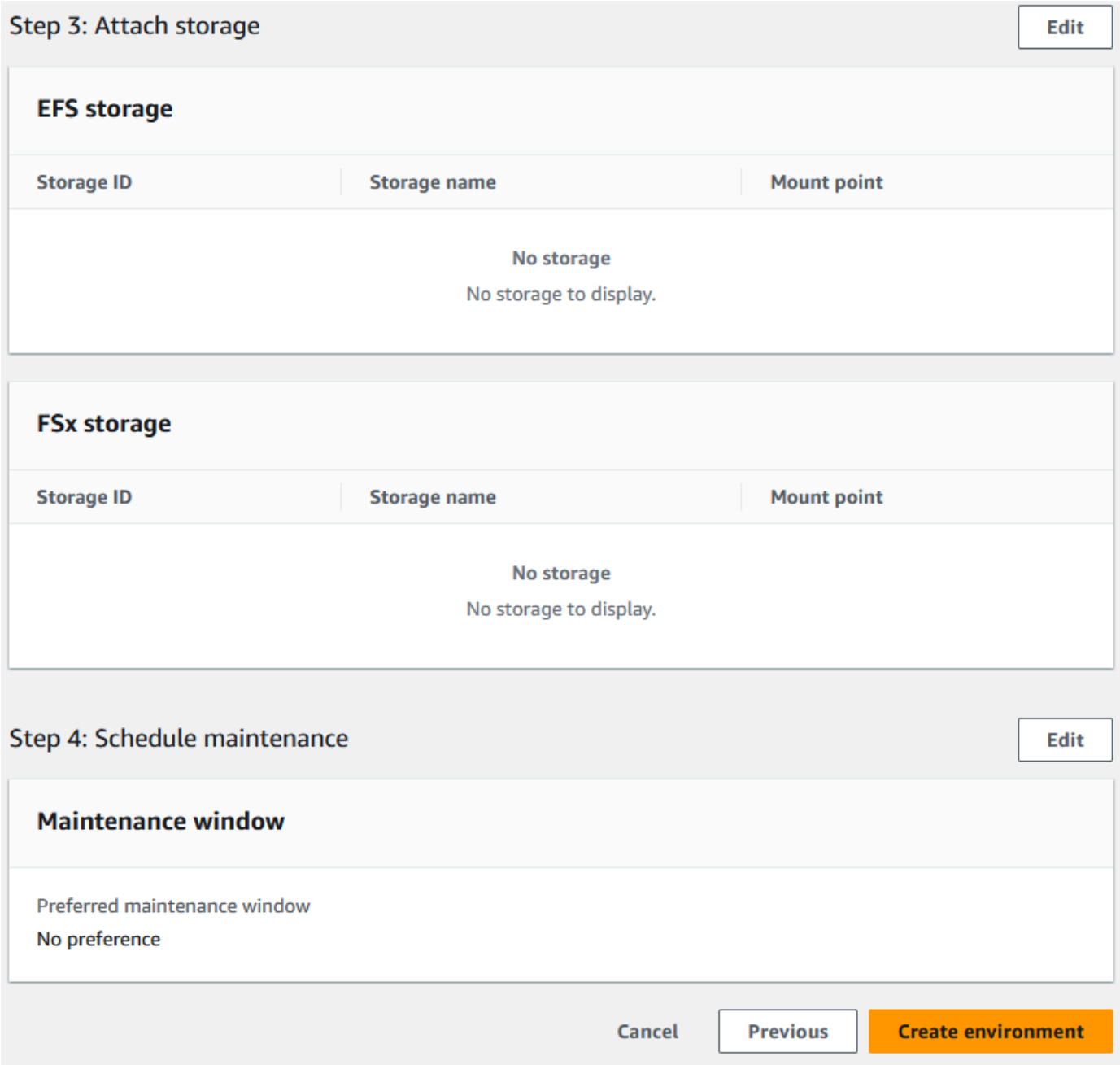
subnet-6685 | us-west-2b

Security groups
Choose one or more security groups for the chosen VPC.

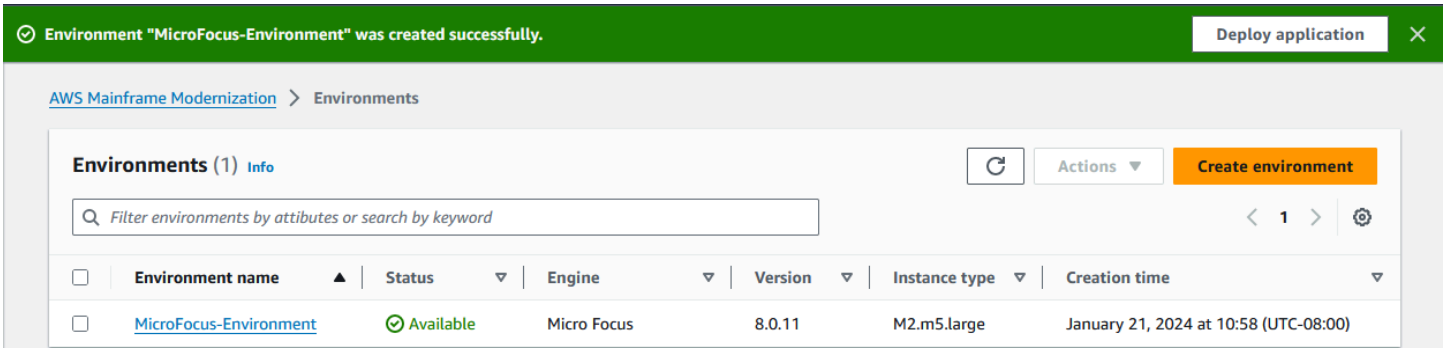
5. Sur la page Joindre un espace de stockage, choisissez Next.
6. Sur la page Planifier la maintenance, choisissez Aucune préférence, puis cliquez sur Suivant.



7. Sur la page Réviser et créer, passez en revue toutes les configurations que vous avez fournies pour l'environnement d'exécution, puis choisissez Créer un environnement.



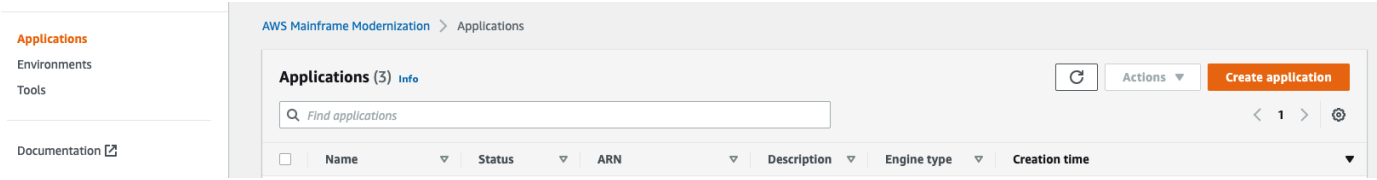
Lorsque vous avez créé votre environnement, une bannière indiquant « `Environment` *name* `was created successfully` Disponible » apparaît et le champ État devient « Disponible ». Le processus de création de l'environnement prend plusieurs minutes, mais vous pouvez passer aux étapes suivantes pendant son exécution.



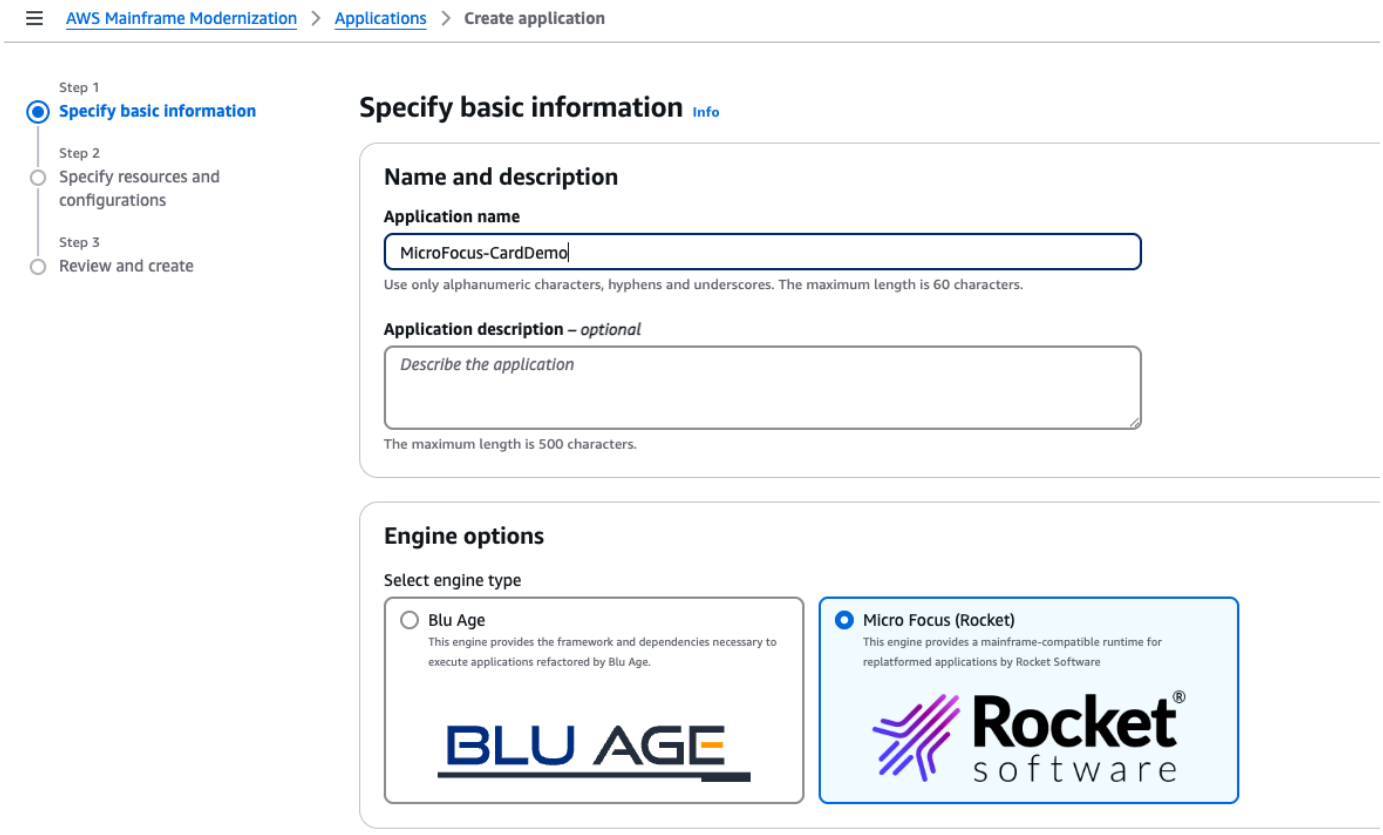
Étape 7 : Création d'une application

Pour créer une application

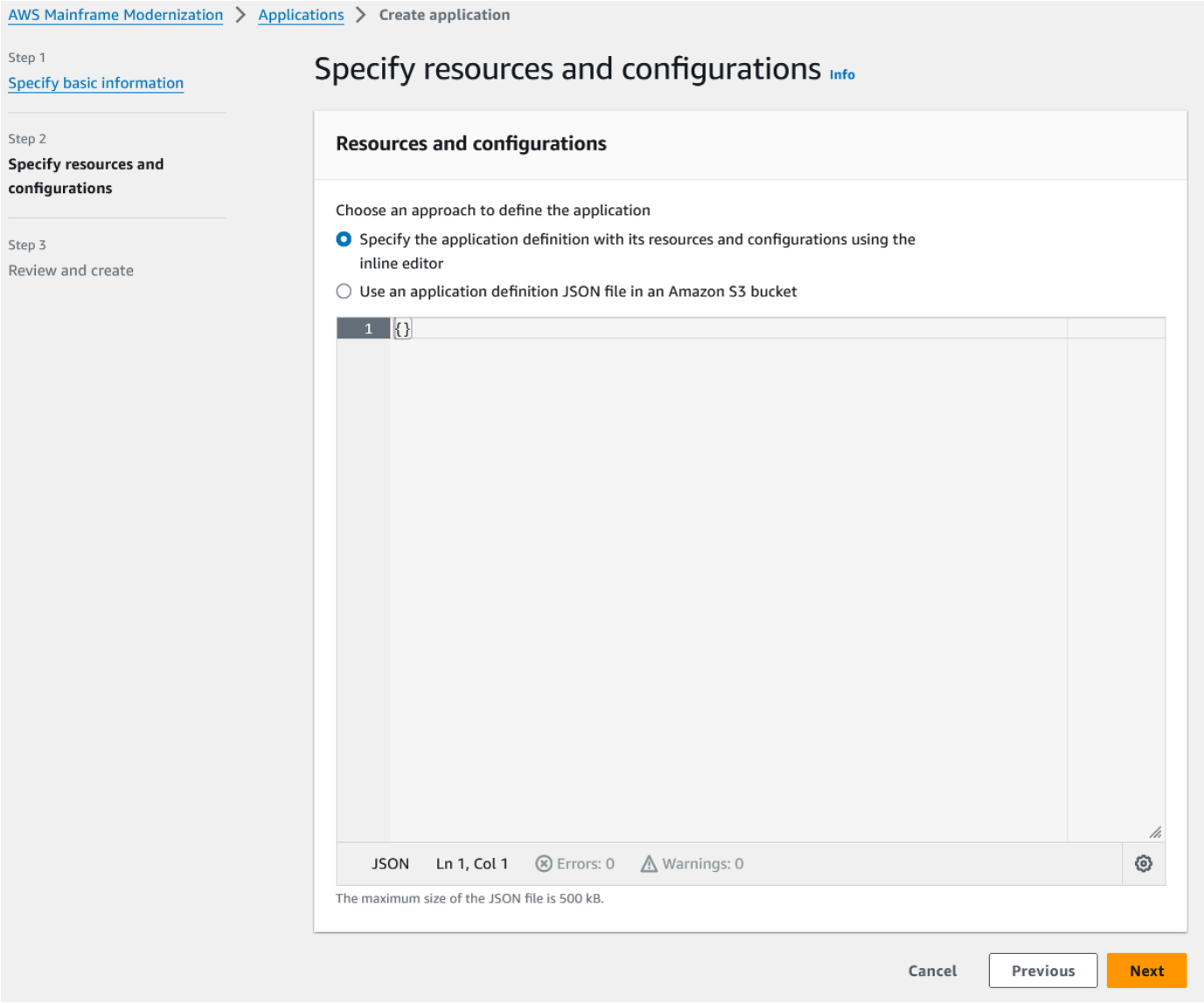
1. Dans le volet de navigation, choisissez Applications. Choisissez ensuite Créer une application.



2. Sur la page Créer une application, sous Spécifier les informations de base, entrez `MicroFocus-CardDemo` le nom de l'application et sous Type de moteur, assurez-vous que Micro Focus (Rocket) est sélectionné. Ensuite, sélectionnez Suivant.



3. Sous Spécifier les ressources et les configurations, choisissez l'option permettant de spécifier la définition de l'application avec ses ressources et configurations à l'aide de l'éditeur en ligne.



Entrez la définition d'application suivante dans l'éditeur :

```
{
  "template-version": "2.0",
  "source-locations": [
    {
      "source-id": "s3-source",
      "source-type": "s3",
      "properties": {
        "s3-bucket": "yourname-aws-region-carddemo",
        "s3-key-prefix": "CardDemo_runtime"
      }
    }
```

```
  ],
  "definition": {
    "listeners": [
      {
        "port": 6000,
        "type": "tn3270"
      }
    ],
    "dataset-location": {
      "db-locations": [
        {
          "name": "Database1",
          "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx"
        }
      ]
    },
    "batch-settings": {
      "initiators": [
        {
          "classes": [
               "A",
               "B"
            ],
          "description": "initiator_AB...."
        },
        {
          "classes": [
                "C",
                "D"
             ],
          "description": "initiator_CD...."
        }
      ],
      "jcl-file-location": "${s3-source}/catalog/jcl"
    },
    "cics-settings": {
      "binary-file-location": "${s3-source}/loadlib",
      "csd-file-location": "${s3-source}/rdef",
      "system-initialization-table": "CARDSIT"
    },
    "xa-resources": [
      {
```

```
        "name": "XASQL",
        "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx",
        "module": "${s3-source}/xa/ESPGSQLXA64.so"
      }
    ]
  }
}
```

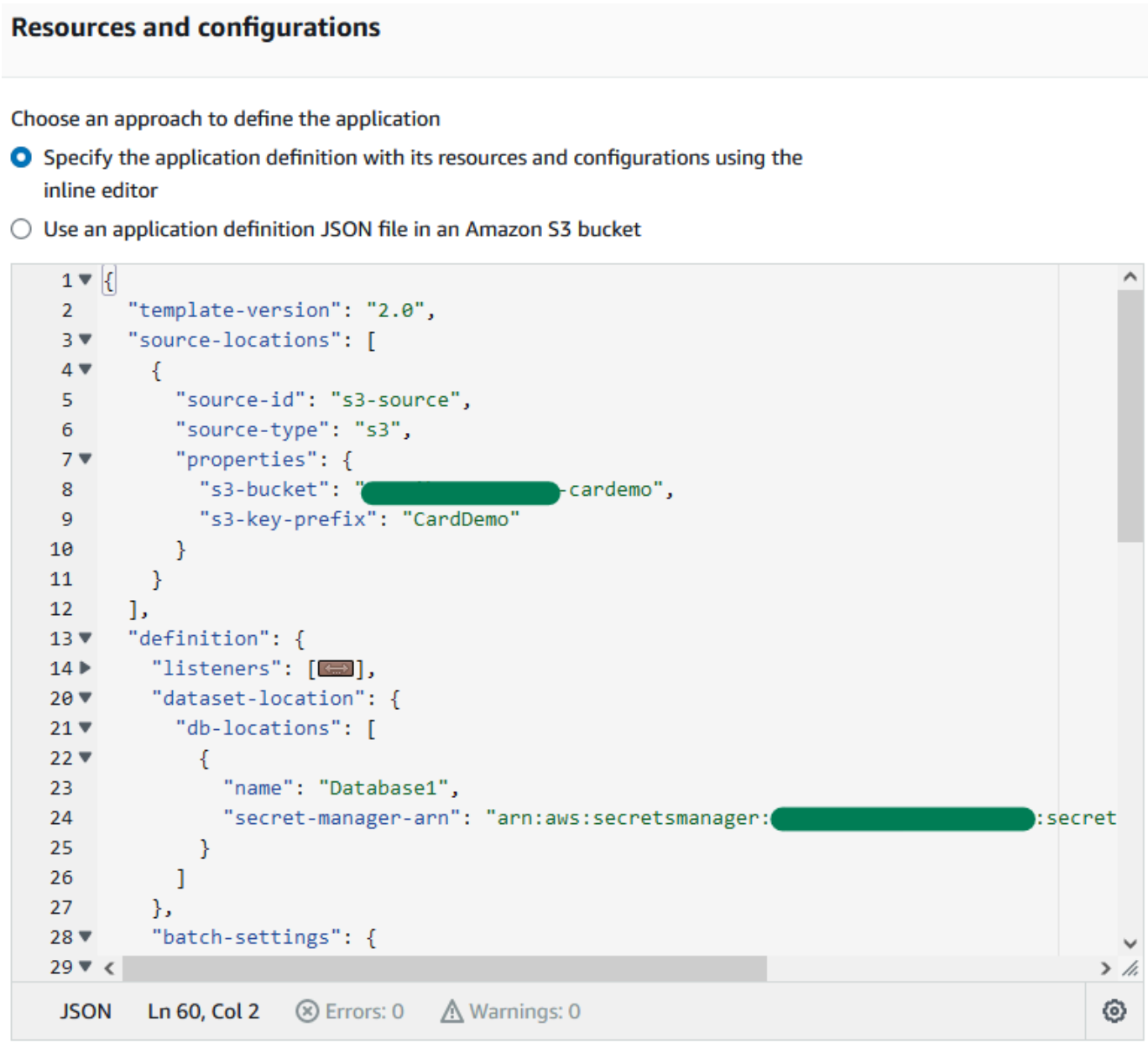
> **Note**
>
> Ce fichier est sujet à modification.

4. Modifiez le JSON de l'application dans l'objet de propriétés de source-locations comme suit :

   a. Remplacez la valeur pour `s3_bucket` par le nom du compartiment Amazon S3 que vous avez créé à l'étape 1.

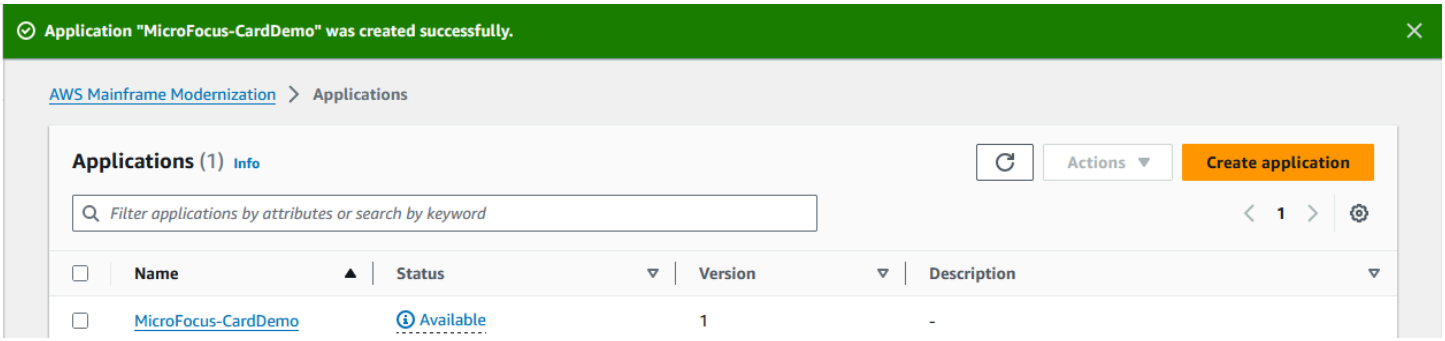
   b. Remplacez la valeur pour `s3-key-prefix` par le dossier (préfixe clé) dans lequel vous avez chargé les fichiers CardDemo d'exemple. Si vous avez chargé le `CardDemo` répertoire directement dans un compartiment Amazon S3, il `s3-key-prefix` n'est pas nécessaire de le modifier.

   c. Remplacez les deux `secret-manager-arn` valeurs par l'ARN du secret de base de données que vous avez créé à l'étape 4.



Pour plus d'informations sur la définition de l'application, consultezDéfinition de l'application Rocket Software (anciennement Micro Focus).

5. Choisissez Next (Suivant) pour continuer.
6. Sur la page Réviser et créer, passez en revue les informations que vous avez fournies, puis choisissez Créer une application.

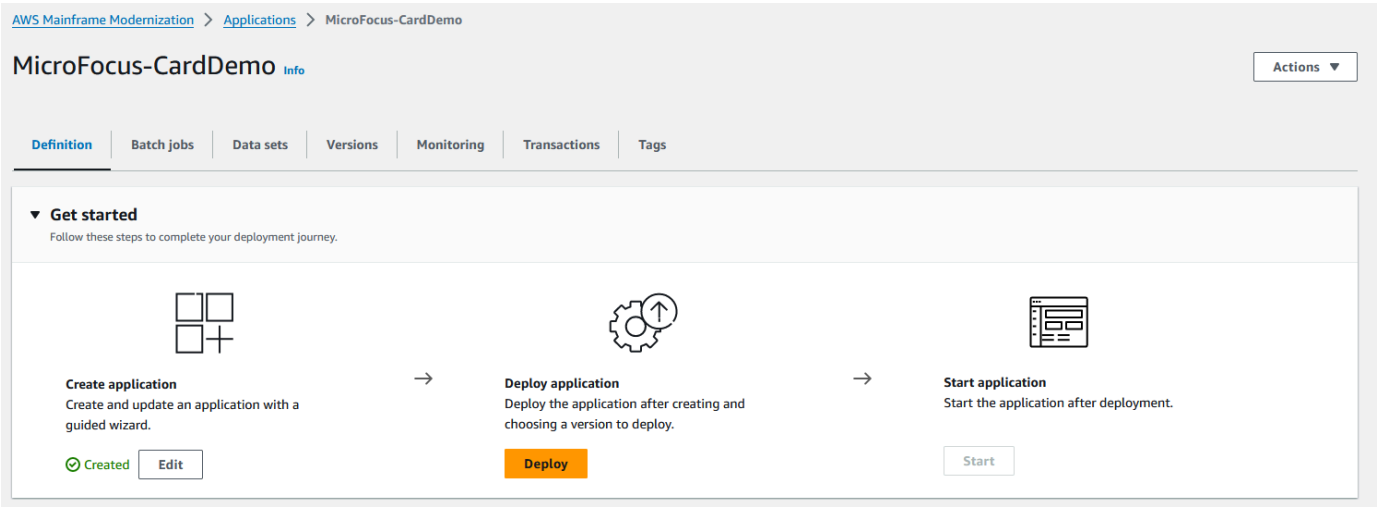


Lorsque vous avez créé votre application, une bannière apparaît indiquant`Application` *`name`* `was created successfully`. Et le champ État devient Disponible.

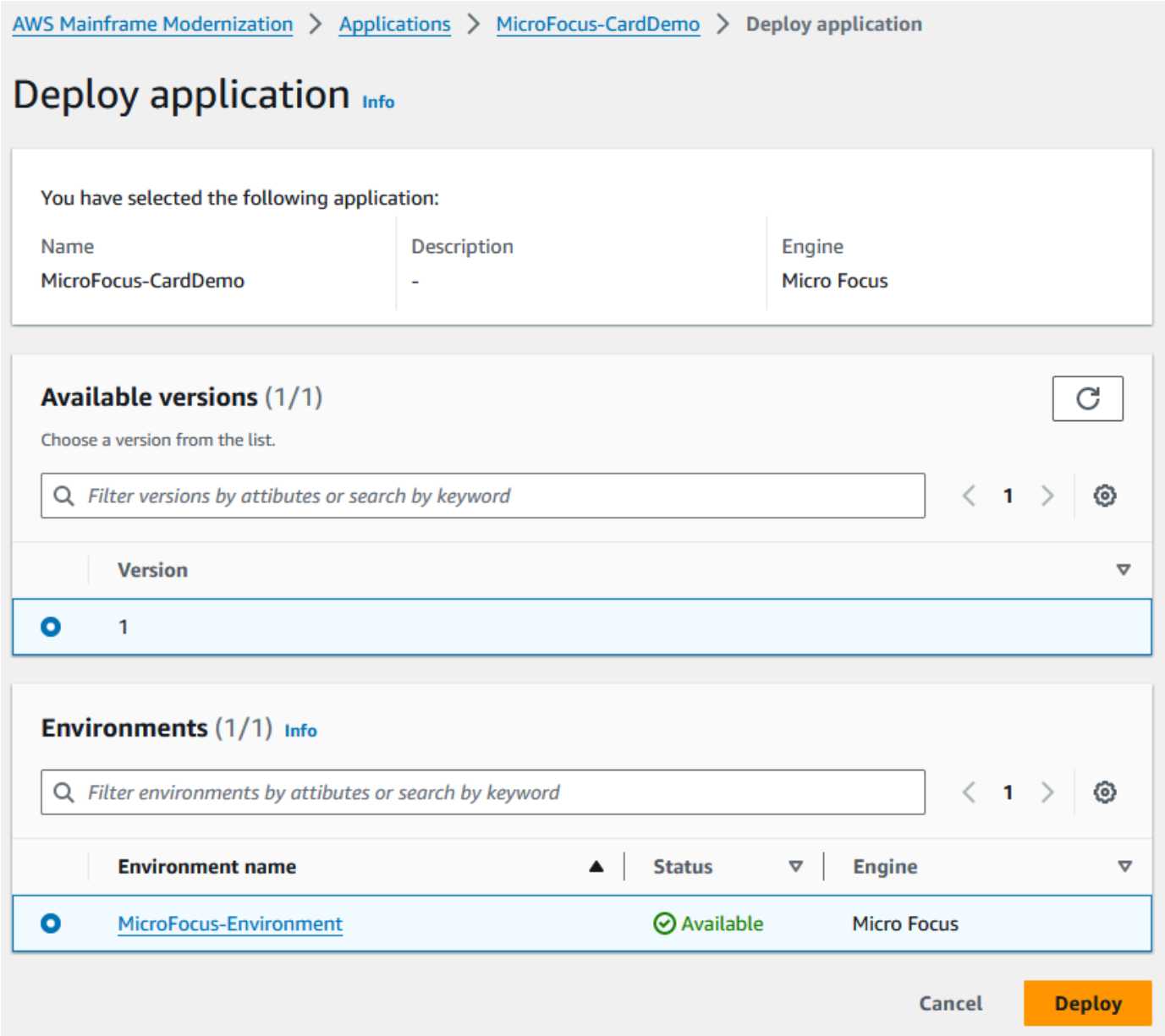
Étape 8 : Déploiement d'une application

Pour déployer une application

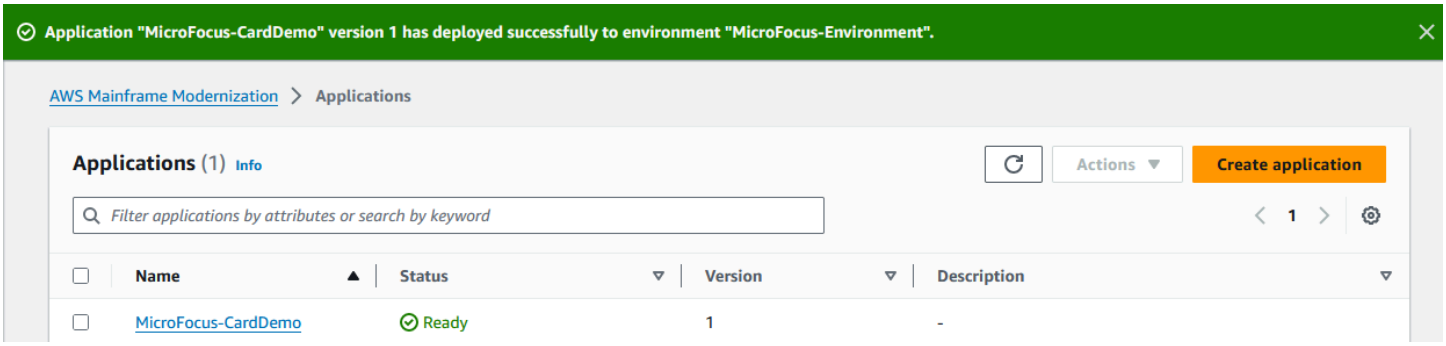
1. Dans le volet de navigation, choisissez Applications, puis choisissez`MicroFocus-CardDemo`.
2. Sous Déployer l'application, choisissez Déployer.



3. Choisissez la dernière version de l'application et l'environnement que vous avez créés précédemment, puis choisissez Déployer.



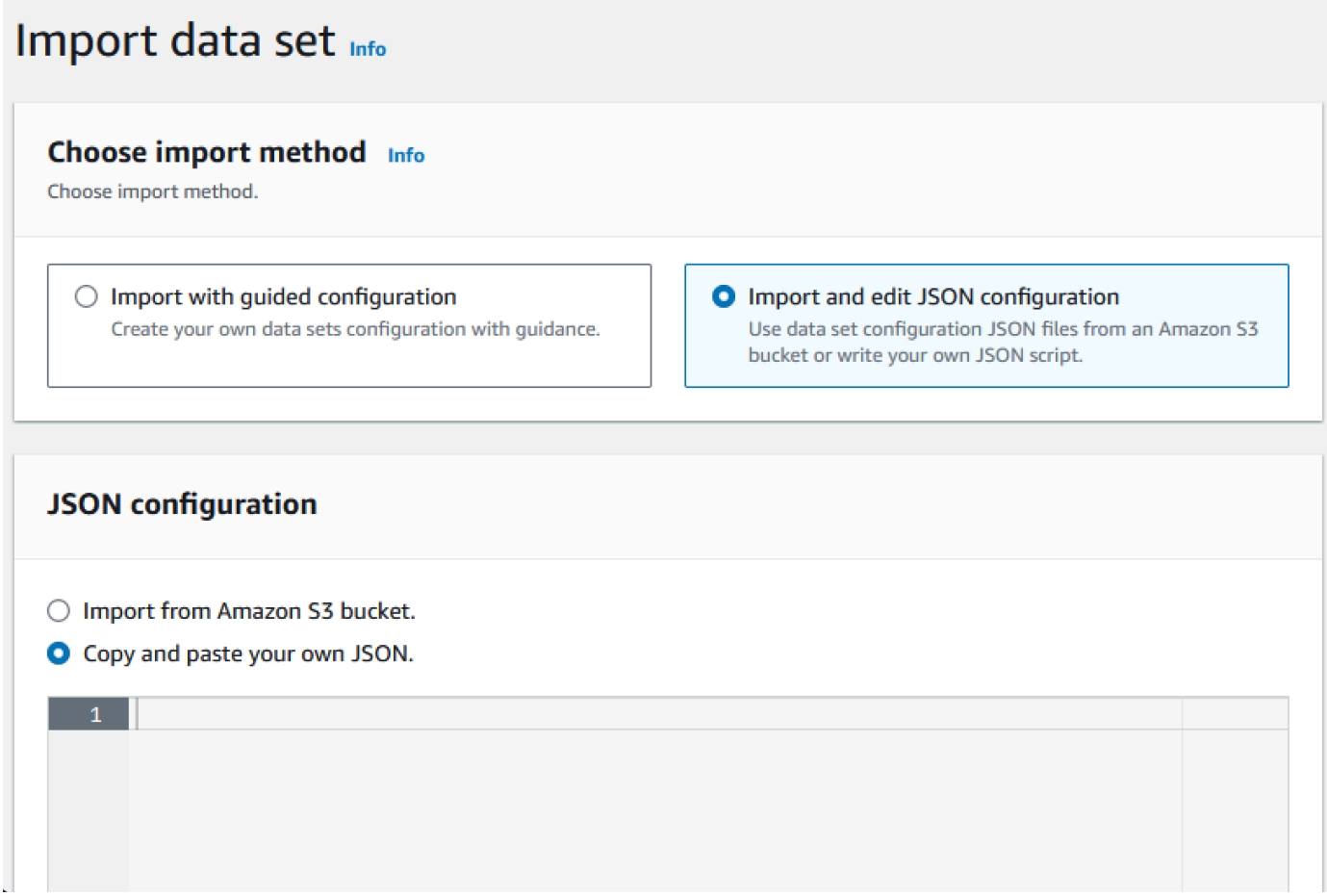
Lorsque l' CardDemo application est déployée avec succès, le statut passe à Prêt.



Étape 9 : Importer des ensembles de données

Pour importer des ensembles de données

1. Dans le volet de navigation, choisissez Applications, puis choisissez l'application.
2. Choisissez l'onglet Ensembles de données. Choisissez ensuite Import (Importer).
3. Choisissez Importer et modifier la configuration JSON, puis choisissez l'option Copier et collez votre propre JSON.



4. Copiez et collez le code JSON suivant, mais ne choisissez pas encore « Soumettre ». Ce JSON contient tous les ensembles de données requis pour l'application de démonstration, mais nécessite les détails de votre compartiment Amazon S3.

```
{
    "dataSets": [
        {
            "dataSet": {
                "storageType": "Database",
```

```
                "datasetName": "AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 300,
                    "max": 300
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 16
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
```

```
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
```

```
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 9,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 500,
                    "max": 500
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
```

```
                        "primaryKey": {
                            "length": 11,
                            "offset": 25
                        }
                    }
                },
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 350,
                    "max": 350
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
```

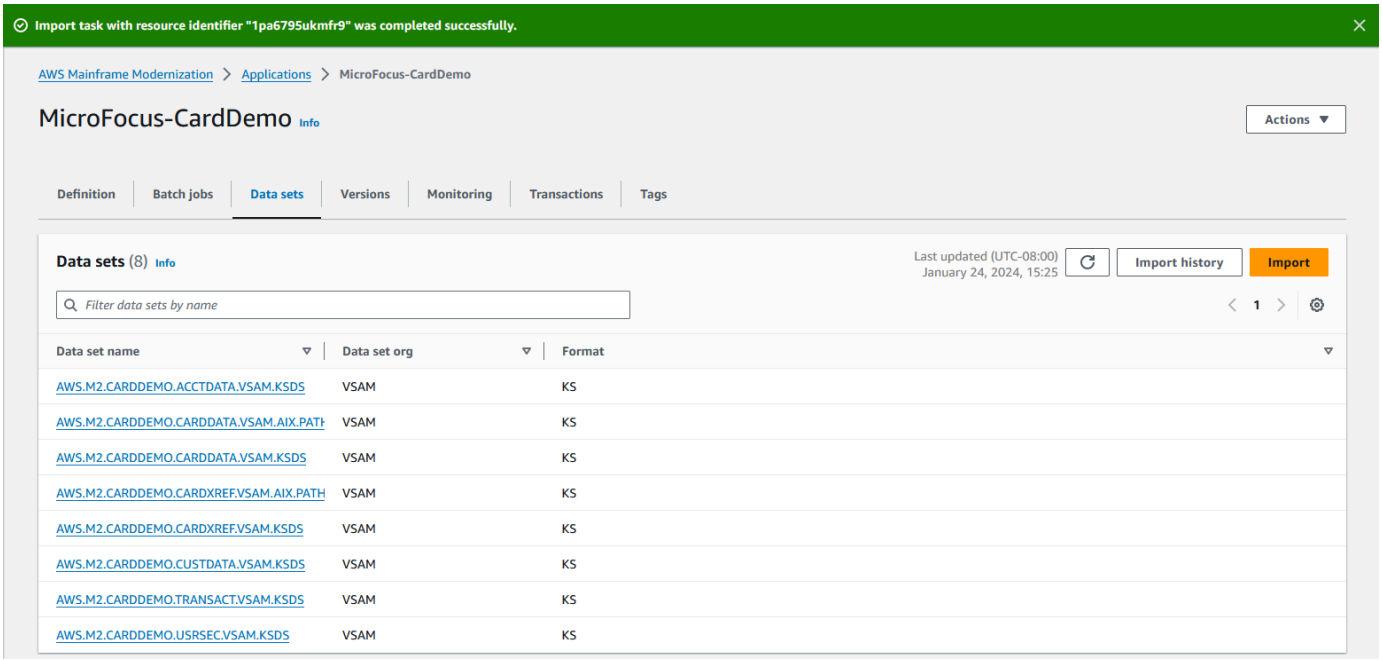
```
                "datasetName": "AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 8,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 80,
                    "max": 80
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS.DAT"
            }
        }
    ]
}
```

5. Remplacez chaque occurrence de `<s3-bucket-name>` (il y en a huit) par le nom du compartiment Amazon S3 qui contient le CardDemo dossier, par exemple,`your-name-aws-region-carddemo`.

   > **Note**
   >
   > Pour copier l'URI Amazon S3 du dossier dans Amazon S3, sélectionnez le dossier, puis choisissez Copier l'URI Amazon S3.

6. Sélectionnez Envoyer.

   Lorsque l'importation est terminée, une bannière apparaît avec le message suivant : `Import task with resource identifier` *`name`* `was completed successfully.` La liste des ensembles de données importés s'affiche.

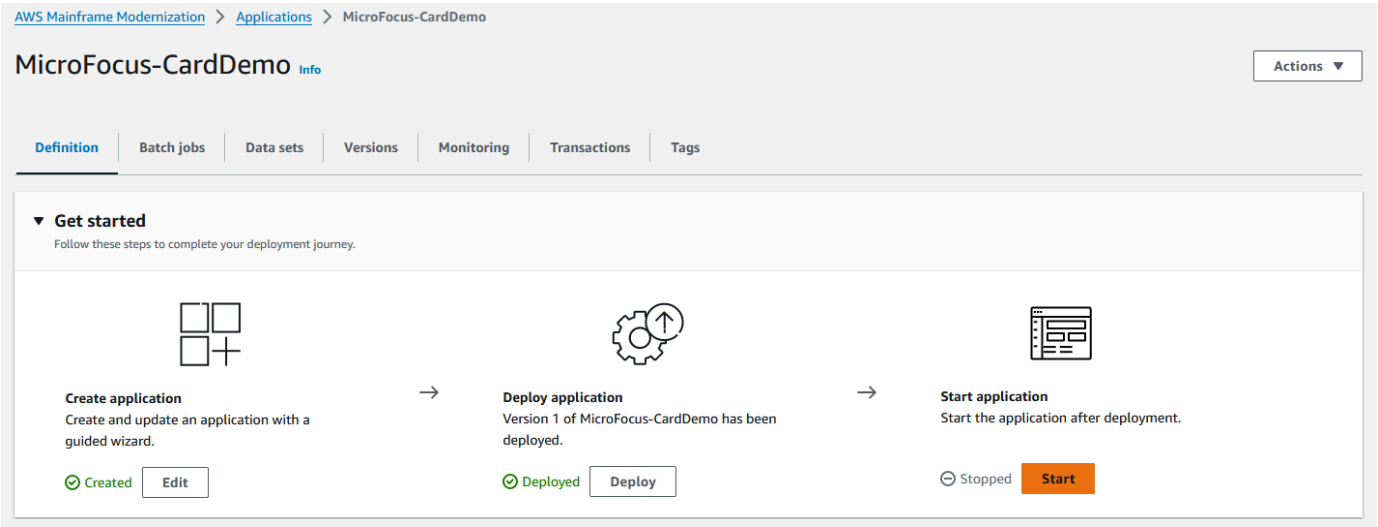


Vous pouvez également consulter l'état de toutes les importations de jeux de données en choisissant Historique des importations dans l'onglet Ensembles de données.

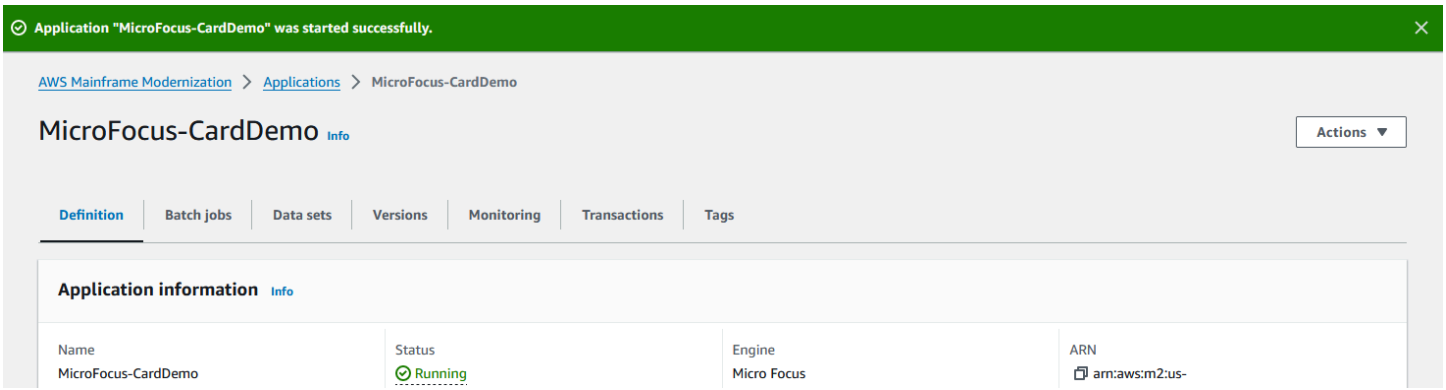
Étape 10 : démarrer une application

Pour démarrer une application

1. Dans le volet de navigation, choisissez Applications, puis choisissez l'application.
2. Choisissez Démarrer l'application.



Lorsque l' CardDemo application démarre avec succès, une bannière apparaît avec le message suivant : `Application` *name* `was started successfully` Le champ Status devient Running.



Étape 11 : Connectez-vous à l'application CardDemo CICS

Avant de vous connecter, assurez-vous que le VPC et le groupe de sécurité que vous avez spécifiés pour l'application sont les mêmes que ceux que vous avez demandés pour l'interface réseau à partir de laquelle vous allez vous connecter.

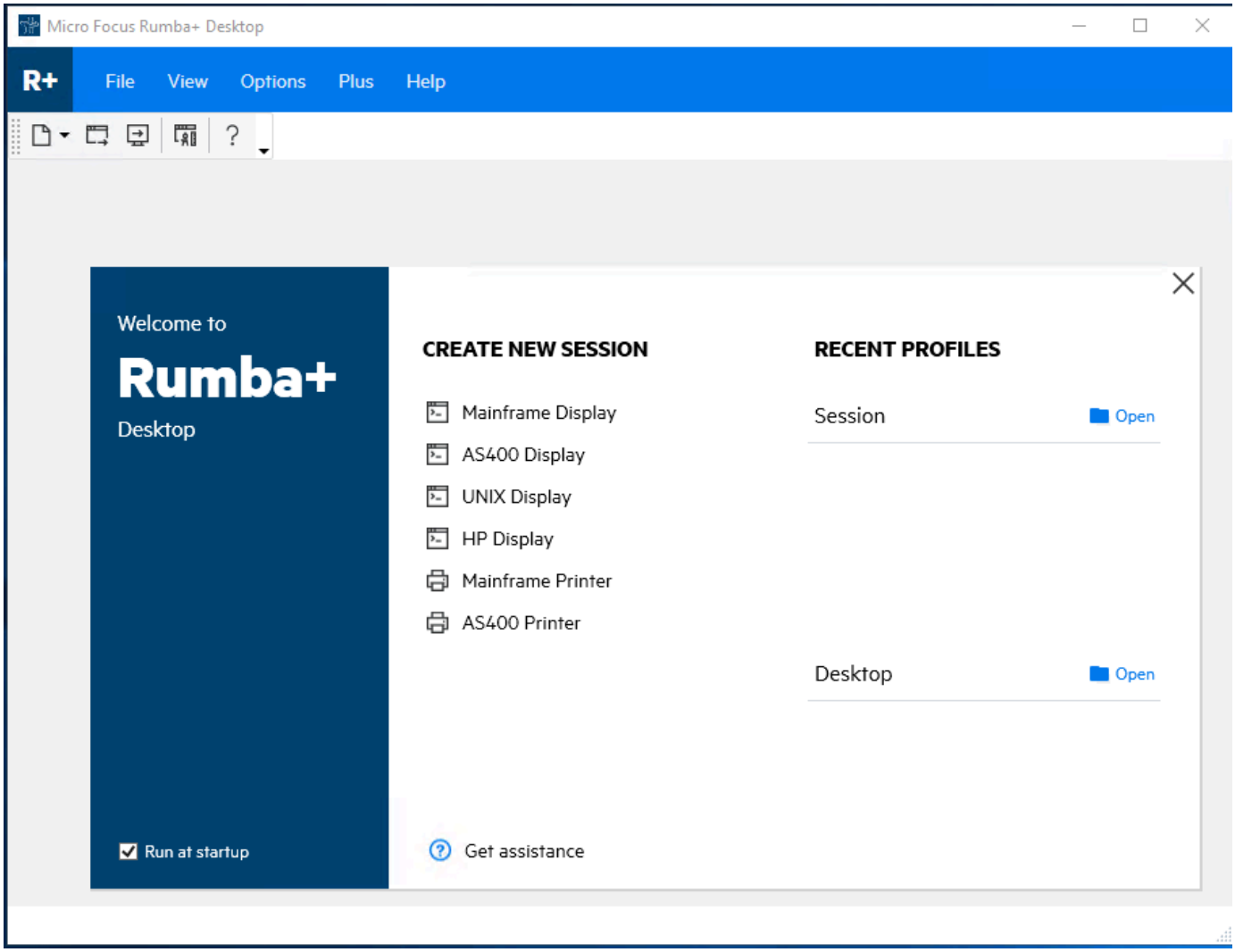
Pour configurer la connexion TN327 0, vous avez également besoin du nom d'hôte DNS et du port de l'application.

Pour configurer et connecter une application au mainframe à l'aide de l'émulateur de terminal

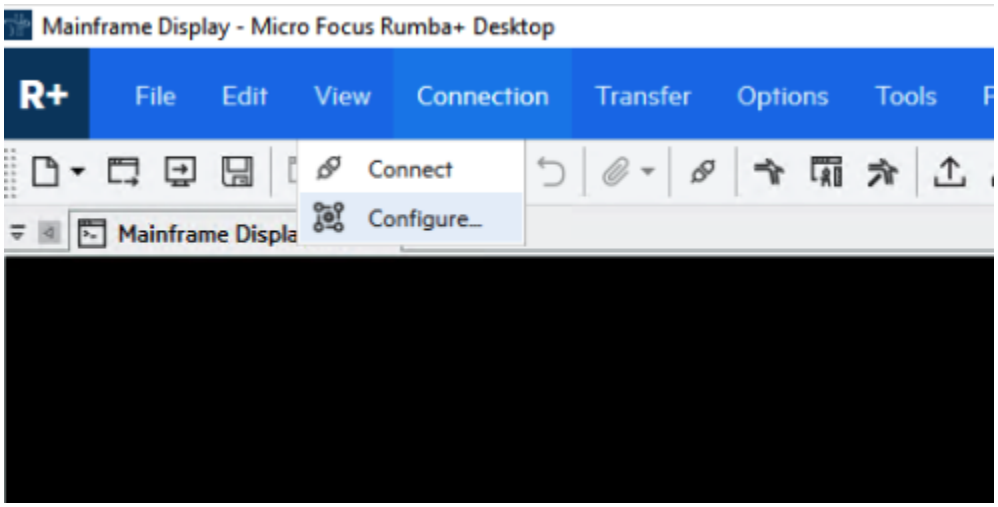
1. Ouvrez la console AWS Mainframe Modernization et choisissez Applications, puis choisissez`MicroFocus-CardDemo`.
2. Cliquez sur l'icône de copie pour copier le nom d'hôte DNS. Assurez-vous également de noter le numéro de port.
3. Démarrez un émulateur de terminal. Ce didacticiel utilise Micro Focus Rumba+.

   > **Note**
   >
   > Les étapes de configuration varient en fonction de l'émulateur.

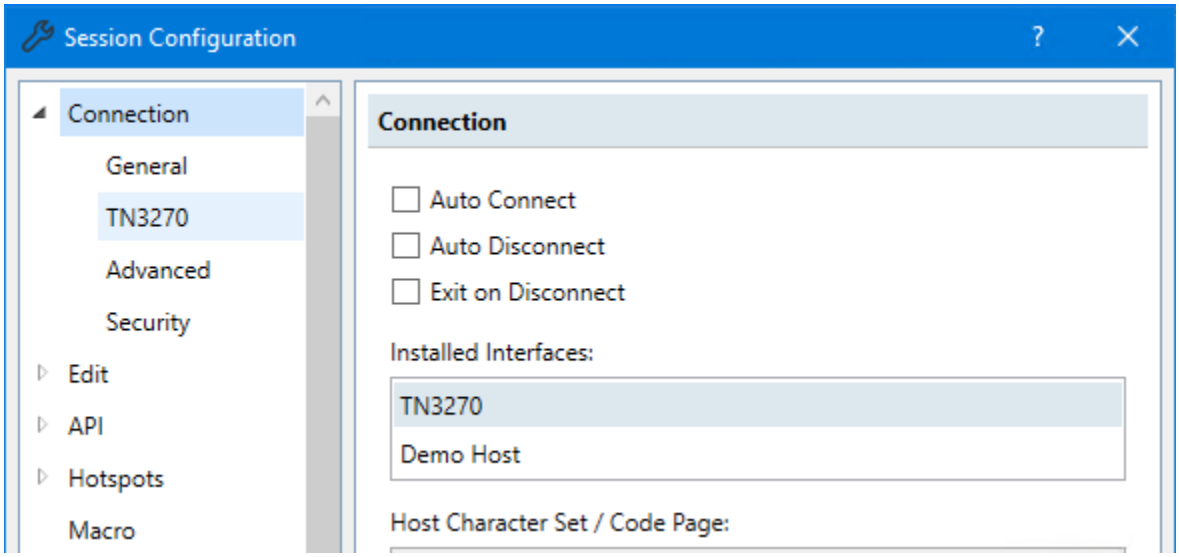
4. Choisissez Mainframe Display.



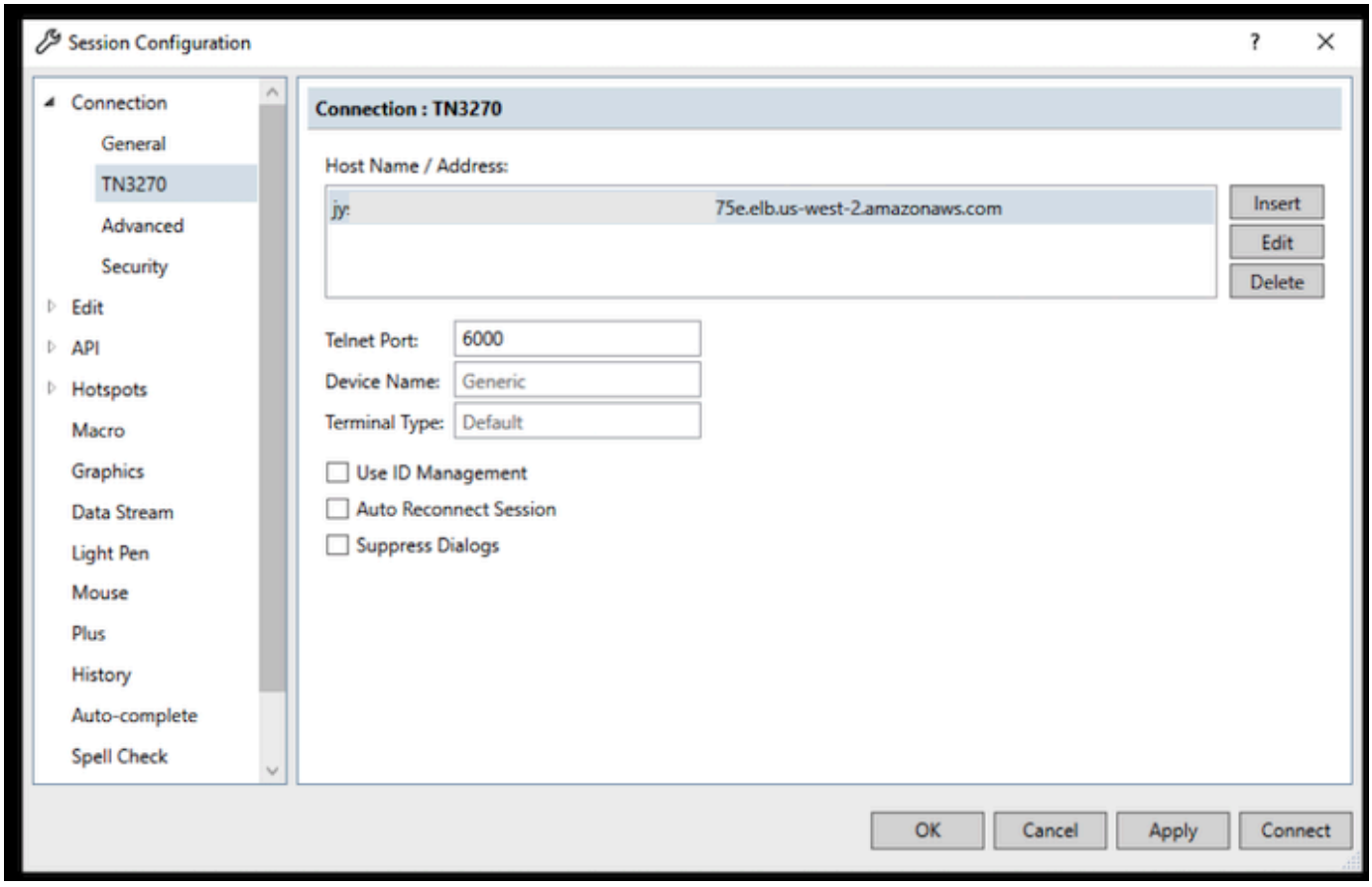
5. Choisissez Connexion, puis Configurer.



6. Sous Interfaces installées, choisissez`TN3270`, puis choisissez `TN3270` à nouveau dans le menu Connexion.



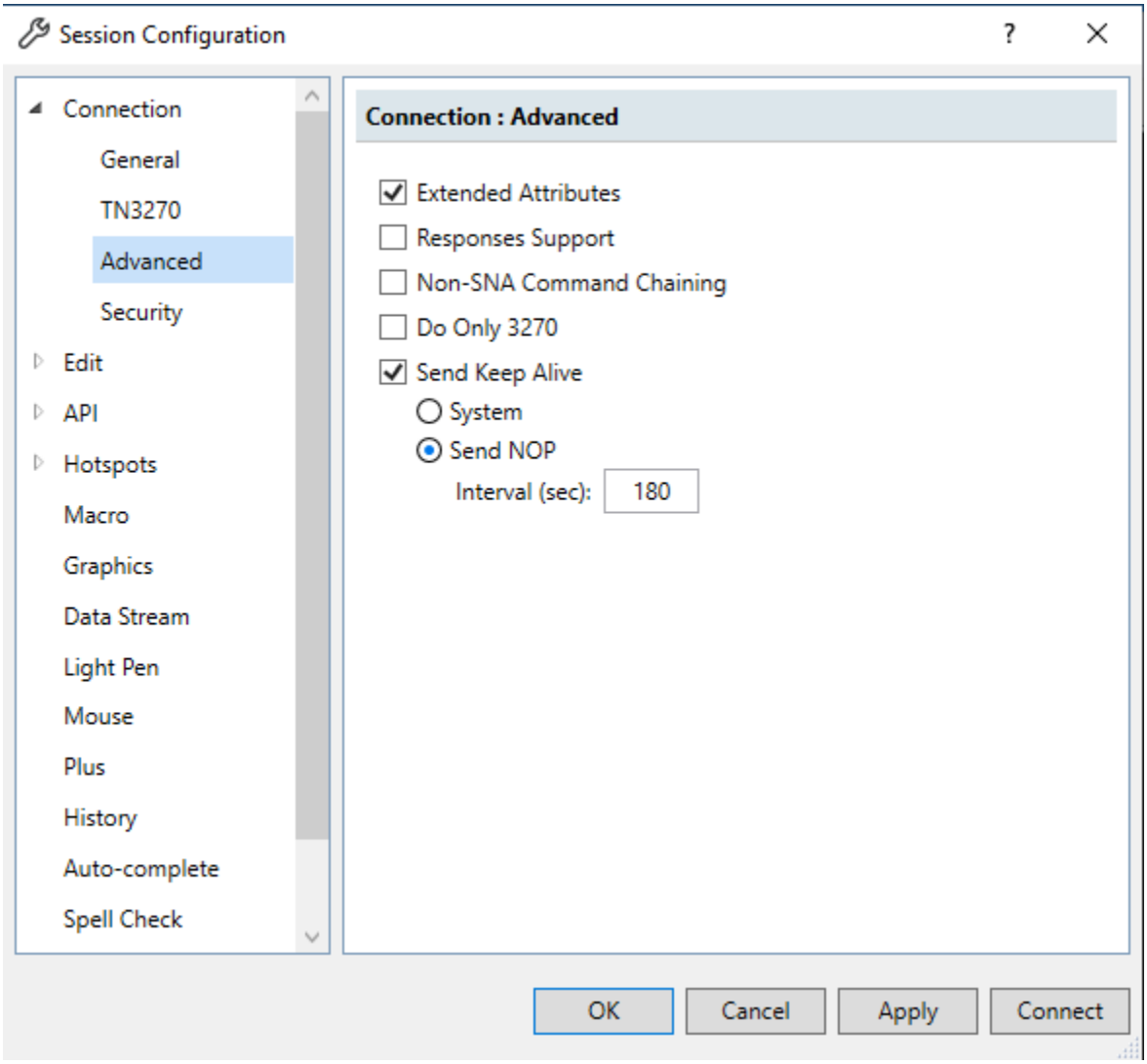
7. Choisissez Insérer, puis collez le `DNS Hostname` pour l'application. Spécifiez `6000` le port Telnet.



> **Note**
>
> Si vous utilisez AWS AppStream 2.0 dans un navigateur et que vous rencontrez des difficultés pour coller des valeurs, reportez-vous à la section Résolution des problèmes liés aux utilisateurs de la AppStream version 2.0.

8. Sous Connexion, choisissez Avancé, puis choisissez Send Keep Alive et Send NOP, puis entrez 180 pour l'intervalle.

> **Note**
>
> La configuration du paramètre Keep Alive sur votre terminal TN327 0 à au moins 180 secondes permet de garantir que le Network Load Balancer n'interrompe pas votre connexion.



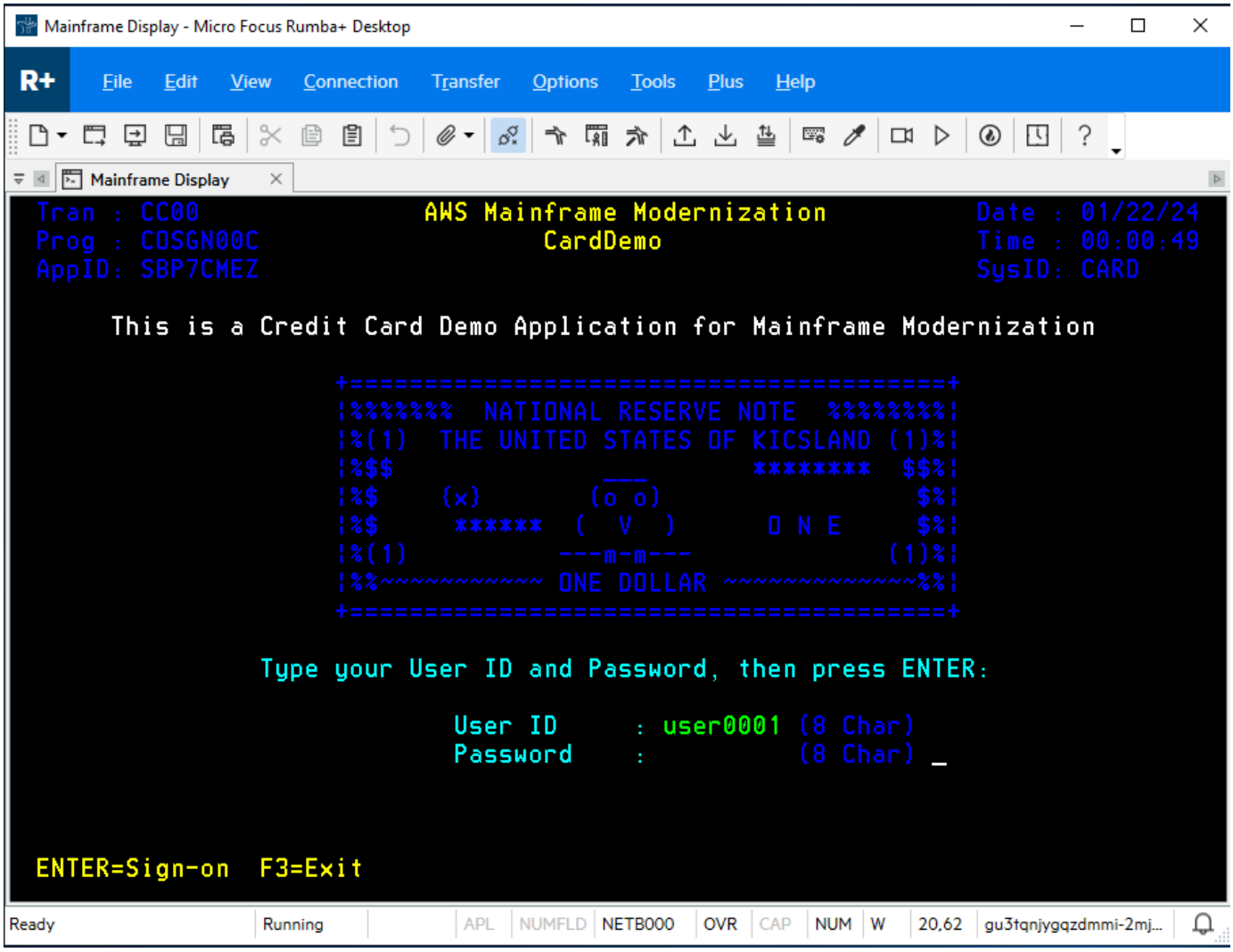
9. Choisissez Se connecter.

> **Note**
>
> En cas d'échec de la connexion :
>
> - Si vous utilisez la AppStream version 2.0, vérifiez que le VPC et le groupe de sécurité spécifiés pour l'environnement de l'application sont identiques à ceux du parc 2.0. AppStream
> - Utilisez le VPC Reachability Analyzer pour analyser la connexion. Vous pouvez accéder à l'Analyzer de Reachability via la console.

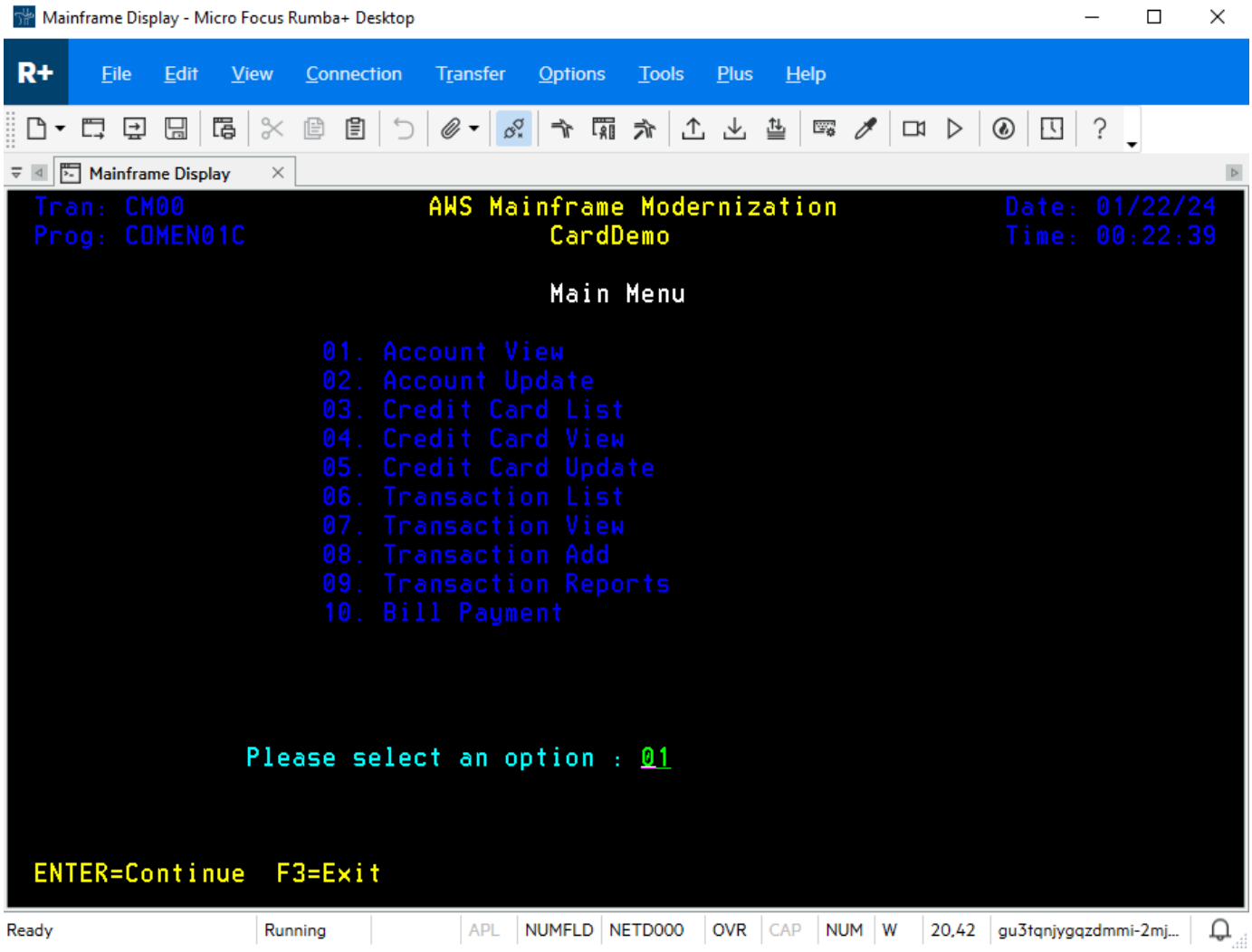
- À titre d'étape de diagnostic, essayez d'ajouter ou de modifier les règles entrantes du groupe de sécurité pour l'application afin d'autoriser le trafic vers le port 6000 depuis n'importe où (c'est-à-dire le bloc CIDR 0.0.0.0/0). Si vous vous connectez avec succès, vous savez que le groupe de sécurité bloquait votre trafic. Remplacez la source du groupe de sécurité par une source plus spécifique. Pour plus d'informations sur les groupes de sécurité, voir Notions de base sur les groupes de sécurité.

10. Entrez `USER0001` le nom d'utilisateur et `password` le mot de passe.

> **Note**
>
> Dans Rumba, la valeur par défaut pour Clear est ctrl-shift-z, et la valeur par défaut pour Reset est ctrl-r.

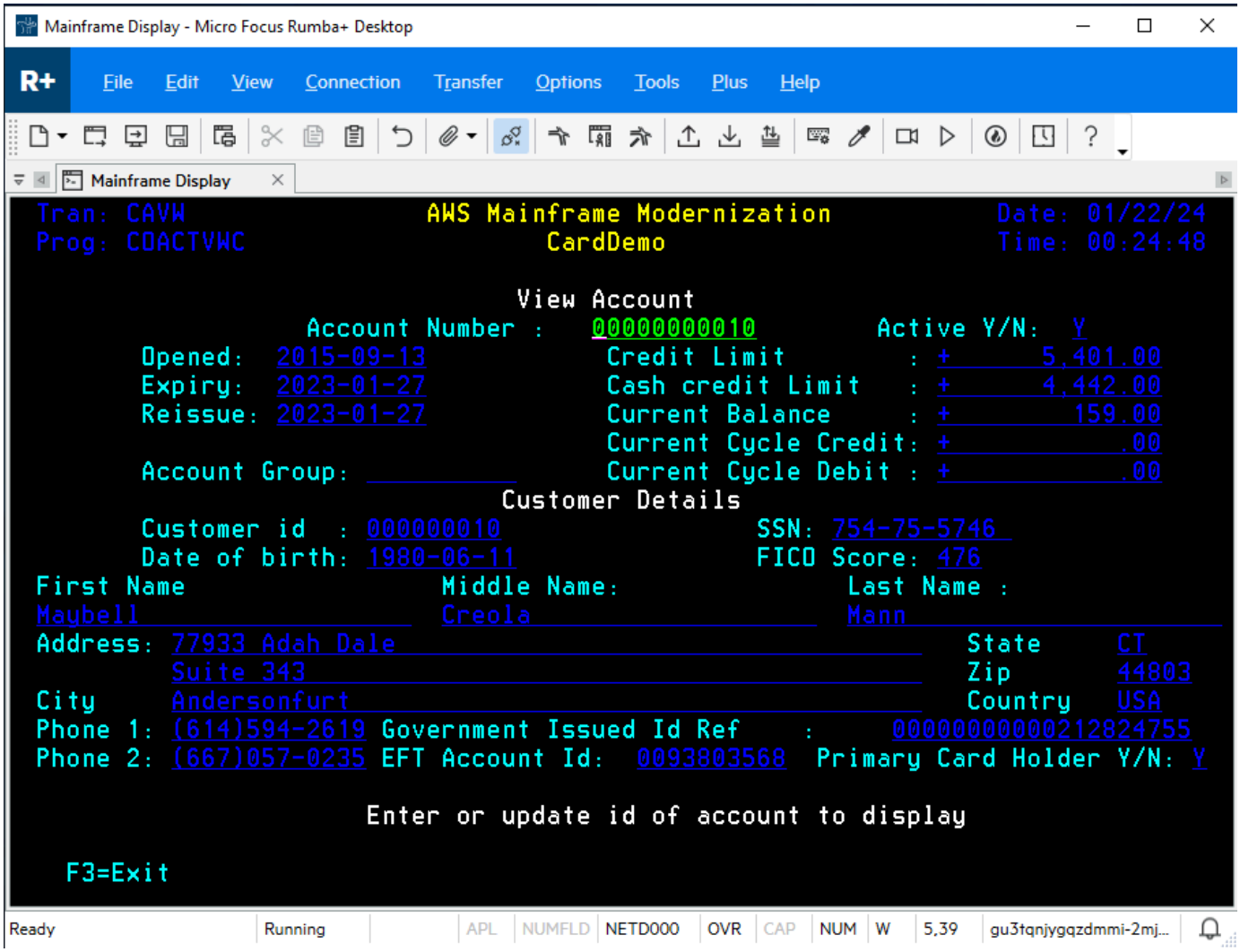


11. Une fois connecté, vous pouvez naviguer dans l' CardDemoapplication.
12. Entrez `01` pour la vue du compte.



13. Entrez le numéro `00000000010` de compte et appuyez sur la touche Entrée de votre clavier.

> **Note**
>
> Les autres comptes valides sont `0000000011` et`00000000020`.



14. Appuyez sur Quitter F3 pour accéder au menu et F3 pour terminer la transaction.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Si nécessaire, arrêtez l'application.
- Supprimez l'application . Pour de plus amples informations, veuillez consulter Supprimer une AWS Mainframe Modernization application.
- Supprimez l'environnement d'exécution. Pour de plus amples informations, veuillez consulter Supprimer un environnement d'exécution de modernisation du AWS mainframe.

- Supprimez les compartiments Amazon S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez la section Suppression d'un compartiment dans le guide de l'utilisateur Amazon S3.
- Supprimez le AWS Secrets Manager secret que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un secret.
- Supprimez la clé KMS que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Supprimer des clés AWS KMS.
- Supprimez la base de données Amazon RDS que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Supprimer l' EC2 instance et l'instance de base de données dans le guide de l'utilisateur Amazon RDS.
- Si vous avez ajouté une règle de groupe de sécurité pour le port 6000, supprimez-la.

Étapes suivantes

Pour savoir comment configurer un environnement de développement pour vos applications modernisées, consultez Tutoriel : Configuration AppStream 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer.

AWS Cycle de vie des composants de modernisation du mainframe

Chaque composant de la modernisation du AWS mainframe passe par des mises à niveau de version et un cycle de développement. Vous pouvez utiliser cette page comme présentation pour comprendre ces composants, leurs plans de mise à niveau de version et la manière dont AWS Mainframe Modernization communique la publication ou la dépréciation de ces composants ou de leurs versions.

Aperçu du cycle de vie des composants

AWS Le cycle de vie de modernisation des mainframes décrit l'approche et les délais de publication et de prise en charge des composants du service de modernisation des AWS mainframes tout au long de leur cycle de vie. Fournir un cycle de vie prévisible et cohérent vous aide à planifier, tester et déployer les nouvelles versions.

Tous les composants de modernisation du AWS mainframe AWS fournis bénéficient du support produit fourni Support depuis leur sortie jusqu'à leur retrait, conformément au calendrier des versions de chaque composant. Vous pouvez en savoir plus sur le Support champ d'application et les activités sur Compare Support Plans. Au cours des projets de modernisation en cours, nous encourageons généralement les équipes de prestation de services professionnels à fournir d'abord le support client conformément au cahier des charges.

AWS La modernisation du mainframe publie certains composants avec des versions provenant de fournisseurs qui peuvent être AWS lui-même, certains AWS partenaires ou des communautés. Pour chaque composant de modernisation AWS du mainframe, une version possède un numéro de version principal et un numéro de version secondaire. Chaque composant possède sa propre numérotation des versions majeures et mineures.

Pour les composants versionnés, nous avons les objectifs suivants :

- Pour publier des versions plus récentes des composants de modernisation du AWS mainframe sur une base régulière ou à la demande du client. Si la version la plus récente d'un composant est souhaitée mais qu'elle n'est pas encore disponible dans le service de modernisation du AWS mainframe, vous pouvez faire une demande explicite via la demande de fonctionnalité Support du produit (PFR).

- Faire en sorte que les dates de fin de support et de retrait des versions spécifiques aux composants de la modernisation du AWS mainframe soient alignées sur les dates de fin de support du fournisseur de composants.
- Informer les clients environ un an avant le retrait de la version majeure d'un composant.

Bien que nous nous efforcions de respecter ces directives, dans certains cas, nous pouvons retirer certaines versions plus tôt avec des délais de notification plus courts. Par exemple, nous pouvons retirer rapidement une version présentant des problèmes de sécurité avec un délai de notification plus court. Nous pouvons également retirer les versions mineures de manière anticipée lorsqu'une version mineure présente des bogues importants ou des problèmes de sécurité qui ont été résolus dans une version mineure ultérieure. Dans le cas peu probable où de tels cas se produiraient, nous en informerons les clients et les informerons du plan et du calendrier de retraite. Des circonstances spécifiques peuvent dicter des délais différents en fonction de la situation.

> **Note**
>
> Des mises à jour critiques des composants peuvent être mises à disposition à tout moment. Par exemple, de nouvelles versions peuvent être mises à disposition rapidement pour des raisons de sécurité ou pour apporter des correctifs aux environnements de production. Pour les demandes effectuées Support, le plan de support définit les processus, la gravité et les délais de réponse.

Lorsqu'une version d'un composant est retirée, AWS Mainframe Modernization ne distribue pas ces versions aux clients pour les nouveaux déploiements. Par conséquent, ces versions ne sont pas non plus prises en charge par Support. Les clients exécutant des déploiements de composants existants après la date de retrait de leur version doivent être conscients des risques liés à une telle opération. AWS n'est pas responsable de fournir des mises à jour de sécurité, un support technique ou des correctifs pour les versions de composants retirées. De plus, nous ne supprimons pas automatiquement l'accès ni ne supprimons les ressources de votre environnement. Nous vous recommandons vivement de vérifier les nouvelles versions tous les 3 mois et de mettre à niveau tous les composants de modernisation de votre AWS mainframe vers les versions prises en charge récentes.

Mise à niveau de version

AWS La modernisation du mainframe fournit des versions plus récentes de chaque composant pris en charge afin que vous puissiez rester au courant up-to-date des dernières mises à jour et fonctionnalités de maintenance. Les nouvelles versions peuvent inclure des corrections de bogues, des améliorations de sécurité et d'autres améliorations des composants. Nous vous recommandons de procéder à une mise à niveau régulière pour bénéficier des correctifs de sécurité, des corrections de bogues et des améliorations des fonctionnalités. Lorsque AWS Mainframe Modernization publie une nouvelle version, vous pouvez choisir comment et quand mettre à niveau vos déploiements existants. Il existe deux types de mises à niveau : les mises à niveau de versions majeures et les mises à niveau de versions mineures. En général, une mise à niveau de version majeure de moteur peut introduire des modifications non compatibles avec les applications existantes. Dans ce cas, des modifications substantielles de l'application peuvent être nécessaires pour une mise à niveau de version majeure. En revanche, une mise à niveau de version mineure inclut des modifications qui sont pour la plupart rétrocompatibles avec les applications existantes. Peu ou pas de modifications peuvent être nécessaires pour une mise à niveau de version mineure.

Vous devez effectuer des tests de non-régression avant d'effectuer les mises à niveau de version des composants. Il est recommandé d'utiliser des pipelines de DevOps test et de déploiement. DevOps des pipelines de test peuvent être créés dans le cadre de projets de modernisation et doivent être maintenus pour automatiser les tests d'applications lors de la mise à niveau des composants et des modifications du code des applications. Vous pouvez également utiliser des déploiements bleu/vert ou Canary lors des mises à niveau. Pour en savoir plus sur ces déploiements et la gestion du changement, consultez le site AWS Well-Architected Reliability Pillar.

AWS Présentation de la version de Mainframe Modernisation Refactor with AWS Blu Age

Avec AWS Blu Age Runtime, la version suit un `Major.Minor.Patch` schéma. Par exemple, pour la version d'exécution de AWS Blu Age`4.1.0`, la version principale est 4, la version mineure est 1 et la version patch est 0.

Nous avons l'intention de publier de nouvelles versions majeures du runtime AWS Blu Age lorsque des modifications importantes seront apportées au runtime ou à ses dépendances. AWS Les versions majeures de Blu Age Runtime sont prises en charge pendant au moins 12 mois, sauf si certaines vulnérabilités et expositions courantes (CVEs) apparaissent. Le support couvre les bogues dans les fonctionnalités d'exécution, comme indiqué dans notre documentation. En cas de

dépendance entre Critical et High CVEs (Spring, Java, Tomcat, etc.), la durée de prise en charge des versions majeures est réduite à 6 mois pour High CVEs et à 3 mois pour Critical à compter CVEs de la date de sortie de la nouvelle version d'exécution corrigeant le CVE, sauf indication contraire explicite.

Nous avons l'intention de publier de nouvelles versions mineures de AWS Blu Age tous les mois. Les clients sont tenus de mettre régulièrement à jour les versions pour obtenir les derniers correctifs de sécurité, corrections de bogues et améliorations de fonctionnalités. Les projets actifs qui ne sont pas encore en production doivent adopter la dernière version d'exécution dès qu'elle sera disponible.

De nouveaux correctifs sont fournis dans la dernière version mineure pour la version majeure particulière dans laquelle un problème est signalé. Si vous avez besoin de nouveaux correctifs, vous devez passer à une nouvelle version mineure pour appliquer ces correctifs.

Les versions corrigées pour les versions prises en charge sont fournies uniquement pour corriger les défauts d'exécution critiques absents des versions mineures prises en charge précédentes.

Les pré-versions alpha sont des versions éphémères mises à disposition pour une itération rapide lors des projets de livraison. Les correctifs pour les problèmes identifiés dans les versions préliminaires alpha sont fournis dans les versions mineures ultérieures, car aucun correctif n'est fourni pour les versions préliminaires d'Alpha.

Vous trouverez les dates de sortie et des informations sur chaque version d'exécution dans lethe section called “AWS Notes de publication de Blu Age”.

Les scans de sécurité sont effectués par Amazon Inspector.

Comprenez les applications gérées dans AWS Mainframe Modernization

Si vous êtes nouveau, AWS Mainframe Modernization consultez les rubriques suivantes pour commencer :

- Qu'est-ce que la modernisation AWS du mainframe ?
- Configuration pour la modernisation du AWS mainframe
- Tutoriel : Configuration d'un environnement d'exécution géré pour AWS Blu Age
- Tutoriel : Configuration du runtime géré pour Rocket Software (anciennement Micro Focus)

Une application AWS Mainframe Modernization contient une charge de travail de mainframe migrée. L'application est analogue à une charge de travail sur le mainframe et est associée à un environnement d'exécution. Vous pouvez ajouter des fichiers batch et des ensembles de données aux applications et surveiller les applications pendant leur exécution. Vous créez des AWS Mainframe Modernization applications pour chaque charge de travail que vous migrez. Lorsque vous créez une AWS Mainframe Modernization application, vous spécifiez le moteur sur lequel elle s'exécute lorsque vous la créez. Choisissez AWS Blu Age si vous utilisez le modèle de refactorisation automatique, et choisissez Rocket Software (anciennement Micro Focus) si vous utilisez le modèle de replateforme.

Rubriques

- Création de AWS ressources pour une application migrée
- Création d'une AWS Mainframe Modernization application
- Déployer une AWS Mainframe Modernization application
- Mettre à jour une AWS Mainframe Modernization application
- Supprimer une AWS Mainframe Modernization application
- Soumettre des tâches par lots pour les AWS Mainframe Modernization candidatures
- Annuler les tâches par lots pour les AWS Mainframe Modernization applications
- Importer des ensembles de données pour les AWS Mainframe Modernization applications
- Exporter des ensembles de données pour les AWS Mainframe Modernization applications
- Gérez les transactions pour les AWS Mainframe Modernization applications

- Configuration de l'application gérée par Rocket Software (anciennement Micro Focus)
- Configuration de l'application gérée AWS Blu Age
- AWS Mainframe Modernization référence de définition d'application
- AWS Référence de définition des ensembles de données sur la modernisation du mainframe

Création de AWS ressources pour une application migrée

Pour exécuter votre application migrée dans AWS, vous devez créer des AWS ressources avec d'autres Services AWS. Les ressources que vous devez créer sont les suivantes :

- Un compartiment S3 contenant le code de l'application, la configuration, les fichiers de données et les autres artefacts requis.
- Une base de données Amazon RDS ou Amazon Aurora contenant les données requises par l'application.
- Et AWS KMS key, qui est requis AWS Secrets Manager pour créer et stocker des secrets.
- Un secret du Gestionnaire de Secrets pour contenir les informations d'identification de la base de données.

> **Note**
>
> Chaque application migrée nécessite son propre ensemble de ressources. Il s'agit d'un ensemble minimum. Votre application peut également nécessiter des ressources supplémentaires, telles que les secrets Amazon Cognito ou les files d'attente MQ.

Autorisations requises

Vérifiez que vous disposez des autorisations suivantes :

- `s3:CreateBucket`, `s3:PutObject`
- `rds:CreateDBInstance`
- `kms:CreateKey`
- `secretsmanager:CreateSecret`

Compartiment Amazon S3

Les applications refactorisées et replateformes nécessitent un compartiment Amazon S3 que vous configurez comme suit :

```
bucket-name/root-folder-name/application-name
```

bucket-name

N'importe quel nom respectant les contraintes de dénomination d'Amazon S3. Nous vous recommandons d'inclure le Région AWS nom dans le nom de votre bucket. Assurez-vous de créer le compartiment dans la même région que celle dans laquelle vous prévoyez de déployer l'application migrée.

root-folder-name

Nom requis pour satisfaire aux contraintes de la définition de l'application, que vous créez dans le cadre de l' AWS Mainframe Modernization application. Vous pouvez utiliser le `root-folder-name` pour distinguer les différentes versions d'une application, par exemple V1 et V2.

nom-application

Le nom de votre application migrée, par exemple, PlanetsDemo ou BankDemo.

Base de données

Les applications refactorisées et replateformes peuvent nécessiter une base de données. Vous devez créer, configurer et gérer la base de données conformément aux exigences spécifiques de chaque moteur d'exécution. AWS Mainframe Modernization prend en charge le chiffrement en transit sur cette base de données. Si vous activez le protocole SSL sur votre base de données, assurez-vous de le spécifier `sslMode` dans le secret de la base de données ainsi que les détails de connexion de la base de données. Pour de plus amples informations, veuillez consulter AWS Secrets Manager secret.

Si vous utilisez le modèle de refactorisation AWS Blu Age et que vous avez besoin d'une BluSam base de données, le moteur d'exécution AWS Blu Age nécessite une base de données Amazon Aurora PostgreSQL, que vous devez créer, configurer et gérer. La BluSam base de données est facultative. Créez cette base de données uniquement si votre application l'exige. Pour créer la base de données, suivez les étapes décrites dans la section Création d'un cluster de base de données Amazon Aurora dans le guide de l'utilisateur Amazon Aurora.

Si vous utilisez le modèle de replateforme Rocket Software, vous pouvez créer une base de données Amazon RDS ou Amazon Aurora PostgreSQL. Pour créer la base de données, suivez les étapes décrites dans Création d'une instance de base de données Amazon RDS dans le guide de l'utilisateur Amazon RDS ou dans Création d'un cluster de base de données Amazon Aurora dans le guide de l'utilisateur Amazon Aurora.

Pour les deux moteurs d'exécution, vous devez stocker les informations d'identification de la base de données à AWS Secrets Manager l'aide d'un AWS KMS key pour les chiffrer.

AWS Key Management Service clé

Vous devez stocker les informations d'identification de la base de données de l'application de manière sécurisée dans AWS Secrets Manager. Pour créer un secret dans Secrets Manager, vous devez créer un AWS KMS key. Pour créer une clé KMS, suivez les étapes décrites dans la section Création de clés du Guide du AWS Key Management Service développeur.

Après avoir créé la clé, vous devez mettre à jour la politique de clé pour accorder des autorisations de AWS Mainframe Modernization déchiffrement. Ajoutez les déclarations de politique suivantes :

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "kms:Decrypt",
   "Resource" : "*"
   }
```

AWS Secrets Manager secret

Vous devez stocker les informations d'identification de la base de données de l'application de manière sécurisée dans AWS Secrets Manager. Pour créer un secret, suivez les étapes décrites dans la section Créer un secret de base de données dans le Guide de AWS Secrets Manager l'utilisateur.

AWS Mainframe Modernization prend en charge le chiffrement en transit sur cette base de données. Si vous activez le protocole SSL sur votre base de données, assurez-vous de le spécifier `sslMode` dans le secret de la base de données ainsi que les détails de connexion de la base de données. Vous pouvez spécifier l'une des valeurs suivantes pour `sslMode` : `verify-full`verify-ca, ou`disable`.

Au cours du processus de création de la clé, choisissez Autorisations relatives aux ressources - facultatif, puis sélectionnez Modifier les autorisations. Dans l'éditeur de stratégie, ajoutez une stratégie basée sur les ressources, telle que la suivante, pour récupérer le contenu des champs chiffrés.

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "secretsmanager:GetSecretValue",
   "Resource" : "*"
   }
```

Création d'une AWS Mainframe Modernization application

Utilisez la AWS Mainframe Modernization console pour créer une AWS Mainframe Modernization application. La création d'une application vous permet d'effectuer des tâches avec la charge de travail du mainframe migré.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Création d'une application

Pour créer une application

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'application.
3. Dans la page Applications, choisissez Create application (Créer une application).
4. Sur la page Spécifier les informations de base, dans la section Nom et description, entrez le nom de l'application.
5. (Facultatif) Dans le champ Description de l'application, entrez une description de l'application. Cette description peut vous aider, ainsi que les autres utilisateurs, à identifier l'objectif de l'application.

6. Dans la section Type de moteur, choisissez Blu Age pour le refactoring automatique ou Micro Focus (Rocket) pour le replatforming.
7. Dans la section Clé KMS, choisissez Personnaliser les paramètres de chiffrement si vous souhaitez utiliser une AWS KMS clé gérée par le client. Pour de plus amples informations, veuillez consulter Chiffrement des données arrêté pour le service de modernisation des AWS mainframes.

   > **Note**
   >
   > Par défaut, AWS Mainframe Modernization chiffre vos données à l'aide d'une AWS KMS clé qui les AWS Mainframe Modernization détient et les gère pour vous. Toutefois, vous pouvez choisir d'utiliser une AWS KMS clé gérée par le client.

8. (Facultatif) Choisissez une AWS KMS clé par son nom ou Amazon Resource Name (ARN), ou choisissez Create an AWS KMS key pour accéder à la AWS KMS console et créer une nouvelle AWS KMS clé.
9. (Facultatif) Dans la section Balises, choisissez Ajouter une nouvelle balise pour ajouter une ou plusieurs balises d'application à votre application. Une balise d'application est une étiquette d'attribut personnalisée qui vous aide à organiser et à gérer vos AWS ressources).
10. Choisissez Suivant.
11. Dans la section Ressources et configurations, utilisez l'éditeur intégré pour saisir la définition de l'application. Vous pouvez également choisir Utiliser un fichier JSON de définition d'application dans un compartiment Amazon S3 et indiquer l'emplacement de la définition d'application que vous souhaitez utiliser. Pour plus d'informations, consultez AWS Exemple de définition d'application Blu Age ou Définition de l'application Rocket Software (anciennement Micro Focus).
12. Choisissez Suivant.
13. Sur la page Réviser et créer, passez en revue les informations que vous avez saisies, puis choisissez Créer une application.

Déployer une AWS Mainframe Modernization application

Utilisez la AWS Mainframe Modernization console pour déployer une AWS Mainframe Modernization application. Vous devez déployer vos applications dans un environnement d'exécution avant d'effectuer des tâches.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Déployer une application

Pour exécuter une AWS Mainframe Modernization application, vous devez d'abord la déployer dans un environnement d'exécution. Une application peut avoir plusieurs versions. Chaque version d'une application possède sa propre définition d'application. Pour déployer une application, vous devez spécifier la version que vous souhaitez déployer.

Vous ne pouvez déployer qu'une seule version d'une application donnée à la fois. Si vous déployez une version d'une application, puis décidez de déployer une version différente, vous devez d'abord arrêter l'application si elle est en cours d'exécution.

Pour déployer une application

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'application.
3. Sur la page Applications, choisissez l'application que vous souhaitez déployer.
4. Choisissez Déployer l'application.
5. Dans la section Versions disponibles, choisissez la version que vous souhaitez déployer.
6. Dans la section Environnements, choisissez un environnement d'exécution dans lequel vous souhaitez exécuter votre application.
7. Choisissez Déployer.

Pour déployer une version différente d'une application déployée

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'application.
3. Sur la page Applications, choisissez l'application que vous souhaitez déployer.
4. Dans le menu Actions, choisissez Arrêter l'application.
5. Une fois l'application arrêtée, choisissez Déployer l'application.

6. Dans la section Versions disponibles, choisissez la version que vous souhaitez déployer. Dans la section Environnements, l'environnement dans lequel l'application est déjà déployée est présélectionné.
7. Choisissez Déployer.

Mettre à jour une AWS Mainframe Modernization application

Utilisez la AWS Mainframe Modernization console pour mettre à jour une AWS Mainframe Modernization application. La mise à jour d'une application crée une nouvelle version de l'application.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Mise à jour d'une application

Une AWS Mainframe Modernization application peut avoir plusieurs versions, chacune ayant sa propre définition d'application. Pour mettre à jour une application, fournissez une nouvelle définition d'application. Cela crée une nouvelle version de l'application.

Pour mettre à jour une application

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application que vous souhaitez mettre à jour a été créée.
3. Sur la page Applications, choisissez l'application que vous souhaitez mettre à jour.
4. Sur la page des détails de l'application, dans la section Définition actuelle, choisissez Modifier pour mettre à jour la définition actuelle de l'application.
5. Sur la page Mettre à jour l'application, utilisez l'éditeur intégré pour mettre à jour la définition actuelle de l'application.

   Vous pouvez également choisir Utiliser un fichier JSON de définition d'application dans un compartiment Amazon S3 et indiquer l'emplacement de la définition d'application que vous souhaitez utiliser. Pour plus d'informations, consultez AWS Exemple de définition d'application Blu Age ou Définition de l'application Rocket Software (anciennement Micro Focus).
6. Lorsque vous avez terminé de mettre à jour la définition de l'application, choisissez Mettre à jour.

> **Note**
>
> Après avoir mis à jour l'application, vous devez la déployer à nouveau. Pour de plus amples informations, veuillez consulter Déployer une AWS Mainframe Modernization application.

Supprimer une AWS Mainframe Modernization application

Vous pouvez supprimer une AWS Mainframe Modernization application d'un environnement à l'aide de la AWS Mainframe Modernization console.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Supprimer une application

Si vous devez supprimer une AWS Mainframe Modernization application alors qu'elle est en cours d'exécution, assurez-vous de l'arrêter au préalable. Vous pouvez voir le statut de la demande sur la page des candidatures.

Pour supprimer une application

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application que vous souhaitez supprimer de l'environnement a été créée.
3. Sur la page Applications, choisissez l'application que vous souhaitez supprimer de l'environnement, puis sélectionnez Actions.
4. (Facultatif) Si le statut de l'application est`Running`, choisissez Arrêter l'application.
5. Choisissez Supprimer de l'environnement.

Le processus de suppression démarre immédiatement.

Soumettre des tâches par lots pour les AWS Mainframe Modernization candidatures

AWS Mainframe Modernization Vous pouvez y soumettre des tâches par lots pour vos candidatures. Vous pouvez soumettre ou annuler des tâches par lots et consulter les informations relatives à l'exécution des tâches par lots. Chaque fois que vous soumettez un traitement par lots, une exécution de traitement par lots distincte est AWS Mainframe Modernization créée. Vous pouvez surveiller l'exécution de cette tâche. Vous pouvez rechercher des tâches par lots par nom et fournir des fichiers JCL ou des fichiers de script aux tâches par lots.

> **Important**
>
> Si vous annulez un travail par lots, cela ne le supprime pas. Cela annule une exécution particulière du traitement par lots. Les enregistrements des tâches par lots restent disponibles pour que vous puissiez les consulter dans les détails de l'exécution des tâches par lots.

Si votre traitement par lots nécessite l'accès à un ou plusieurs ensembles de données, utilisez la AWS Mainframe Modernization console pour importer les ensembles de données. Pour de plus amples informations, veuillez consulter Importer des ensembles de données pour les AWS Mainframe Modernization applications.

Ces instructions supposent que vous avez effectué les étapes dans Configuration pour la modernisation du AWS mainframe et dansCréation d'une AWS Mainframe Modernization application.

Rubriques

- Soumettre une tâche par lots
- Redémarrer un traitement par lots

Soumettre une tâche par lots

Pour soumettre une tâche par lots

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application pour laquelle vous souhaitez soumettre un traitement par lots a été créée.

3. Sur la page Applications, choisissez l'application pour laquelle vous souhaitez soumettre un traitement par lots.

> **Note**
>
> Avant de pouvoir soumettre un traitement par lots à une application, vous devez déployer l'application avec succès.

4. Sur la page des détails de l'application, sélectionnez Batch jobs.
5. Choisissez Soumettre une tâche.
6. Dans la section Sélectionnez un script, choisissez un script. Vous pouvez rechercher le script que vous souhaitez par son nom.
7. Choisissez Soumettre une tâche.

Redémarrer un traitement par lots

Pour redémarrer un traitement par lots

> **Important**
>
> Le redémarrage par lots est disponible sur les versions de moteur suivantes :
>
> - Version 8.0.6 ou ultérieure du moteur d'environnement Micro Focus (Rocket). Vous devez également disposer d'un EFS ou d'un système de FSx fichiers attaché à votre environnement.
> - AWS Version 4.3.0 ou ultérieure du moteur d'environnement Blu Age. Vous devez également disposer d'un EFS ou d'un système de FSx fichiers attaché s'il s'agit d'un environnement HA.

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application et votre traitement par lots ont été créés.
3. Sur la page Applications, choisissez l'application dans laquelle vous souhaitez redémarrer un traitement par lots.

4. Sur la page des détails de l'application, sélectionnez Batch jobs.
5. Sélectionnez le traitement par lots que vous souhaitez redémarrer dans la liste générée. Accédez au menu Actions, puis choisissez Redémarrer le travail.
6. Spécifiez la manière dont vous souhaitez redémarrer le traitement par lots. Vous pouvez effectuer les opérations suivantes pour le moteur d'environnement Micro Focus (Rocket) et le moteur d'environnement AWS Blu Age :

   - Pour le moteur d'environnement Micro Focus (Rocket), vous pouvez choisir de redémarrer depuis le début ou de redémarrer en utilisant steps ou procsteps.
     - L'option Redémarrer depuis le début vous permet de recommencer toutes les étapes d'un traitement par lots depuis le début.
     - Redémarrer à l'aide des étapes ou de l'option procsteps vous permet de choisir une étape ou une étape de procédure spécifique que vous souhaitez redémarrer, et éventuellement une étape ou une étape de procédure que vous souhaitez terminer.

   > **Note**
   >
   > L'étape finale ou le procstep doit être supérieur ou égal à l'étape de début ou au numéro du procstep.

   - Pour le moteur d'environnement AWS Blu Age, vous pouvez soit redémarrer l'exécution la plus récente d'une tâche par lots à partir d'une étape JCL/PROC précédemment échouée, soit effectuer un redémarrage différé en contournant les étapes précédemment réussies.
     - Vous pouvez choisir un nom d'étape spécifique que vous souhaitez redémarrer.
     - Vous pouvez éventuellement utiliser l'étape Ignorer pour contourner l'étape sélectionnée et recommencer à partir de l'étape suivante du flux de travail.
7. Choisissez Soumettre une tâche.

Annuler les tâches par lots pour les AWS Mainframe Modernization applications

AWS Mainframe Modernization Vous pouvez y annuler des tâches par lots pour vos applications. Vous pouvez consulter les détails relatifs aux exécutions de tâches par lots. Chaque fois que vous

soumettez un traitement par lots, une exécution de traitement par lots distincte est AWS Mainframe Modernization créée. Vous pouvez surveiller l'exécution de cette tâche. Vous pouvez rechercher des tâches par lots par nom et fournir des fichiers JCL ou des fichiers de script aux tâches par lots.

> **Important**
>
> Si vous annulez un travail par lots, cela ne le supprime pas. Cela annule une exécution particulière du traitement par lots. Les enregistrements des tâches par lots restent disponibles pour que vous puissiez les consulter dans les détails de l'exécution des tâches par lots.

Annuler une tâche par lots

Lorsque vous annulez un traitement par lots, cela ne supprime pas un traitement par lots, mais l'exécution des tâches associées à ce traitement par lots. Vous pouvez toujours consulter les détails de votre traitement par lots.

Pour annuler un traitement par lots

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région contenant l'application pour vos tâches par lots.
3. Dans la liste des tâches par lots, recherchez et sélectionnez la tâche par lots que vous souhaitez annuler.
4. Choisissez Actions, puis Annuler la tâche.
5. Choisissez Annuler le traitement par lots.

Cela annulera toutes les tâches par lots que vous aviez planifiées d'exécuter.

Importer des ensembles de données pour les AWS Mainframe Modernization applications

AWS Mainframe Modernization Vous pouvez y importer des ensembles de données à utiliser avec vos applications. Vous pouvez spécifier les ensembles de données à importer dans un fichier JSON stocké dans un compartiment Amazon S3, ou vous pouvez spécifier les valeurs de configuration

des ensembles de données séparément. Après avoir importé les ensembles de données, vous pouvez consulter les détails de la tâche d'importation pour confirmer que les ensembles de données souhaités ont été importés. Tous les ensembles de données catalogués pour une application sont répertoriés ensemble dans la console.

Utilisez la AWS Mainframe Modernization console pour importer des ensembles de données pour une AWS Mainframe Modernization application.

Ces instructions supposent que vous avez effectué les étapes dans Configuration pour la modernisation du AWS mainframe et dansCréation d'une AWS Mainframe Modernization application.

Importer un ensemble de données

Pour importer un ensemble de données

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application pour laquelle vous souhaitez importer des ensembles de données a été créée.
3. Sur la page Applications, choisissez l'application pour laquelle vous souhaitez importer des ensembles de données.
4. Sur la page des détails de l'application, sélectionnez Ensembles de données.
5. Choisissez Importer.
6. Effectuez l'une des actions suivantes :
   - Choisissez Utiliser le fichier JSON de configuration d'ensemble de données dans un compartiment Amazon S3 et indiquez l'emplacement de la configuration de l'ensemble de données.
   - Choisissez Spécifier les valeurs de configuration de l'ensemble de données séparément avec une configuration guidée. Reportez-vous the section called "Référence de définition de l'ensemble de données" aux détails de définition spécifiques.

     Entrez le nom, l'organisation de l'ensemble de données (VSAM, GDG, PO, PS), l'emplacement et l'emplacement externe d'Amazon S3, ainsi que les paramètres pour chaque valeur de configuration de l'ensemble de données. Dans la configuration guidée, vous pouvez également choisir Generate JSON pour revoir la configuration JSON à partir de vos entrées.
7. Sélectionnez Envoyer.

Exporter des ensembles de données pour les AWS Mainframe Modernization applications

AWS Mainframe Modernization Vous pouvez ainsi exporter des ensembles de données à utiliser avec vos applications. Vous pouvez spécifier les ensembles de données à exporter dans un fichier JSON stocké dans un compartiment Amazon S3, ou vous pouvez spécifier les valeurs de configuration des ensembles de données séparément. Après avoir exporté les ensembles de données, vous pouvez consulter les détails de la tâche d'exportation pour confirmer que les ensembles de données souhaités ont été exportés.

Utilisez la AWS Mainframe Modernization console pour exporter les ensembles de données d'une AWS Mainframe Modernization application.

Ces instructions supposent que vous avez effectué les étapes dans Configuration pour la modernisation du AWS mainframe et dansCréation d'une AWS Mainframe Modernization application.

Exporter un ensemble de données

Pour exporter un ensemble de données

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application pour laquelle vous souhaitez importer des ensembles de données a été créée.
3. Sur la page Applications, choisissez l'application pour laquelle vous souhaitez exporter des ensembles de données.
4. Sur la page des détails de l'application, sélectionnez Ensembles de données.
5. Cliquez sur Exporter.
6. Effectuez l'une des actions suivantes :
    - Choisissez Utiliser le fichier JSON de configuration d'ensemble de données dans un compartiment Amazon S3 et indiquez l'emplacement de la configuration de l'ensemble de données.
    - Choisissez Spécifier les valeurs de configuration de l'ensemble de données séparément avec une configuration guidée. Pour de plus amples informations, veuillez consulter the section called "Référence de définition de l'ensemble de données".

Entrez le nom du jeu de données, l'emplacement externe d'Amazon S3 et les paramètres pour chaque valeur de configuration de l'ensemble de données. Dans la configuration guidée, vous pouvez également choisir Generate JSON pour revoir la configuration JSON à partir de vos entrées.

7. Sélectionnez Envoyer.

Gérez les transactions pour les AWS Mainframe Modernization applications

AWS Mainframe Modernization Vous pouvez exécuter une application, sur demande, en même temps que de nombreux autres utilisateurs qui soumettent des demandes pour exécuter la même application en utilisant les mêmes fichiers et programmes. Une transaction unique consiste en un ou plusieurs programmes d'application qui effectuent le traitement nécessaire.

Ces instructions supposent que vous avez effectué les étapes dans Configuration pour la modernisation du AWS mainframe et dansCréation d'une AWS Mainframe Modernization application.

Gérez les transactions pour les applications

Pour gérer les transactions pour les applications

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application que vous souhaitez exécuter a été créée.
3. Sur la page Applications, choisissez l'application dans laquelle vous souhaitez gérer les transactions.
4. Dans l'onglet Transactions, sous Ressources relatives aux transactions, choisissez le mode d'affichage de vos ressources dans la liste déroulante. Vous pouvez afficher les ressources en fonction des ressources de transaction, des groupes, des listes ou SITs.

   - Les ressources de transaction vous permettent de choisir le type de ressource en fonction des définitions de fichiers, des définitions de transactions, des définitions de programmes ou des définitions de files d'attente de données transitoires.

> **Note**
>
> Le AWS Mainframe Modernization service prend en charge des types de ressources supplémentaires pour gérer les transactions des applications et est accessible depuis la console.

- Les groupes sont un ensemble de ressources transactionnelles. Vous pouvez choisir les groupes que vous souhaitez associer à votre ressource de transaction.
- Les listes sont des ensembles ordonnés de groupes. Vous pouvez voir toutes vos ressources et groupes de transactions dans une vue de liste. La liste de démarrage détermine les ressources qui sont chargées lors de l'initialisation du serveur.
  - Avec le moteur de refactorisation AWS Blu Age, vous spécifiez les listes à inclure au démarrage. Il n'y a pas de limite au nombre de listes.
  - Avec le moteur de replateforme Rocket Software, vous pouvez spécifier jusqu'à quatre listes dans un SIT.
- La table d'initialisation du système (SIT) affiche toutes les configurations de transaction disponibles. Vous pouvez le trouver en SITs fonction des propriétés (nom, description et listes de démarrage). Vous pouvez également choisir des listes à associer au SIT de votre choix.

> **Note**
>
> SITs ne s'appliquent qu'au moteur de replateforme Rocket Software.

5. Choisissez une ressource de transaction pour afficher toutes les informations relatives à la ressource. Vous pouvez également afficher tous les attributs associés à votre ressource de transaction.

Configuration de l'application gérée par Rocket Software (anciennement Micro Focus)

Vous pouvez configurer vos applications avec le moteur d'exécution Rocket Software pour personnaliser des propriétés supplémentaires, notamment les intégrations.

Rubriques

- Intégrations tierces prises en charge pour Rocket Software

- Imprimantes

Intégrations tierces prises en charge pour Rocket Software

Pour utiliser des intégrations tierces, votre environnement géré AWS Mainframe Modernisation doit utiliser une version du moteur Rocket Software prenant en charge ce type de configuration. Les versions du moteur avec le suffixe R (par exemple, la version 9.0.9.R) sont prises en charge. Cela signifie que la version 9.0.9.R du moteur inclut le support d'installation client pour les intégrations tierces, mais pas la version 9.0.9.

Imprimantes

Les ressources de l'imprimante sont configurées via la définition de l'application Rocket Software, comme décrit dans la the section called "Imprimantes : en option" section.

Une définition d'imprimante peut définir un module de sortie personnalisé ou fourni par un service pour l'imprimante. Voici quelques exemples de configurations possibles du module de sortie :

1. Exemple de service de chargement binaire fourni.

```
...
{
    "name": "p1",
    "classes": [
        "AB"
    ],
    "description": "Using service managed LRS Queue exit module",
    "exit-module": {
        "name": "lrsprte6"
    }
},
...
```

2. Exemple de fourniture de fichiers binaires à partir de S3.

```
"exit-module": {
        "name": "s3Exit",
        "module": "${s3-source}/3pa/s3Exit.so"
}
```

3. Exemple de fourniture de fichiers binaires à partir d'EFS.

> **Note**
>
> Pour utiliser le montage EFS, il doit être attaché lors de la création de l'environnement, ainsi que certaines valeurs supplémentaires à définir, telles que`program-path`.

```
...
"batch-settings": {
    "jes-printers": [
        {
            "name": "p3",
            "classes": [
                "EF"
            ],
            "description": "Using binary from customer provided exit module on EFS
 Mount",
            "exit-module": {
                "name": "efsExit"
            }
        }
    ],
    "program-path": "$EFS_MOUNT/path/to/directory/containing/binaries/"
},
"runtime-settings": {
    "environment-variables": {
        "EFS_MOUNT": "/m2/mount/efs"
    }
}
...
```

File d'attente LRS - optionnelle

Pour utiliser la file d'attente LRS, vous devez utiliser un moteur Rocket Software qui prend en charge les artefacts tiers (c'est-à-dire les moteurs se terminant par`.R`). Outre la configuration d'une imprimante avec un module de sortie pointé `lrsprte6` comme nom d'entrée du module de sortie, la file d'attente LRS nécessite une variable d'environnement supplémentaire, telle que définie dans le bloc « runtime-settings » préexistant de la définition de l'application Rocket Software.

ADRESSE_LRSQ

(Obligatoire) Spécifie l'adresse du serveur LRS à laquelle le module de sortie d'impression LRSQ doit être envoyé.

Imprimantes LRS - La configuration d'une imprimante LRS nécessite la définition d'une imprimante JES, comme indiqué dans la the section called “Imprimantes : en option” section.

En outre, la LRSQ_ADDRESS doit être spécifiée dans le `runtime-settings` champ de la définition de l'application.

```
"runtime-settings": {
    "environment-variables": {
        "LRSQ_ADDRESS": "<lrsq-address>"
    }
}
```

Configuration de l'application gérée AWS Blu Age

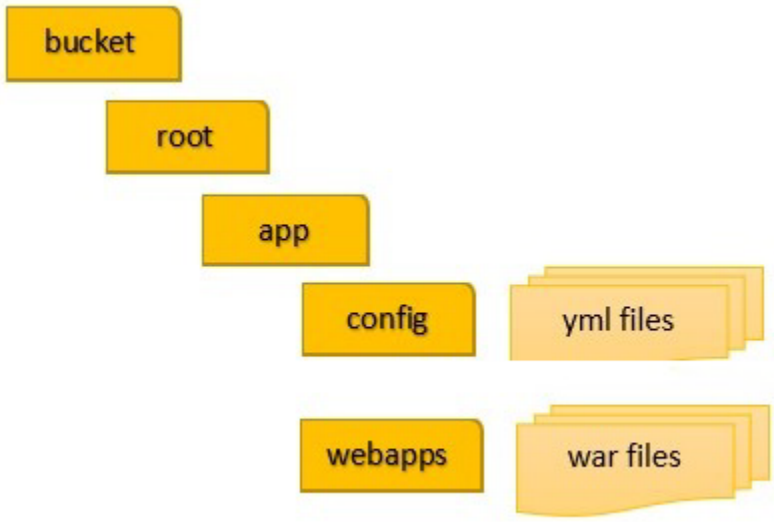
Vous pouvez configurer votre application pour inclure l'accès aux anciens utilitaires. Vous pouvez également personnaliser des propriétés supplémentaires. Pour comprendre ce que vous pouvez configurer et où, consultez la the section called “Structure des applications gérées par AWS Blu Age” section pour comprendre la structure globale d'une application AWS Blu Age modernisée.

Rubriques

- Structure des applications gérées par AWS Blu Age
- Configuration de l'accès aux utilitaires pour les applications gérées
- Ajoutez des propriétés de configuration pour l'application gérée avec le moteur AWS Blu Age

Structure des applications gérées par AWS Blu Age

Si vous utilisez le modèle de refactorisation AWS Blu Age, le moteur d'exécution AWS Blu Age attend la structure suivante dans le `application-name` dossier de votre compartiment S3 :



config

Contient les fichiers YAML de votre projet. Il s'agit des fichiers YAML spécifiques à votre application, généralement appelés quelque chose comme ça, `application-planetsdemo.yaml` et non le `application-main.yaml` fichier que AWS Mainframe Modernization fournit et configure automatiquement pour vous.

applications Web

Contient les `war` fichiers de votre candidature. Ces fichiers sont le résultat du processus de modernisation.

Une application peut également comporter les dossiers facultatifs suivants :

jics/sql

Contient le `initJics.sql` script qui initialise la base de données JICS pour votre application.

scripts

Contient des scripts d'application, que vous pouvez également fournir directement dans les `war` fichiers.

sql

Contient des fichiers SQL d'application, que vous pouvez également fournir directement dans les `war` fichiers.

lnk

Contient les fichiers LNK de l'application, que vous pouvez également fournir directement dans les `war` fichiers.

supplémentaire

Contient des fichiers jar qui peuvent fournir des fonctionnalités supplémentaires à l'application modernisée.

Gérer les options Java d'une application

Pour gérer certaines options Java de l'application, ajoutez un fichier de propriétés nommé `tomcat.properties` `application-name` dans le dossier. Ce fichier peut avoir trois propriétés : `xms` qui spécifie la consommation de mémoire Java minimale`xmx`, qui spécifie la consommation de mémoire Java maximale et `dnscachettl` qui gère la durée du cache pour les résolutions DNS. Voici un exemple du contenu d'un `tomcat.properties` fichier valide.

```
xms=512M
xmx=1G
dnscachettl=5
```

Les valeurs que vous spécifiez pour les deux premières propriétés peuvent être exprimées dans l'une des unités suivantes :

- Octets : ne spécifiez pas d'unité.
- Kilo-octets : ajoutez un K à la valeur.
- Mégaoctets : ajoutez un M à la valeur.
- Gigaoctets : ajoutez un G à la valeur.

La valeur de la troisième propriété représente la durée du cache en secondes et peut avoir la valeur -1 (cache permanent) ou comprise entre 0 (jamais de cache) et 999. Dans le contexte des déploiements d'applications gérées, la valeur par défaut est -1.

Configuration de l'accès aux utilitaires pour les applications gérées

Lorsque vous refactorisez une application mainframe avec AWS Blu Age, vous devrez peut-être fournir un support pour divers programmes utilitaires de plate-forme existants, tels que IDCAMS, INFUTILB, SORT, etc., si votre application en dépend. AWS Le refactoring de Blu Age fournit

cet accès grâce à une application Web dédiée qui est déployée parallèlement à des applications modernisées. Cette application Web nécessite un fichier de `application-utility-pgm.yml` configuration que vous devez fournir. Si vous ne fournissez pas ce fichier de configuration, l'application Web ne pourra pas être déployée en même temps que votre application et ne sera pas disponible.

Rubriques

- Propriétés de configuration

Cette rubrique décrit toutes les propriétés possibles que vous pouvez spécifier dans le fichier de `application-utility-pgm.yml` configuration, ainsi que leurs valeurs par défaut. La rubrique décrit les propriétés obligatoires et facultatives. L'exemple suivant est un fichier de configuration complet. Il répertorie les propriétés dans l'ordre que nous recommandons. Vous pouvez utiliser cet exemple comme point de départ pour votre propre fichier de configuration.

```
# If the datasource support mode is not static-xa, spring JTA transactions
 autoconfiguration must be disabled
 spring.jta.enabled: false
 logging.config: 'classpath:logback-utility.xml'

 # Encoding
 encoding: cp1047

 # Encoding to be used by INFUTILB and DSNUTILB to generate and read SYSPUNCH files
 sysPunchEncoding: cp1047

 # Utility database access
 spring.aws.client.datasources.primary.secret: `arn:aws:secretsmanager:us-
west-2:111122223333:secret:business-FfmXLG`

 treatLargeNumberAsInteger: false

 # Zoned mode : valid values = EBCDIC_STRICT, EBCDIC_MODIFIED, AS400
 zonedMode: EBCDIC_STRICT

 jcl.type: mvs

 # Unload properties
 # For date/time: if use database configuration is enabled, formats are ignored
 # For nbi; use hexadecimal syntaxe to specify the byte value
 unload:
```

```
  sqlCodePointShift: 384
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
  chunkSize:500
  fetchSize: 500
  varCharIsNull: false
  columnFiller: space

# Load properties
# Batch size for DSNUTILB Load Task
load:
  sqlCodePointShift: 384
  batchSize: 500
  format:
    localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
    dbDate: yyyy-MM-dd
    localTime: 'HH:mm:ss|HH.mm.ss'
    dbTime: 'HH:mm:ss'

table-mappings:
  TABLE_1_NAME : LEGACY_TABLE_1_NAME
  TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

Propriétés de configuration

Vous pouvez spécifier les propriétés suivantes dans votre fichier de configuration.

spring.jta.enabled

(Facultatif) Contrôle si le support JTA est activé. Pour les utilitaires, nous vous recommandons de définir cette valeur sur`false`.

```
spring.jta.enabled : false
```

logging.config

(Obligatoire) Spécifie le chemin d'accès au fichier de configuration de l'enregistreur dédié. Nous vous recommandons d'utiliser le nom `logback-utility.xml` et de fournir ce fichier dans le cadre de l'application modernisée. La méthode courante pour organiser ces fichiers consiste à placer tous les fichiers de configuration de l'enregistreur au même endroit, généralement dans le sous-dossier contenant le dossier `/config/logback` contenant les fichiers de configuration YAML. `/config` Pour plus d'informations, consultez le Chapitre 3 : Configuration de Logback dans la documentation Logback.

```
logging.config : classpath:logback-utility.xml
```

encoding

(Obligatoire) Spécifie le jeu de caractères utilisé par le programme utilitaire. Dans la plupart des cas, lorsque vous migrez depuis des plateformes z/OS, ce jeu de caractères est une variante EBCDIC et doit correspondre au jeu de caractères configuré pour les applications modernisées. La valeur par défaut si elle n'est pas définie est`ASCII`.

```
encoding : cp1047
```

sysPunchEncoding

(Facultatif) Spécifie le jeu de caractères utilisé par INFUTILB et DSNUTILB pour générer et lire les fichiers SYSPUNCH. Si vous utilisez les fichiers SYSPUNCH de l'ancienne plateforme tels quels, cette valeur doit être une variante EBCDIC. La valeur par défaut si elle n'est pas définie est`ASCII`.

```
sysPunchEncoding : cp1047
```

Configuration de source de données

Certains utilitaires liés aux bases de données, tels que LOAD et UNLOAD, nécessitent l'accès à une base de données cible via une source de données. Comme les autres définitions de sources de données dans AWS Mainframe Modernization, cet accès nécessite que vous AWS Secrets Manager utilisiez. Les propriétés qui pointent vers les secrets appropriés dans Secrets Manager sont les suivantes :

Source de données principale

Il s'agit de la base de données d'applications métier principale.

spring.aws.client.datasources.primary.secret

(Facultatif) Spécifie le secret dans Secrets Manager qui contient les propriétés de la source de données.

```
spring.aws.client.datasources.primary.secret: datasource-secret-ARN
```

spring.aws.client.datasources.primary.dbname

(Facultatif) Spécifie le nom de la base de données cible s'il n'est pas fourni directement dans le secret de base de données, avec la `dbname` propriété.

```
spring.aws.client.datasources.primary.dbname: target-database-name
```

spring.aws.client.datasources.primary.type

(Facultatif) Spécifie le nom complet de l'implémentation du pool de connexions à utiliser. La valeur par défaut est `com.zaxxer.hikari.HikariDataSource`.

```
spring.aws.client.datasources.primary.type: target-datasource-type
```

Si le type de la source de données principale est`com.zaxxer.hikari.HikariDataSource`, vous pouvez spécifier des propriétés supplémentaires comme suit :

spring.datasource.primary. [nom_propriété]

(Facultatif) Vous pouvez utiliser ce format pour spécifier des propriétés supplémentaires pour configurer l'implémentation d'un pool de connexions à une source de données principale.

Voici un exemple de source de données principale de ce type`com.zaxxer.hikari.HikariDataSource`.

```
spring:
   datasource:
       primary:
         autoCommit: XXXX
         maximumPoolSize: XXXX
         keepaliveTime: XXXX
         minimumIdle: XXXX
```

```
        idleTimeout: XXXX
        connectionTimeout: XXXX
        maxLifetime: XXXX
```

Autres sources de données utilitaires

Outre la source de données principale, vous pouvez fournir d'autres sources de données utilitaires.

spring.aws.client.utility.pgm.datasources.names

(Facultatif) Spécifie la liste des noms des sources de données utilitaires.

```
spring.aws.client.utility.pgm.datasources.names: dsname1, dsname2, dsname3
```

spring.aws.client.utility.pgm.sources de données. [nom de domaine] .secret

(Facultatif) Spécifie l'ARN secret dans SSM qui héberge les propriétés de la source de données. Indiquez [dsname] dans la liste des noms spécifiée dans`spring.aws.client.utility.pgm.datasources.names`.

```
spring.aws.client.utility.pgm.datasources.dsname1.secret: datasource-secret-ARN
```

spring.aws.client.utility.pgm.sources de données. [nom de domaine] .dbname

(Facultatif) Spécifie le nom de la base de données cible s'il n'est pas fourni directement dans le secret de base de données à l'aide de la `dbname` propriété. Indiquez [dsname] dans la liste des noms spécifiée dans`spring.aws.client.utility.pgm.datasources.names`.

```
spring.aws.client.utility.pgm.datasources.dsname1.dbname: target-database-name
```

spring.aws.client.utility.pgm.sources de données. [nom de domaine] .type

(Facultatif) Spécifie le nom complet de l'implémentation du pool de connexions à utiliser. La valeur par défaut est `com.zaxxer.hikari.HikariDataSource`. Indiquez [dsname] dans la liste des noms spécifiée dans`spring.aws.client.utility.pgm.datasources.names`.

```
spring.aws.client.utility.pgm.datasources.dsname1.type: target-datasource-type
```

Si le type de source de données utilitaire est`com.zaxxer.hikari.HikariDataSource`, vous pouvez fournir des propriétés supplémentaires comme suit :

spring. Source de données. [nom de domaine]. [nom_propriété]

(Facultatif) Spécifie un ensemble de propriétés supplémentaires pour configurer l'implémentation d'un pool de connexions à une source de données utilitaire. Indiquez [dsname] dans la liste des noms spécifiée dans`spring.aws.client.utility.pgm.datasources.names`. Spécifiez les propriétés au format suivant : `property_name : value`

Voici un exemple de sources de données utilitaires supplémentaires de type `com.zaxxer.hikari.HikariDataSource` :

```
spring:
   datasource:
       dsname1:
         connectionTimeout: XXXX
         maxLifetime: XXXX
       dsname2:
         connectionTimeout: XXXX
         maxLifetime: XXXX
       dsname3:
         connectionTimeout: XXXX
         maxLifetime: XXXX
```

treatLargeNumberAsInteger

(Facultatif) Lié aux spécificités du moteur de base de données Oracle et à l'utilisation des DSNTEP4 utilitaires DSNTEP2/. Si vous définissez cet indicateur sur true, les grands nombres provenant de la base de données Oracle (NUMBER (38,0)) sont traités comme des entiers. Par défaut : `false`

```
treatLargeNumberAsInteger : false
```

Mode zoné

(Facultatif) Définit le mode zoné pour coder ou décoder les types de données zonés. Ce paramètre influence la façon dont les chiffres des signes sont représentés. Les valeurs suivantes sont valides :

- EBCDIC_STRICT : Par défaut. Utilisez une définition stricte pour la gestion des panneaux. Selon que le jeu de caractères est EBCDIC ou ASCII, la représentation numérique des signes utilise les caractères suivants :

    - Caractères EBCDIC correspondant à des octets (`Cn+Dn`) pour représenter des plages de chiffres positifs et négatifs (`+0`à`+9`, `-0` à`-9`). Les caractères sont affichés sous la forme `{I`, `A à}`, `J à R`
    - Caractères ASCII correspondant à des octets (`3n+7n`) pour représenter des plages de chiffres positifs et négatifs (`+0`à`+9`, `-0` à`-9`). Les caractères sont affichés comme `0 9` suit `p : y`
- EBCDIC_MODIFIED : utilisez une définition modifiée pour la gestion des signes. Pour l'EBDIC et l'ASCII, la même liste de caractères représente les chiffres du signe, c'est-à-dire `+9` mappés `+0` vers `{` + vers `I` et mappés `A` vers `+ -0` vers`-9`. `} J R \`
- AS400 : À utiliser pour les actifs existants modernisés provenant des plateformes iSeries (AS400).

```
zonedMode:EBCDIC_STRICT
```

jcl type

(Facultatif) Indique le type existant de scripts JCL modernisés. L'utilitaire IDCAMS utilise ce paramètre pour personnaliser le code de retour si le JCL invoquant est de type. `vse` Les valeurs valides sont les suivantes :

- `mvs` (par défaut)
- `vse`

```
jcl.type : mvs
```

Propriétés associées aux utilitaires de déchargement de base de données

Utilisez ces propriétés pour configurer les utilitaires qui déchargent les tables de base de données vers des ensembles de données. Toutes les propriétés suivantes sont facultatives.

Cet exemple montre toutes les propriétés de déchargement possibles.

```
# Unload properties
 # For date/time: if use database configuration is enabled, formats are ignored
 # For nbi; use hexadecimal syntaxe to specify the byte value
 unload:
 sqlCodePointShift: 0
 nbi:
```

```
whenNull: "6F"
whenNotNull: "00"
useDatabaseConfiguration: false
format:
date: MM/dd/yyyy
time: HH.mm.ss
timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
chunkSize: 0
fetchSize: 0
varCharIsNull: false
columnFiller: space
```

sqlCodePointShift

(Facultatif) Spécifie une valeur entière qui représente le décalage de points de code SQL utilisé sur les données. La valeur par défaut est 0. Cela signifie qu'aucun changement de point de code n'est effectué. Alignez ce paramètre avec le paramètre de décalage des points de code SQL utilisé pour les applications modernisées. Lorsque le décalage des points de code est utilisé, la valeur la plus courante pour ce paramètre est 384.

```
unload.sqlCodePointShift: 0
```

nbi

(Facultatif) Spécifie un octet indicateur nul. Il s'agit d'une valeur hexadécimale (sous forme de chaîne) ajoutée à droite de la valeur des données. Les deux valeurs possibles sont les suivantes :

- WhenNull : ajoutez la valeur hexadécimale lorsque la valeur des données est nulle. La valeur par défaut est 6`. Parfois, la valeur la plus élevée FF est utilisée à la place.

  ```
  unload.nbi.whenNull: "6F"
  ```

- whenNotNull: Ajoutez la valeur hexadécimale lorsque la valeur des données n'est pas nulle, mais que la colonne est nullable. La valeur par défaut est `00` (faible valeur).

  ```
  unload.nbi.whenNotNull: "00"
  ```

useDatabaseConfiguration

(Facultatif) Spécifie les propriétés de formatage de la date et de l'heure. Ceci est utilisé pour traiter les objets date/heure dans les requêtes UNLOAD. La valeur par défaut est `false`.

- S'il est défini sur `true`pgmDateFormat, utilise les pgmTimestampFormat propriétéspgmTimeFormat, et du fichier de configuration principal (`application-main.yml`).
- S'il est défini sur`false`, utilise les propriétés de mise en forme de date et d'heure suivantes :
  - `unload.format.date`: Spécifie un modèle de mise en forme de date. La valeur par défaut est `MM/dd/yyyy`.
  - `unload.format.time`: Spécifie un modèle de formatage de l'heure. La valeur par défaut est `HH.mm.ss`.
  - `unload.format.timestamp`: Spécifie un modèle de formatage d'horodatage. La valeur par défaut est `yyyy-MM-dd-HH.mm.ss.SSSSSS`.

Taille du morceau

(Facultatif) Spécifie la taille des segments de données utilisés pour créer des ensembles de données SYSREC. Ces ensembles de données sont la cible de l'opération de déchargement des ensembles de données, avec des opérations parallèles. La valeur par défaut est `0` (pas de morceaux).

```
unload.chunkSize:0
```

Taille de récupération

(Facultatif) Spécifie la taille d'extraction des données. La valeur est le nombre d'enregistrements à récupérer simultanément lorsqu'une stratégie de segmentation de données est utilisée. Par défaut: `0`.

```
unload.fetchSize:0
```

varCharIsNull

(Facultatif) Spécifie comment gérer une colonne varchar non nullable dont le contenu est vide. La valeur par défaut est `false`.

Si vous définissez cette valeur sur`true`, le contenu de la colonne est traité comme une chaîne vide à des fins de déchargement, au lieu d'une seule chaîne d'espace. Définissez cet indicateur sur `true` pour le cas du moteur de base de données Oracle uniquement.

```
unload.varCharIsNull: false
```

Remplisseur de colonnes

(Facultatif) Spécifie la valeur à utiliser pour le remplissage des colonnes déchargées dans les colonnes varchar. Les valeurs possibles sont l'espace ou les valeurs faibles. La valeur par défaut est l'espace.

```
unload.columnFiller: space
```

Propriétés liées au chargement de la base de données

Utilisez ces propriétés pour configurer des utilitaires qui chargent des enregistrements d'ensembles de données dans une base de données cible, par exemple DSNUTILB. Toutes les propriétés suivantes sont facultatives.

Cet exemple montre toutes les propriétés de charge possibles.

```
# Load properties
 # Batch size for DSNUTILB Load Task
 load:
 sqlCodePointShift: 384
 batchSize: 500
 format:
 localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
 dbDate: yyyy-MM-dd
 localTime: HH:mm:ss|HH.mm.ss
 dbTime: HH:mm:ss

 table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

sqlCodePointShift

(Facultatif) Spécifie une valeur entière qui représente le décalage de points de code SQL utilisé sur les données. La valeur par défaut est 0, ce qui signifie que les applications ne modifient aucun point de code. Alignez ce paramètre avec le paramètre de décalage des points de code SQL utilisé pour les applications modernisées. Lorsque vous utilisez des décalages de points de code, la valeur la plus courante pour ce paramètre est 384.

```
load.sqlCodePointShift : 384
```

batchSize

(Facultatif) Spécifie une valeur entière qui représente le nombre d'enregistrements à traiter avant d'envoyer une instruction de lot réelle à la base de données. La valeur par défaut est 0.

```
load.batchSize: 500
```

format

(Facultatif) Spécifie les modèles de mise en forme de date et d'heure à utiliser pour les conversions de date et d'heure lors des opérations de chargement de la base de données.

- `load.format.localDate`: modèle de formatage de date local. La valeur par défaut est `dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd`.
- `load.format.dbDate`: modèle de formatage des dates de base de données. La valeur par défaut est `yyyy-MM-dd`.
- `load.format.localTime`: modèle de formatage de l'heure locale. La valeur par défaut est `HH:mm:ss|HH.mm.ss`.
- `load.format.dbTime`: modèle de formatage horaire de la base de données. La valeur par défaut est `HH:mm:ss`.

mappages de tables

(Facultatif) Spécifie un ensemble de mappages fournis par le client entre les noms de table anciens et modernes. Le programme utilitaire DSNUTILB utilise ces mappages.

Spécifiez les valeurs au format suivant : MODERN_TABLE_NAME : LEGACY_TABLE_NAME

Voici un exemple :

```
table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
 ...
 TABLE_*N*_NAME : LEGACY_TABLE_*N*_NAME
```

> **Note**
>
> Lorsque l'application utilitaire démarre, elle enregistre explicitement tous les mappages fournis.

Ajoutez des propriétés de configuration pour l'application gérée avec le moteur AWS Blu Age

Vous pouvez ajouter un fichier dans le `config` dossier de votre application refactorisée qui vous donnera accès aux nouvelles fonctionnalités du moteur d'exécution AWS Blu Age. Vous devez donner un nom à ce fichier`user-properties.yml`. Ce fichier ne remplace pas la définition de l'application mais l'étend. Cette rubrique décrit les propriétés que vous pouvez inclure dans le `user-properties.yml` fichier.

> **Note**
>
> Vous ne pouvez pas modifier certains paramètres car ils sont contrôlés soit par la modernisation du AWS mainframe, soit par la définition de l'application. Tous les paramètres définis dans la définition de l'application pour votre application ont priorité sur les paramètres que vous spécifiez dans`user-properties.yml`.

Pour plus d'informations sur la structure des applications refactorisées, consultez. Structure des applications gérées par AWS Blu Age

Le schéma suivant indique où placer le `user-properties.yml` fichier dans la structure de l'exemple d'application AWS Blu Age, PlanetsDemo.

```
PlanetsDemo-v1/
   ## config/
   #  ## application-PlanetsDemo.yml
   #  ## user-properties.yml
   ## jics/
   ## webapps/
```

Référence sur les propriétés de configuration

Voici la liste des propriétés disponibles. Tous les paramètres sont facultatifs.

Rubriques

- Propriétés de l'application Gapwalk
- Propriétés du batchscript Gapwalk
- Propriétés de Gapwalk Blugen

- Propriétés de la commande Gapwalk CL
- Propriétés du coureur Gapwalk CL
- Propriétés de Gapwalk JHDB
- Propriétés de Gapwalk JICS
- Propriétés d'exécution de Gapwalk
- Propriétés du programme utilitaire Gapwalk
- Autres propriétés

Propriétés de l'application Gapwalk

Intervalle entre Bluesam.fileloading.commit

Facultatif. L'intervalle de validation BluSam.

Type : nombre

Par défaut : 100000

encodage de la carte

Facultatif. Encodage de la carte : à utiliser avec`useControlMVariable`.

Type : chaîne

Par défaut : CP1145

vérifier la taille du fichier d'entrée

Facultatif. Spécifie s'il faut lancer une vérification si la taille du fichier est un multiple de la taille de l'enregistrement.

Type : valeur booléenne

Valeur par défaut : false

database.cursor.overflow.allowed

Facultatif. Spécifie s'il faut autoriser le dépassement du curseur. Réglez sur `true` pour effectuer un appel suivant sur le curseur, quelle que soit sa position. Réglez `false` sur pour vérifier si le curseur est à la dernière position avant d'effectuer un prochain appel sur le curseur. Activé uniquement si le curseur est SCROLLABLE (SENSITIVE ou INSENSITIVE)

Type : valeur booléenne

Valeur par défaut : true

Simplificateur de données. onInvalidNumericDonnées

Facultatif. Comment réagir lors du décodage de données numériques non valides. Les valeurs autorisées sont `rejecttoleratespaces`,`toleratespaceslowvalues`,`toleratemost`.

Type : chaîne

Par défaut : rejeter

defaultKeepExistingDossiers

Facultatif. Spécifie s'il faut définir la valeur précédente par défaut de l'ensemble de données.

Type : valeur booléenne

Valeur par défaut : false

disposition.checkexistence

Facultatif. Spécifie s'il convient de vérifier l'existence du fichier pour Dataset avec DISP SHR ou OLD.

Type : valeur booléenne

Valeur par défaut : false

ExternalSort.Threshold

Facultatif. Le seuil de tri : quand passer au tri externe (fusion).

Type : chaîne

Par défaut:null

`externalSort.threshold: 12MB`

blockSizeDefault

Facultatif. Taille de bloc par défaut à utiliser pour les octets BDW.

Type : nombre

Par défaut : 32760

```
blockSizeDefault: 32760
```

Force HR

Facultatif. Spécifie s'il faut utiliser SYSPRINT lisible par l'homme, soit en sortie de console, soit en sortie de fichier.

Type : valeur booléenne

Valeur par défaut : false

Date forcée

Facultatif. Force la saisie d'une date et d'une heure spécifiques dans la base de données. À utiliser uniquement pendant le développement et les tests.

Par défaut:null

```
forcedDate: 2022-08-26T12:59:58.123456+01:57
```

Date congelée

Facultatif. Gèle la date et l'heure dans la base de données. À utiliser uniquement pendant le développement et les tests.

Valeur par défaut : false

```
frozenDate: false
```

IMS. Messages. Taille étendue

Facultatif. Spécifie s'il faut définir la valeur ExtendedSize pour les messages ims.

Type : valeur booléenne

Valeur par défaut : false

Délai de verrouillage

Facultatif. Délai d'expiration en millisecondes d'une transaction en cas d'impossibilité d'obtenir un verrou dans un délai spécifié.

Type : nombre

Par défaut : 500

MapTransfo. Prefixes

Facultatif. Liste des préfixes à utiliser lors de la transformation des variables ControlM. Chacun d'eux est séparé par une virgule.

Type : chaîne

Par défaut : &, @, % %

requête. useConcatCondition

Facultatif. Spécifie si la condition clé est créée par concaténation de clés ou non.

Type : valeur booléenne

Valeur par défaut : false

Retour sur RTE

Facultatif. Spécifie s'il faut annuler la transaction d'unité d'exécution implicite sur les exceptions d'exécution.

Type : valeur booléenne

Valeur par défaut : false

sctThreadLimit

Facultatif. La limite de threads pour le déclenchement de scripts.

Type : nombre

Par défaut: 5

sqlCodePointShift

Facultatif. Le changement de point de code SQL. Déplace le point de code pour les caractères de contrôle que nous pouvons rencontrer lors de la migration des données d'un SGBDR existant vers un SGBDR moderne. Par exemple, vous pouvez spécifier de `384` faire correspondre le caractère Unicode`\u0180`.

Type : nombre

Par défaut : 0

sqlIntegerOverflowAutorisé

Facultatif. Spécifie s'il faut autoriser le dépassement des nombres entiers SQL, c'est-à-dire s'il est permis de placer des valeurs plus importantes dans la variable hôte.

Type : valeur booléenne

Valeur par défaut : false

stepFailWhenAbend

Facultatif. Spécifie s'il faut déclencher un abend en cas d'échec ou de fin d'exécution d'une étape.

Type : valeur booléenne

Valeur par défaut : true

stopExecutionWhenProgNotFound

Facultatif. Spécifie s'il faut arrêter l'exécution si aucun programme n'est trouvé. S'il est défini sur`true`, interrompt l'exécution si aucun programme n'est trouvé.

Type : valeur booléenne

Valeur par défaut : true

uppercaseUserInput

Facultatif. Spécifie si les entrées utilisateur doivent être en majuscules.

Type : valeur booléenne

Valeur par défaut : true

Contrôle de l'utilisation MVariable

Facultatif. Spécifie s'il faut utiliser la spécification Control-M pour le remplacement des variables.

Type : valeur booléenne

Valeur par défaut : false

jcl.checkpoint.expireTimeout

Facultatif. Spécifie la durée pendant laquelle les points de contrôle JCL doivent être conservés dans le fournisseur de persistance ou dans le registre en mémoire.

Type : nombre

Par défaut : -1

jcl.checkpoint. expireTimeoutUnit

Facultatif. Spécifie l'unité de durée de la `jcl.checkpoint.expireTimeout` propriété. Valeurs constantes enum prises en charge :`java.util.concurrent.TimeUnit`.

Type : chaîne

Par défaut : SECONDS

Propriétés du batchscript Gapwalk

encoding

Facultatif. L'encodage utilisé dans les projets de script par lots (pas avec groovy). Exige un encodage valide `CP1047IBM930`,,`ASCII`,`UTF-8`...

Type : chaîne

Par défaut : ASCII

Propriétés de Gapwalk Blugen

managers.trancode

Facultatif. Le mappage des trancodes du gestionnaire de dialogues. Permet de mapper un code de transaction JICS à un gestionnaire de dialogue. Le format attendu est`trancode1:dialogManager1;trancode2:dialogManager2;`.

Type : chaîne

Par défaut:null

```
managers.trancode: OR12:MYDIALOG1
```

Propriétés de la commande Gapwalk CL

commandes désactivées

Facultatif. Liste des commandes à désactiver, séparées par des virgules. Les valeurs autorisées sont

PGM_BASICRCVMSG,SNDRCVF,CHGVAR,QCLRDTAQ,,RTVJOBA,ADDLFM,ADDPFM,RCVF,OVRDBF,DLTOVR,CP
Utile lorsque vous souhaitez désactiver ou remplacer un programme existant. PGM_BASICest un programme AWS Blu Age Runtime spécifique conçu à des fins de débogage.

Type : chaîne

Par défaut:null

spring.datasource.primary.jndi-name

Facultatif. La principale source de données JNDI (Java Naming And Directory Interface).

Type : chaîne

Par défaut : jdbc/primary

Mode zoné

Facultatif. Mode de codage ou de décodage des types de données zonés. Les valeurs autorisées sontEBCDIC_STRICT/EBCDIC_MODIFIED/AS400.

Type : chaîne

Par défaut : EBCDIC_STRICT

Propriétés du coureur Gapwalk CL

cl.configuration.context.encoding

Facultatif. L'encodage des fichiers CL. Exige un encodage valide CP1047IBM930,,ASCII,UTF-8...

Type : chaîne

Par défaut : CP297

CL. Mode zoné

Facultatif. Mode de codage ou de décodage des commandes du langage de contrôle (CL). Les valeurs autorisées sontEBCDIC_STRICT/EBCDIC_MODIFIED/AS400.

Type : chaîne

Par défaut : EBCDIC_STRICT

Propriétés de Gapwalk JHDB

ims.programs

Facultatif. Liste des programmes IMS à utiliser. Séparez chaque paramètre par un point-virgule (`;`) et chaque transaction par une virgule (). `,` Par exemple : `ims.programs: PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`

Type : chaîne

Par défaut:null

JHDB.Checkpoint Path

Facultatif. Si ce n'`jhdb.checkpointPersistence`est pas le cas`none`, ce paramètre vous permet de configurer le chemin de persistance du point de contrôle (emplacement de stockage du fichier checkpoint.dat). Toutes les données des points de contrôle contenues dans le registre sont sérialisées et sauvegardées dans un fichier (checkpoint.dat) situé dans le dossier fourni. Notez que seules les données de point de contrôle (ScriptID, StepID, position de la base de données et zone de point de contrôle) sont concernées par cette sauvegarde.

Type : chaîne

Par défaut : file :. /configuration/

JHDB.Checkpoint Persistance

Facultatif. Le mode de persistance du point de contrôle. Les valeurs autorisées sont`none`/`add`/`end`. `add`À utiliser pour conserver les points de contrôle lorsqu'un nouveau point est créé et ajouté au registre. `end`À utiliser pour conserver le point de contrôle lors de l'arrêt du serveur. Toute autre valeur désactive la persistance. Notez que chaque fois qu'un nouveau point de contrôle est ajouté au registre, tous les points de contrôle existants sont sérialisés et le fichier est effacé. Il ne s'agit pas d'un ajout aux données existantes du fichier. Ainsi, en fonction du nombre de points de contrôle, cela peut avoir un effet sur les performances.

Type : chaîne

Par défaut: Aucun

jhdb.configuration.context.encoding

Facultatif. Le codage JHDB (base de données hiérarchique Java). Exige une chaîne de codage valide`CP1047`,`IBM930`,`ASCII`,`UTF-8`...

Type : chaîne

Par défaut : CP297

jhdb. identificationCardData

Facultatif. Utilisé pour coder en dur certaines « données de la carte d'identification de l'opérateur » dans le champ MID désigné par le paramètre CARD.

Type : chaîne

Par défaut: ""

jhdb.lterm

Facultatif. Permet de forcer un identifiant de terminal logique commun dans le cas d'une émulation IMS. S'il n'est pas défini, SessionId est utilisé.

Type : chaîne

Par défaut:null

jhdb.metadata.extrapath

Paramètre de configuration qui spécifie un dossier racine supplémentaire spécifique à l'exécution pour les dossiers psbs et dbds.

Type : chaîne

Par défaut : file :. /configuration/

> **Note**
>
> Actuellement, pour des raisons de déploiement, vous devez copier vos répertoires dbds et psbs dans le répertoire de configuration de votre application ou dans un sous-répertoire du répertoire de configuration : par exemple, config/setup
>
> ```
> config
> |- setup
>    |- dbds
>    |- psbs
> ```
>
> et défini dans application-jhdb.yml
> `jhdb.metadata.extrapath: file: ./config/setup/`

jhdb.navigation.cachenexts

Facultatif. Durée du cache (en millisecondes) utilisée dans la navigation hiérarchique pour un SGBDR.

Type : nombre

Par défaut : 5000

jhdb.query. limitJoinUsage

Facultatif. Spécifie s'il faut utiliser le paramètre d'utilisation limite des jointures sur les graphes RDBMS.

Type : valeur booléenne

Valeur par défaut : true

jhdb. use-db-prefix

Facultatif. Spécifie s'il faut activer un préfixe de base de données dans la navigation hiérarchique pour un SGBDR.

Type : valeur booléenne

Valeur par défaut : true

Propriétés de Gapwalk JICS

jics.data. dataJsonInitEmplacement

Facultatif. Emplacement du fichier json préparé par l'analyseur à partir de l'analyse CSD et utilisé pour initialiser la base de données jics,

Type : chaîne

Par défaut: ""

jics.db. dataScriptLocation

Facultatif. Emplacement du script initJics.sql, préparé par Analyzer à partir de l'analyse des exportations CSD depuis le mainframe.

Type : chaîne

Par défaut: ""

jics.db. dataTestQueryEmplacement

Facultatif. Emplacement d'un script SQL contenant une seule requête SQL censée renvoyer un nombre d'objets (par exemple : compter le nombre d'enregistrements dans la table du programme jics). Si le nombre est égal à 0, la base de données sera chargée à l'aide du `jics.db.dataScriptLocation` script, sinon le chargement de la base de données sera ignoré.

Type : chaîne

Par défaut: ""

jics.db. ddlScriptLocation

Facultatif. L'emplacement du script Jics DDL. Permet de lancer le schéma de base de données jics à l'aide d'un script .sql.

Type : chaîne

Par défaut: ""

```
jics.db.ddlScriptLocation: ./jics/sql/jics.sql
```

jics.db. schemaTestQueryEmplacement

Facultatif. Emplacement du fichier SQL qui doit contenir une requête unique renvoyant le nombre d'objets du schéma jics (le cas échéant).

Type : chaîne

Par défaut: ""

jics. runUnitLauncherPool.Enable

Facultatif. Spécifie s'il faut activer le pool de lanceurs d'unités exécutées dans JICS.

Type : valeur booléenne

Valeur par défaut : false

jics. runUnitLauncherTaille de la piscine

Facultatif. Taille du pool de lanceurs d'unités exécutées dans JICS.

Type : nombre

Valeur par défaut : 20

jics. runUnitLauncherPool.Intervalle de validation

Facultatif : intervalle de validation du pool de lanceurs d'unités exécutées dans JICS, exprimé en millisecondes.

Type : nombre

Par défaut: 1000

jics.queues.sqs.region

Facultatif. Le Région AWS pour Amazon SQS, utilisé dans JICS. Il est conseillé de définir la même région de l'application déployée pour des raisons de performances, mais ce n'est pas obligatoire.

Type : chaîne

Par défaut : eu-west-1

jics.xa.agent.timeout

Facultatif. Définit la durée maximale pendant laquelle l'agent xa chargé de gérer les transactions distribuées doit effectuer ses opérations.

Type : nombre

Par défaut:null

mq.queues.sqs.region

Facultatif. Le Région AWS pour le service Amazon SQS MQ.

Type : chaîne

Par défaut : eu-west-3

Exécuteur de tâches. allowCoreThreadTimeOut

Facultatif. Spécifie s'il faut autoriser les threads principaux à expirer dans JCIS. Cela permet une croissance et une réduction dynamiques, même en combinaison avec une file d'attente différente de zéro (étant donné que la taille maximale du pool n'augmentera que lorsque la file d'attente sera pleine).

Type : valeur booléenne

Valeur par défaut : false

Exécuteur de tâches. corePoolSize

Facultatif. Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille du pool principal.

Type : nombre

Par défaut: 5

Exécuteur de tâches. maxPoolSize

Facultatif. Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille maximale du pool (nombre maximal de threads parallèles).

Type : nombre

Par défaut: 10

TaskExecutor.QueueCapacity

Facultatif. Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille de la file d'attente. (= nombre maximum de transactions en attente lorsqu'il `taskExecutor.maxPoolSize` est atteint)

Type : nombre

Par défaut: 50

Propriétés d'exécution de Gapwalk

Méta-données du cache

Facultatif. Spécifie s'il faut mettre en cache les métadonnées de base de données.

Type : valeur booléenne

Valeur par défaut : true

check-groovy-file

Facultatif. Spécifie s'il faut vérifier le contenu des fichiers groovy avant de les enregistrer.

Type : valeur booléenne

Valeur par défaut : true

Statistiques de base de données

Facultatif. Spécifie s'il faut autoriser les générateurs SQL à collecter et à afficher des informations statistiques.

Type : valeur booléenne

Valeur par défaut : false

dateTimeFormat

Facultatif. dateTimeFormat décrit comment répartir la date, l'heure et le type d'horodatage de la base de données dans des entités simplificatrices de données. Les valeurs autorisées sont`ISO/EUR/USA/LOCAL`

Type : chaîne

Par défaut : ISO

dbDateFormat

Facultatif. Format de date cible de la base de données.

Type : chaîne

Par défaut : yyyy-MM-dd

dbTimeFormat

Facultatif. Format horaire cible de la base de données.

Type : chaîne

Par défaut : HH:MM:SS

dbTimestampFormat

Facultatif. Format d'horodatage cible de la base de données.

Type : chaîne

Par défaut : yyyy-MM-dd HH:MM:SS.ssssss

Taille de récupération

Facultatif. La valeur FetchSize pour les curseurs. À utiliser lors de la récupération de données à l'aide de fragments par des utilitaires de chargement/déchargement.

Type : nombre

Par défaut: 10

Forcer la désactivation SQLTrim StringType

Facultatif. Spécifie s'il faut désactiver le découpage de tous les paramètres de chaîne SQL.

Type : valeur booléenne

Valeur par défaut : false

localDateFormat

Facultatif. Liste des formats de date locaux. Séparez chaque format par |.

Type : chaîne

localTimeFormat

Facultatif. Liste des formats d'heure locale. Séparez chaque format par |.

Type : chaîne

localTimestampFormat

Facultatif. Liste des formats d'horodatage locaux. Séparez chaque format par |.

Type : chaîne

Par défaut :

pgmDateFormat

Facultatif. Format de date et d'heure utilisé dans les programmes.

Type : chaîne

Par défaut : yyyy-MM-dd

pgmTimeFormat

Facultatif. Le format horaire utilisé pour l'exécution de pgm (programmes).

Type : chaîne

Par défaut : HH.mm.ss

pgmTimestampFormat

Facultatif. Le format d'horodatage.

Type : chaîne

Par défaut : yyyy-MM-dd-HH .mm.ss.SSSSSSSS

Propriétés du programme utilitaire Gapwalk

jcl type

Facultatif `.jcl`type de fichier. Les valeurs autorisées sont`jcl`/`vse`. Les commandes PRINT/REPRO de l'utilitaire IDCAMS renvoient 4 si le fichier est vide pour un jcl non vse.

Type : chaîne

Par défaut : mvs

listcat.préprocesseur à longueur variable .enabled

Facultatif. Spécifie s'il faut activer le préprocesseur de longueur variable pour la commande LISTCAT.

Type : valeur booléenne

Valeur par défaut : false

listcat.préprocesseur.type à longueur variable

Facultatif. Type d'objets contenus dans le fichier listcat, si vous l'active`listcat.variablelengthpreprocessor.enabled`. Les valeurs autorisées sont`rdw`/`bdw`.

Type : chaîne

Par défaut : rdw

Charger. Taille du lot

Facultatif. Taille du lot de l'utilitaire de chargement.

Type : nombre

Par défaut : 0

Charger .format.DBDate

Facultatif. Format de base de données de l'utilitaire de chargement à utiliser.

Type : chaîne

Par défaut : yyyy-MM-dd

Charger .format.DBTime

Facultatif. Durée d'utilisation de la base de données de l'utilitaire de chargement.

Type : chaîne

Par défaut : HH:MM:SS

Charger .format.localDate

Facultatif. Format de date local de l'utilitaire de chargement à utiliser.

Type : chaîne

Par défaut : dd/MM/yyyy dd.mm.yyyy| |yyyy-mm-dd

Load.format.LocalTime

Facultatif. Format d'heure locale de l'utilitaire de chargement à utiliser.

Type : chaîne

Par défaut : HH:MM:SS|HH.mm.ss

charge. sqlCodePointShift

Facultatif. L'utilitaire SQL Pointshift for Load. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible DB2 provient de Postgresql.

Type : nombre

Par défaut : 0

sysPunchEncoding

Facultatif. Le jeu de caractères de codage Syspunch. Les valeurs prises en charge sont`Cp1047`/`ASCII`.

Type : chaîne

Par défaut : ASCII

treatLargeNumberAsInteger

Facultatif. Spécifie s'il faut traiter les grands nombres comme`Integer`. Ils sont traités comme `BigDecimal` par défaut.

Type : valeur booléenne

Valeur par défaut : false

Déchargez .chunksize

Facultatif. Taille du morceau utilisée pour l'utilitaire de déchargement.

Type : nombre

Par défaut : 0

Décharger .columnFiller

Facultatif. Le remplisseur de colonnes utilitaire de déchargement.

Type : chaîne

Par défaut : espace

Décharger .fetchSize

Facultatif. Vous permet de régler la taille de lecture lorsque vous manipulez des curseurs dans l'utilitaire de déchargement.

Type : nombre

Par défaut : 0

décharger.format.date

Facultatif. Si cette option `unload.useDatabaseConfiguration` est activée, le format de date à utiliser dans l'utilitaire de déchargement.

Type : chaîne

Par défaut : MM/dd/yyyy

unload.format.time

Facultatif. Si cette option `unload.useDatabaseConfiguration` est activée, le format d'heure à utiliser dans l'utilitaire de déchargement.

Type : chaîne

Par défaut : HH.mm.ss

décharger.format.timestamp

Facultatif. Si cette option `unload.useDatabaseConfiguration` est activée, le format d'horodatage à utiliser dans l'utilitaire de déchargement.

Type : chaîne

Par défaut : yyyy-MM-dd-HH .mm.ss.SSSSSSSS

déchargez .nbi. whenNotNull

Facultatif. La valeur de l'indicateur d'octet nul (nbi) à ajouter lorsque la valeur de la base de données n'est pas nulle.

Type : hexadécimal

Par défaut : 00

Décharger .nbi.whennull

Facultatif. La valeur de l'indicateur d'octet nul (nbi) à ajouter lorsque la valeur de la base de données est nulle.

Type : hexadécimal

Par défaut : 6F

déchargez .nbi. writeNullIndicator

Facultatif. Spécifie s'il faut écrire l'indicateur nul dans le fichier de sortie de déchargement.

Type : valeur booléenne

Valeur par défaut : false

décharger. sqlCodePointShift

Facultatif. L'utilitaire SQL Pointshift for Unload. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible DB2 provient de Postgresql.

Type : nombre

Par défaut : 0

décharger. useDatabaseConfiguration

Facultatif. Spécifie s'il faut utiliser la configuration de date ou d'heure de application-main.yml dans l'utilitaire de déchargement.

Type : valeur booléenne

Valeur par défaut : false

décharger. varCharIsNull

Facultatif. Utilisez ce paramètre dans le programme INFTILB. S'il est défini sur, tous les champs non nullables contenant des valeurs vides (espaces) renvoient une chaîne vide. `true`

Type : valeur booléenne

Valeur par défaut : false

Autres propriétés

qtemp.cleanup.threshold. hours

Facultatif. Pour spécifier quand `qtemp.dblog` est activé. Durée de vie de la partition de base de données (en heures).

Type : nombre

Par défaut : 0

qtemp.dblog

Facultatif. S'il faut activer la journalisation de la base de données QTEMP.

Type : valeur booléenne

Valeur par défaut : false

qtemp.uuid.length

Facultatif. La longueur de l'identifiant unique QTEMP.

Type : nombre

Par défaut : 9

quartz.scheduler. stand-by-if-error

Facultatif. Spécifie s'il faut déclencher l'exécution des tâches si le planificateur de tâches est en mode veille. Si vrai, lorsque cette option est activée, l'exécution de la tâche n'est pas déclenchée.

Type : valeur booléenne

Valeur par défaut : false

warmUpCache

Facultatif. Spécifie s'il faut charger toutes les données de la table Datacom dans un cache de préchauffage au démarrage du serveur.

Type : valeur booléenne

Valeur par défaut : false

AWS Mainframe Modernization référence de définition d'application

Dans AWS Mainframe Modernization, vous configurez les applications mainframe migrées dans un fichier JSON de définition d'application, spécifique au moteur d'exécution que vous choisissez. Une définition d'application contient à la fois des informations générales et des informations spécifiques au moteur. Cette rubrique décrit les définitions des applications AWS Blu Age et Rocket Software (anciennement Micro Focus) et identifie tous les éléments obligatoires et facultatifs.

Table des matières

- Section d'en-tête générale
- Présentation de la section consacrée aux définitions
- AWS Exemple de définition d'application Blu Age
- AWS Détails de la définition de Blu Age
  - Écouteur (s) - obligatoire
  - AWS Application Blu Age - obligatoire
  - BluSam - facultatif
  - AWS files d'attente de messages Blu Age - facultatif

  - AWS Configuration EFS du stockage des applications Blu Age (en option)
- Définition de l'application Rocket Software (anciennement Micro Focus)
- Détails de la définition du logiciel Rocket
  - Écouteur (s) - obligatoire
  - Emplacement des ensembles de données : obligatoire
  - Gestionnaire d'authentification et d'autorisation Amazon Cognito : facultatif
  - Gestionnaire LDAP et Active Directory (facultatif)
  - Réglages par lots : obligatoire
  - Réglages CICS - obligatoires
  - Imprimantes : en option
  - Ressources XA - facultatives
  - Paramètres d'exécution : facultatif

Section d'en-tête générale

Chaque définition d'application commence par des informations générales sur la version du modèle et les emplacements des sources. La version actuelle de la définition de l'application est 2.0.

Utilisez la structure suivante pour spécifier la version du modèle et les emplacements des sources.

```
"template-version": "2.0",
    "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket",
                "s3-key-prefix": "v1"
            }
        }
    ]
```

> **Note**
>
> Vous pouvez utiliser la syntaxe suivante si vous souhaitez saisir l'ARN S3 en tant que s3-bucket :

```
"template-version": "2.0",
  "source-locations": [
      {
          "source-id": "s3-source",
          "source-type": "s3",
          "properties": {
              "s3-bucket": "arn:aws:s3:::mainframe-deployment-bucket",
              "s3-key-prefix": "v1"
          }
      }
  ]
```

version du modèle

(Obligatoire) Spécifie la version du fichier de définition de l'application. Ne modifiez pas cette valeur. Actuellement, la seule valeur autorisée est 2,0. Spécifiez `template-version` avec une chaîne.

localisations des sources

Spécifie l'emplacement des fichiers et des autres ressources dont l'application a besoin pendant l'exécution.

identificateur de source

Spécifie le nom de l'emplacement. Ce nom est utilisé pour référencer l'emplacement de la source selon les besoins dans le JSON de définition de l'application.

type de source

Spécifie le type de source. Actuellement, la seule valeur autorisée est s3.

propriétés

Fournit les détails de l'emplacement de la source. Chaque propriété est spécifiée avec une chaîne.

- `s3-bucket`- Obligatoire. Spécifie le nom du compartiment Amazon S3 dans lequel les fichiers sont stockés.
- `s3-key-prefix`- Obligatoire. Spécifie le nom du dossier dans le compartiment Amazon S3 où les fichiers sont stockés.

Présentation de la section consacrée aux définitions

Spécifie les définitions des ressources des services, des paramètres, des données et des autres ressources typiques dont l'application a besoin pour s'exécuter. Lorsque vous mettez à jour une définition d'application, AWS Mainframe Modernization détecte les modifications en comparant les `definition` listes `source-locations` et des versions précédentes et actuelles du fichier JSON de définition d'application.

La section de définition est spécifique au moteur et peut être modifiée. Les sections suivantes présentent des exemples de définitions d'applications spécifiques au moteur pour les deux moteurs.

AWS Exemple de définition d'application Blu Age

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 8194,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/murachs-v6/"
        },
        "blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
```

```
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
        }
    }
}
```

AWS Détails de la définition de Blu Age

Écouteur (s) - obligatoire

Spécifiez le port que vous utiliserez pour accéder à l'application par le biais de l'Elastic Load Balancing AWS Mainframe Modernization créé. Utilisez la structure suivante :

```
"listeners": [{
            "port": 8194,
            "type": "http"
}],
```

port

(Obligatoire) Vous pouvez utiliser n'importe quel port disponible, à l'exception des ports connus compris entre 0 et 1023. Nous recommandons d'utiliser la plage de 8192 à 8199. Assurez-vous qu'aucun autre écouteur ou application ne fonctionne sur ce port.

type

(Obligatoire) Actuellement, seul `http` est pris en charge.

AWS Application Blu Age - obligatoire

Spécifiez l'emplacement où le moteur récupère le fichier image de l'application à l'aide de la structure suivante.

```
"ba-application": {
            "app-location": "${s3-source}/murachs-v6/",
            "files-directory": "/m2/mount/myfolder",
```

```
            "enable-jics": <true|false>,
            "enable-batch-restart": <true|false>,
            "shared-app-location": "${s3-source}/shared/"
},
```

emplacement de l'application

Emplacement spécifique dans Amazon S3 où le fichier image de l'application est stocké.

répertoire-fichiers

(Facultatif) Emplacement des fichiers d'entrée/sortie pour les lots. Il doit s'agir d'un sous-dossier de la configuration du point de FSx montage Amazon EFS ou Amazon au niveau de l'environnement. Le sous-dossier doit appartenir à un utilisateur approprié pour être utilisé par l'application Blu Age exécutée à l'intérieur AWS Mainframe Modernization. Pour ce faire, lorsque vous attachez le lecteur à une EC2 instance Amazon Linux, un groupe avec un ID `101` et un utilisateur avec un ID `3001` doivent être créés, et le dossier souhaité doit appartenir à cet utilisateur. Par exemple, le *`testclient`* dossier peut être utilisé par Blu Age AWS Mainframe Modernization Managed.

```
groupadd -g 101 mygroup
useradd -M -g mygroup -p mypassword -u 3001 myuser
mkdir testclient
chown myuser:mygroup testclient
```

activer-jics

(Facultatif) Spécifie s'il faut activer JICS. La valeur par défaut est true (vrai). La définition de ce paramètre sur false empêche la création de la base de données JICS.

enable-batch-restart

(Facultatif) Spécifie s'il faut activer la fonctionnalité de redémarrage pour les tâches par lots. La valeur par défaut est false. Pour plus d'informations sur les configurations de redémarrage par lots, voir Propriétés du moteur AWS Blu Age préfixées `jcl.checkpoint` dans Propriétés de configuration de l'application gérée avec le moteur AWS Blu Age.

shared-app-location

(Facultatif) Emplacement supplémentaire dans Amazon S3 où les éléments d'application partagés sont stockés. Il peut contenir le même type de structure d'application que app-location.

BluSam - facultatif

Spécifiez la base de données BluSam et le cache Redis à l'aide de la structure suivante.

```
"blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
}
```

db

Spécifie les propriétés de la base de données utilisée avec l'application. La base de données doit être une base de données Aurora PostgreSQL. Vous pouvez définir les propriétés suivantes :

- `nb-threads`- (Facultatif) Spécifie le nombre de threads dédiés utilisés pour le mécanisme d'écriture secondaire sur lequel repose le moteur BluSam. La valeur par défaut est de 8.
- `batch-size`- (Facultatif) Spécifie le seuil utilisé par le mécanisme d'écriture différée pour démarrer les opérations de stockage par lots. Le seuil représente le nombre d'enregistrements modifiés qui démarreront une opération de stockage par lots afin de garantir la persistance des enregistrements modifiés. Le déclencheur lui-même est basé sur la combinaison de la taille du lot et d'un temps écoulé d'une seconde, selon la première valeur atteinte. La valeur par défaut est 10 000.
- `name`- (Facultatif) Spécifie le nom de la base de données.
- `secret-manager-arn`- Spécifie le nom de ressource Amazon (ARN) du secret qui contient les informations d'identification de la base de données. Pour de plus amples informations, veuillez consulter Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données.

Redis

Spécifie les propriétés du cache Redis que l'application utilise pour stocker les données temporaires dont elle a besoin dans un emplacement central afin d'améliorer les performances. Nous vous recommandons de chiffrer et de protéger le cache Redis par mot de passe.

- `hostname`- Spécifie l'emplacement du cache Redis.
- `port`- Spécifie le port, généralement 6379, où le cache Redis envoie et reçoit les communications.
- `useSsl`- Spécifie si le cache Redis est crypté. Si le cache n'est pas chiffré, `useSsl` définissez-le sur false.
- `secret-manager-arn`- Spécifie le nom de ressource Amazon (ARN) du secret qui contient le mot de passe du cache Redis. Si le cache Redis n'est pas protégé par mot de passe, ne le spécifiez pas. `secret-manager-arn` Pour de plus amples informations, veuillez consulter Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données.

AWS files d'attente de messages Blu Age - facultatif

Spécifiez les détails de connexion JMS-MQ pour l'application AWS Blu Age.

```
"message-queues": [
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr1",
      "channel": "mqChannel1",
      "hostname": "mqserver-host1",
      "port": 1414,
      "user-id": "app-user1",
      "ssl-cipher": "*TLS12ORHIGHER",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  },
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr2",
      "channel": "mqChannel2",
      "hostname": "mqserver-host2",
      "port": 1412,
      "user-id": "app-user2",
      "ssl-cipher": "*TLS12ORHIGHER",
```

```
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  }
]
```

type de produit

(Obligatoire) Spécifie le type de produit. Actuellement, il ne peut s'agir que de « JMS-MQ » pour les applications AWS Blu Age.

gestionnaire de files d'attente

(Obligatoire) Spécifie le nom du gestionnaire de files d'attente.

channel

(Obligatoire) Spécifie le nom du canal de connexion au serveur.

hostname

(Obligatoire) Spécifie le nom d'hôte du serveur de file d'attente de messages.

port

(Obligatoire) Spécifie le numéro de port du récepteur sur lequel le serveur écoute.

identifiant d'utilisateur

(Facultatif) Spécifie l'ID de compte utilisateur autorisé à effectuer des opérations de file de messages sur le canal spécifié.

chiffrement SSL

(Facultatif) Spécifie la spécification du chiffrement SSL pour la connexion.

secret-manager-arn

(Facultatif) Spécifie le nom de ressource Amazon (ARN) de Secrets Manager qui fournit le mot de passe de l'utilisateur spécifié.

AWS Configuration EFS du stockage des applications Blu Age (en option)

Spécifiez les détails du point d'accès EFS pour le stockage des applications à l'aide de la structure suivante.

```
"ba-application": {
            "file-permission-mask": "UMASK002"
```

```
},
"efs-configs": [
           {
            "file-system-id": "fs-01376dfsvfvrsvsr",
            "mount-point": "/m2/mount/efs-ap2",
            "access-point-id": fsap-0eaesefvrefrewgv8"
           }
]
```

file-system-id

(Obligatoire) ID du système de fichiers EFS auquel s'applique le point d'accès. Modèle : « fs- ([0-9a-f] {8,40}) {1,128} $ »

point de montage

(Obligatoire) Point de montage du système de fichiers au niveau de l'application. Il doit être différent du point de montage du stockage au niveau de l'environnement.

access-point-id

(Obligatoire) L'ID du point d'accès, attribué par Amazon EFS. Modèle : « ^fsap- ([0-9a-f] {8,40}) {1,128} $ »

file-permission-mask

(Facultatif) Définit le masque de création de fichiers pour les fichiers créés par le processus de candidature. Par exemple, lorsque la valeur est définie sur`UMASK006`, tous les fichiers auront l'autorisation 660. Cela signifie que seuls le propriétaire du fichier et le groupe de fichiers auront l'accès en lecture et en écriture, tandis que les autres utilisateurs n'auront aucune autorisation.

> **Note**
>
> La valeur définie pour ce champ n'est prise en compte que lors de l'utilisation du stockage EFS au niveau de l'application.

> **Note**
>
> Lorsque la configuration efs est fournie, le répertoire des fichiers doit être spécifié dans la section de définition de l'application. Il doit s'agir d'un sous-dossier du point de montage Amazon EFS configuré au niveau de l'application.

Définition de l'application Rocket Software (anciennement Micro Focus)

L'exemple de section de définition suivant concerne le moteur d'exécution Rocket Software et contient des éléments obligatoires et facultatifs.

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 5101,
            "type": "tn3270"
        }],
        "dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }]
        },
        "cognito-auth-handler": {
            "user-pool-id": "cognito-idp.us-west-2.amazonaws.com/us-west-2_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "us-west-2:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
        },
        "ldap-ad-auth-handler": {
            "ldap-ad-connection-secrets": [LIST OF AD-SECRETS]
        },
        "batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl",
            "program-path": "/m2/mount/libs/loadlib:$EFS_MOUNT/emergency/loadlib",
```

```
            "system-procedure-libraries":"SYS1.PROCLIB;SYS2.PROCLIB",
            "aliases": [
               {"alias": "FDSSORT", "program": "SORT"},
               {"alias": "MFADRDSU", "program": "ADRDSSU"}
            ]
        },
        "cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
        },
        "jes-printers": [
           {
            "name": "printerName",
             "classes": [
                "A",
                "B"
             ],
             "description": "printer desc....",
             "exit-module": {
                 "name": "lrsprte6"
                }
             }
         ],
        "xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "xa-connection-type": "postgres",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
        }],
        "runtime-settings": {
          "base-configuration-location": "${s3-source}/exported.json",
          "environment-variables": {
            "ES_JES_RESTART": "N",
            "EFS_MOUNT": "/m2/mount/efs",
            "LRSQ_ADDRESS": "<lrsq-address>"
          }
        }
    }
}
```

Détails de la définition du logiciel Rocket

Le contenu de la section de définition du fichier de définition de l'application Rocket Software varie en fonction des ressources dont votre application mainframe migrée a besoin au moment de l'exécution.

Écouteur (s) - obligatoire

Spécifiez un écouteur en utilisant la structure suivante :

```
"listeners": [{
    "port": 5101,
    "type": "tn3270"
}],
```

port

Pour tn3270, la valeur par défaut est 5101. Pour les autres types d'auditeurs de service, le port varie. Vous pouvez utiliser n'importe quel port disponible, à l'exception des ports connus compris entre 0 et 1023. Chaque écouteur doit avoir un port distinct. Les écouteurs ne doivent pas partager de ports. Pour plus d'informations, voir Listener Control dans la documentation de Micro Focus Enterprise Server.

type

Spécifie le type d'écouteur de service. Pour plus d'informations, consultez la section Listeners dans la documentation de Micro Focus Enterprise Server.

Emplacement des ensembles de données : obligatoire

Spécifiez l'emplacement de l'ensemble de données à l'aide de la structure suivante.

```
"dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }],
 }
```

emplacements de base de données

Spécifie l'emplacement des ensembles de données créés par l'application migrée. Actuellement, ne AWS Mainframe Modernization prend en charge que les ensembles de données provenant d'une seule base de données VSAM.

- `name`- Spécifie le nom de l'instance de base de données qui contient les ensembles de données créés par l'application migrée.
- `secret-manager-arn`- Spécifie le nom de ressource Amazon (ARN) du secret qui contient les informations d'identification de la base de données.

Gestionnaire d'authentification et d'autorisation Amazon Cognito : facultatif

AWS Mainframe Modernization utilise Amazon Cognito pour l'authentification et l'autorisation des applications migrées. Spécifiez le gestionnaire d'authentification Amazon Cognito à l'aide de la structure suivante.

```
"cognito-auth-handler": {
            "user-pool-id": "cognito-idp.Region.amazonaws.com/Region_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "Region:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
}
```

user-pool-id

Spécifie le groupe d'utilisateurs Amazon Cognito AWS Mainframe Modernization utilisé pour authentifier les utilisateurs de l'application migrée. La Région AWS valeur pour le groupe d'utilisateurs doit correspondre à celle Région AWS pour l' AWS Mainframe Modernization application.

identificateur du client

Spécifie l'application migrée à laquelle l'utilisateur authentifié peut accéder.

identity-pool-id

Spécifie le groupe d'identités Amazon Cognito auquel l'utilisateur authentifié échange un jeton de groupe d'utilisateurs contre des informations d'identification lui permettant d'accéder. AWS Mainframe Modernization Le Région AWS nom du pool d'identités doit correspondre à Région AWS celui de l' AWS Mainframe Modernization application.

Gestionnaire LDAP et Active Directory (facultatif)

Vous pouvez intégrer votre application à Active Directory (AD) ou à n'importe quel type de serveur LDAP pour permettre aux utilisateurs de l'application d'utiliser leurs informations d'identification LDAP/AD pour l'autorisation et l'authentification.

Pour intégrer votre application à AD

1. Suivez les étapes décrites dans Configuration d'Active Directory pour la sécurité des serveurs d'entreprise dans la documentation de Micro Focus Enterprise Server.
2. Créez un AWS Secrets Manager secret avec votre AD/LDAP details for each AD/LDAP serveur que vous souhaitez utiliser avec votre application. Pour plus d'informations sur la création d'un secret, consultez la section Créer un secret AWS Secrets Manager dans le Guide de AWS Secrets Manager l'utilisateur. Pour le type de secret, choisissez Autre type de secret et incluez les paires clé-valeur suivantes.

```
{
    "connectionPath"       : "<HOST-ADDRESS>:<PORT>",
    "authorizedId"         : "<USER-FULL-DN>",
    "password"             : "<PASSWORD>",
    "baseDn"               : "<BASE-FULL-DN>",
    "userClassDn"          : "<USER-TYPE>",
    "userContainerDn"      : "<USER-CONTAINER-DN>",
    "groupContainerDn"     : "<GROUP-CONTAINER-DN>",
    "resourceContainerDn"  : "<RESOURCE-CONTAINER-DN>"
}
```

⚠ Recommandations en matière de sécurité

- Car`connectionPath`, AWS Mainframe Modernization prend en charge les protocoles LDAP et LDAP over SSL (LDAPS). Nous vous recommandons d'utiliser le protocole LDAPS car il est plus sécurisé et empêche les informations d'identification d'apparaître dans les transmissions réseau.
- Pour `authorizedId` et`password`, nous vous recommandons de spécifier les informations d'identification d'un utilisateur ne disposant que des autorisations de lecture seule et de vérification les plus restrictives requises pour l'exécution de votre application.

- Nous vous recommandons de changer régulièrement les informations d'identification AD/LDAP.
- Ne créez pas d'utilisateurs AD avec le nom d'utilisateur `awsuser` ou`mfuser`. Ces deux noms d'utilisateur sont réservés à l' AWS usage.

Voici un exemple.

```
{
    "connectionPath" : "ldaps://msad4.m2.example.people.aws.dev:636",
    "authorizedId" :
 "CN=LDAPUser,OU=Users,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "password" : "ADPassword",
    "userContainerDn" : "CN=Enterprise Server Users,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "groupContainerDn" : "CN=Enterprise Server Groups,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "resourceContainerDn" : "CN=Enterprise Server Resources,CN=Micro
 Focus,CN=Program Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev"
}
```

Créez le secret à l'aide d'une clé KMS gérée par le client. Vous devez accorder AWS Mainframe Modernization les `DescribeSecret` autorisations `GetSecretValue` et sur le secret `Decrypt` et les `DescribeKey` autorisations sur la clé KMS. Pour plus d'informations, consultez la section Autorisations relatives à la clé KMS dans le guide de AWS Secrets Manager l'utilisateur.

3. Ajoutez ce qui suit à la définition de votre application.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": [LIST OF AD/LDAP SECRETS]
}
```

Voici un exemple.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": ["arn:aws:secrets:1234:us-east-1:secret:123456"]
}
```

Si l'application est intégrée à LDAP et a été démarrée, vous devez fournir des informations d'identification pour exécuter au moins une opération liée à l'application mentionnée dans la liste des autorisations prises en charge.

Le gestionnaire d'authentification LDAP/AD est disponible pour Micro Focus (Rocket) 8.0.11 et versions ultérieures.

> **Note**
>
> Actuellement, l'administrateur LDAP doit fournir des autorisations de « modification » sur l'`casstart`utilitaire dans les ressources du serveur d'entreprise « OPERCMDS » de son répertoire LDAP. Cela doit être fait pour que tous les utilisateurs par défaut requis (par exemple, CICSUSER, si l'application est liée au CICS) puissent démarrer l'application avec succès.

Pour fournir des informations d'identification utilisateur LDAP pour l'authentification et l'autorisation

1. Créez un AWS Secrets Manager avec les clés et valeurs suivantes :

```
{
  "username" : "<USERNAME>",
  "password" : "<PASSWORD>"
}
```

> **Important**
>
> Vous devez avoir le droit d'exécuter `DescribeSecrets` et de `GetSecretValue` contrôler le Secrets Manager utilisé. Associez également une clé KMS et les autorisations nécessaires pour le AWS Secrets Manager, comme indiqué dans Choisir un AWS KMS key.

2. Choisissez le paramètre Secrets Manager.

   AWS console

   Lors de l'exécution des opérations depuis la AWS console, il sera possible de choisir le Gestionnaire de Secrets à transmettre.

AWS CLI (or SDK)

Lors de l'exécution d'opérations à partir de l'AWS CLI (ou du SDK), le paramètre d'API `auth-secrets-manager-arn` doit être transmis avec l'ARN du Secrets Manager.

Voici la liste des opérations d'application qui prennent actuellement en charge l'autorisation :

- `StartBatchJob`
- `CancelBatchJobExecution`
- `ListBatchJobRestartPoints`

Réglages par lots : obligatoire

Spécifiez les détails requis par les tâches par lots exécutées dans le cadre de l'application à l'aide de la structure suivante.

```
"batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl",
            "program-path": "/m2/mount/libs/loadlib:$EFS_MOUNT/emergency/loadlib",
            "system-procedure-libraries":"SYS1.PROCLIB;SYS2.PROCLIB",
            "aliases": [
               {"alias": "FDSSORT", "program": "SORT"},
               {"alias": "MFADRDSU", "program": "ADRDSSU"}
    ]
}
```

initiateurs

Spécifie un initiateur par lots qui démarre lorsque l'application migrée démarre avec succès et continue de fonctionner jusqu'à ce que l'application s'arrête. Vous pouvez définir une ou plusieurs classes par initiateur. Vous pouvez également définir plusieurs initiateurs. Par exemple :

```
"batch-settings": {
            "initiators": [
                {
```

```
                "classes": ["A", "B"],
                "description": "initiator...."
              },
              {
                "classes": ["C", "D"],
                "description": "initiator...."
              }

        ],
}
```

Pour plus d'informations, voir Pour définir un initiateur par lots ou un SEP d'imprimante dans la documentation de Micro Focus Enterprise Server.

- `classes`- Spécifie les classes de travail que l'initiateur peut exécuter. Vous pouvez utiliser jusqu'à 36 caractères. Vous pouvez utiliser les caractères suivants : A-Z ou 0-9.
- `description`- Décrit à quoi sert l'initiateur.

jcl-file-location

Spécifie l'emplacement des fichiers JCL (Job Control Language) requis par les tâches par lots exécutées par l'application migrée.

cheminement du programme

Spécifie le chemin requis pour exécuter des tâches par lots lorsqu'un programme d'une JCL ne se trouve pas à l'emplacement par défaut. Les différents noms de chemin sont séparés par deux points (:).

> **Note**
>
> Le chemin du programme ne peut être qu'un chemin EFS.

system-procedure-libraries

Spécifie les ensembles de données partitionnés par défaut dans lesquels les procédures JCL seront recherchées. La procédure est cependant introuvable dans la JCL ou via les instructions JCLLIB. Ces ensembles de données doivent être catalogués et le nom du catalogue doit être utilisé. Et les entrées sont séparées par un point-virgule (;).

alias

Définit un mappage entre les noms d'utilitaires et de programmes utilisés dans JCL et le nom d'implémentation de l'utilitaire. AWS et les utilitaires de traitement par lots tiers (par exemple M2SFTP, M2WAIT, Syncsort, etc.) peuvent éventuellement avoir des alias pour éviter de devoir modifier la JCL. Par exemple :

- Alias FDSSORT FDSSORT pour SORT et alias FDSICET pour ICETOOL
- Alias ADRDSSU MFADRDSU pour ADRDSSU
- Alias Syncsort DMXMFSRT pour SORT

Réglages CICS - obligatoires

Spécifiez les détails requis pour les transactions CICS exécutées dans le cadre de l'application à l'aide de la structure suivante.

```
"cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
}
```

binary-file-location

Spécifie l'emplacement des fichiers du programme de transaction CICS.

csd-file-location

Spécifie l'emplacement du fichier de définition de ressource CICS (CSD) pour cette application. Pour plus d'informations, consultez les définitions des ressources CICS dans la documentation de Micro Focus Enterprise Server.

system-initialization-table

Spécifie la table d'initialisation du système (SIT) utilisée par l'application migrée. Le nom de la table SIT peut comporter jusqu'à 8 caractères. Vous pouvez utiliser A-Z, 0-9, $, @ et #. Pour plus d'informations, consultez les définitions des ressources CICS dans la documentation de Micro Focus Enterprise Server.

Imprimantes : en option

Spécifiez une imprimante JES en utilisant la structure suivante.

```
"jes-printers": [
    {
        "name": "printerName",
        "classes": [
            "A",
            "B"
        ],
        "description": "printer desc....",
        "exit-module": {
            "name": "lrsprte6",
            "module" : "program"
        }
    }
],
```

> **Note**
>
> Un maximum de 25 imprimantes peuvent être configurées pour une application donnée.

name

(Obligatoire) Spécifie le nom à associer à cette ressource d'imprimante. Les noms doivent être uniques pour chaque imprimante, et une limite de 128 caractères alphanumériques peut être utilisée.

cours

(Obligatoire) Spécifie les classes de sortie applicables à cette ressource d'imprimante. Une limite de 36 caractères alphanumériques peut être utilisée.

description

(Facultatif) Texte descriptif supplémentaire pour l'imprimante.

module de sortie

(Facultatif) Spécifie un module personnalisé pour la sortie de l'impression. Il n'existe aucune valeur par défaut. Si elle n'est pas spécifiée, aucun module de sortie ne sera utilisé. Vous

pouvez utiliser un module de sortie d'impression géré ou fournir le vôtre. Les modules de sortie d'impression gérés sont définis à l'aide d'un nom réservé `lrsprte6` pour la file d'attente LRS ou fournissent le vôtre à l'aide du paramètre du module pour spécifier l'emplacement et le nom.

La structure de `exit-module` comporte deux éléments :

- `name`- (Obligatoire), s'`exit-module`il est utilisé. Nom de l'entrée du module de sortie. Le nom d'entrée du module de sortie est limité à 8 caractères.
- `module`- (Facultatif) L'emplacement S3 du binaire du module de sortie d'impression.

Vous pouvez voir d'autres exemples de définition du module de sortie dans la the section called "Imprimantes" section.

Ressources XA - facultatives

Spécifiez les détails requis pour les ressources XA dont l'application a besoin à l'aide de la structure suivante.

```
"xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "xa-connection-type": "postgres",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
}]
```

> **Note**
>
> La définition de la ressource XA a été mise à jour pour inclure un `xa-connection-type` champ facultatif. S'il n'est pas fourni, le type de connexion est supposé être « postgres ».

name

(Obligatoire) Spécifie le nom de la ressource XA.

secret-manager-arn

(Obligatoire) Spécifie le nom de ressource Amazon (ARN) pour le secret qui contient les informations d'identification pour la connexion à la base de données.

xa-connection-type

(Facultatif) Spécifie le type de connexion à la ressource XA.

module

(Obligatoire) Spécifie l'emplacement du fichier exécutable du module de commutation RM. Pour plus d'informations, consultez la section Planification et conception XARs dans la documentation de Micro Focus Enterprise Server.

Paramètres d'exécution : facultatif

Spécifiez les détails requis pour les paramètres d'exécution afin de gérer les variables d'environnement autorisées à l'aide de la structure suivante.

```
"runtime-settings": {
          "base-configuration-location": "${s3-source}/exported.json",
          "environment-variables": {
            "ES_JES_RESTART": "N",
            "EFS_MOUNT": "/m2/mount/efs"
    }
}
```

base-configuration-location

(Facultatif) Spécifie l'emplacement de l'importation en bloc pour une configuration de serveur Micro Focus. Ce fichier doit être un JSON valide et se trouver dans le même emplacement S3 que l'emplacement des artefacts de l'application défini ci-dessus. Pour exporter la configuration d'une application existante, consultez la section Exporter une région depuis la CESAO dans la documentation du logiciel Rocket.

variables d'environnement

Spécifie les variables d'environnement prises en charge par Micro Focus qui sont appliquées au runtime de cette application.

- `ES_JES_RESTART`Test une variable d'environnement Rocket Software qui permet le traitement du redémarrage de JCL. En option, vous pouvez également l'utiliser `ES_ALLOC_OVERRIDE` comme variable d'environnement Rocket Software.

- `EFS_MOUNT`est une variable d'environnement personnalisée que votre application peut utiliser pour identifier l'emplacement du montage EFS de l'environnement.

Vous pouvez accéder à toutes les variables d'environnement du logiciel Rocket dans le guide Rocket Enterprise Server pour UNIX.

AWS Référence de définition des ensembles de données sur la modernisation du mainframe

Si le traitement de votre application nécessite plusieurs ensembles de données, il est inefficace de les saisir un par un dans la console de modernisation du AWS mainframe. Nous vous recommandons plutôt de créer un fichier JSON pour spécifier chaque ensemble de données. Les différents types d'ensembles de données sont spécifiés différemment dans le JSON, bien que de nombreux paramètres soient communs. Ce document décrit les détails du JSON requis pour importer différents types d'ensembles de données.

> **Note**
>
> Avant d'importer des ensembles de données, vous devez les transférer du mainframe vers AWS. Les ensembles de données doivent être dans un format qui peut être chargé sur le moteur d'exécution sélectionné. Dans de nombreux cas, il peut s'agir d'un fichier séquentiel, mais pour Rocket Software (anciennement Micro Focus) VSAM, il devra être dans son format propriétaire. L'`DFCONV`utilitaire est la méthode suggérée pour convertir le fichier. Spécifiez le nom du compartiment et du dossier dans le fichier JSON de définition de l'ensemble de données.
>
> Pour plus d'informations sur le moteur d'exécution Rocket Software, consultez la section Conversion de fichiers par lots DFCONV dans la documentation de Rocket Software.
>
> Pour plus d'informations sur AWS Blu Age, consultezthe section called “AWS Configuration de Blu Age Runtime”.

Rubriques

- Propriétés communes
- Exemple de format de demande d'ensemble de données pour VSAM
- Exemple de format de demande d'ensemble de données pour la base GDG

- Exemple de format de demande d'ensemble de données pour les générations PS ou GDG
- Exemple de format de demande d'ensemble de données pour PO

Propriétés communes

Plusieurs paramètres sont communs à tous les ensembles de données. Ces paramètres couvrent les domaines suivants :

- Informations sur l'ensemble de données (`datasetName`,`datasetOrg`,`recordLength`,`encoding`)
- Informations sur l'emplacement d'où vous effectuez l'importation, c'est-à-dire l'emplacement source de l'ensemble de données. Il ne s'agit pas de l'emplacement sur le mainframe. Il s'agit du chemin d'accès à l'emplacement Amazon S3 où vous avez chargé le jeu de données (`externalLocation`).
- Informations sur l'emplacement vers lequel vous effectuez l'importation, c'est-à-dire l'emplacement cible de l'ensemble de données. Cet emplacement est soit une base de données, soit un système de fichiers, selon votre moteur d'exécution. (`storageTypeetrelativePath`).
- Informations sur le type de jeu de données (type d'ensemble de données spécifique, format, encodage, etc.).

Chaque définition d'ensemble de données possède la même structure JSON. L'exemple JSON suivant montre tous ces paramètres courants.

```
{
    "dataSet": {
        "storageType": "Database",
        "datasetName": "MFI01V.MFIDEMO.BNKACC",
        "relativePath": "DATA",
        "datasetOrg": {
            "type": {
                type-specific properties
                    ...
            },
        },
    },
}
```

Les propriétés suivantes sont communes à tous les ensembles de données.

storageType

Obligatoire. S'applique à l'emplacement cible. Spécifie si l'ensemble de données est stocké dans une base de données ou un système de fichiers. Les valeurs possibles sont `Database` ou `FileSystem`.

- AWS Moteur d'exécution Blu Age : les systèmes de fichiers ne sont pas pris en charge. Vous devez utiliser une base de données.
- Moteur d'exécution Rocket Software : les bases de données et les systèmes de fichiers sont tous deux pris en charge. Vous pouvez utiliser Amazon Relational Database Service ou Amazon Aurora pour les bases de données, et Amazon Elastic File System ou FSx Amazon for Lustre pour les systèmes de fichiers.

datasetName

(Obligatoire) Spécifie le nom complet de l'ensemble de données tel qu'il apparaît sur le mainframe.

Trajectoire relative

(Obligatoire) S'applique à l'emplacement cible. Spécifie l'emplacement relatif de l'ensemble de données dans la base de données ou le système de fichiers.

Ensemble de données Org

(Obligatoire) Spécifie le type de jeu de données. Les valeurs possibles sont `vsam`, `gdg`, `ps`, `po` ou `unknown`.

- AWS Moteur d'exécution Blu Age : seuls les ensembles de données de type VSAM sont pris en charge.
- Moteur d'exécution Rocket Software : les ensembles de données de type VSAM, GDG, PS, PO ou Unknown sont pris en charge.

> **Note**
>
> Si votre application nécessite des fichiers qui ne sont pas des fichiers de données COBOL mais des fichiers PDF ou d'autres fichiers binaires, vous pouvez les spécifier comme suit :

```
"datasetOrg": {
            "type": PS {
```

```
                "format": U
            },
```

Exemple de format de demande d'ensemble de données pour VSAM

- AWS Moteur d'exécution Blu Age : pris en charge.
- Moteur d'exécution Rocket Software : pris en charge.

Si vous importez des ensembles de données VSAM, spécifiez `vsam` comme. `datasetOrg` Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.VSAM.KSDS",
    "relativePath": "DATA",
    "datasetOrg": {
        "vsam": {

            "encoding": "A",
            "format": "KS",
            "primaryKey": {
                "length": 11,
                "offset": 0
            }
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.VSAM.KSDS.DAT"
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données VSAM.

encoding

(Obligatoire) Spécifie le codage du jeu de caractères du jeu de données. Les valeurs possibles sont ASCII (`A`), EBCDIC () et `E` Unknown (). `?`

format

(Obligatoire) Spécifie le type d'ensemble de données VSAM et le format d'enregistrement.

- AWS Moteur d'exécution Blu Age : les valeurs possibles sont ESDS (`ES`) et KSDS (`KS`). Le format d'enregistrement peut être fixe ou variable.
- Moteur d'exécution Rocket Software : les valeurs possibles sont ESDS (`ES`), KSDS (`KS`) et RRDS (). `RR` La définition VSAM inclut le format d'enregistrement, il n'est donc pas nécessaire de le spécifier séparément.

Clé primaire

(Obligatoire) S'applique uniquement aux ensembles de données VSAM KSDS. Spécifie la clé primaire. Comprend le nom de la clé primaire, le décalage de la clé et la longueur de la clé. Elles `name` `length` sont facultatives `offset` et obligatoires.

Durée de l'enregistrement

(Obligatoire) Spécifie la longueur d'un enregistrement. Pour les formats d'enregistrement de longueur fixe, ces valeurs doivent correspondre.

- AWS Moteur d'exécution Blu Age : pour VSAM ESDS et KSDS, `min` il est facultatif et `max` obligatoire.
- Moteur d'exécution Rocket Software : `min` et `max` sont requis.

Emplacement externe

(Obligatoire) Spécifie l'emplacement source : c'est-à-dire le compartiment Amazon S3 dans lequel vous avez chargé l'ensemble de données.

Propriétés spécifiques au moteur Blu Age

Le moteur d'exécution AWS Blu Age prend en charge la compression des ensembles de données VSAM. L'exemple suivant montre comment vous pouvez spécifier cette propriété au format JSON.

```
{
    common properties
        ...
```

```
        "datasetOrg": {
            "vsam": {
                common properties
                ...
                "compressed": boolean,
                common properties
                ...
            }
        }
}
```

Spécifiez la propriété de compression comme suit :

compression

(Facultatif) Spécifie si les index de cet ensemble de données sont stockés sous forme de valeurs compressées. Si vous disposez d'un ensemble de données volumineux (généralement > 100 Mo), pensez à attribuer à cet indicateur la valeur`true`.

Exemple de format de demande d'ensemble de données pour la base GDG

- AWS Moteur d'exécution Blu Age : non pris en charge.
- Moteur d'exécution Rocket Software : pris en charge.

Si vous importez des ensembles de données de base GDG, spécifiez `gdg` comme. `datasetOrg` Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.GDG",
    "relativePath": "DATA",
    "datasetOrg": {
        "gdg": {
            "limit": "3",
            "rollDisposition": "Scratch and No Empty"
        }
    }
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données de base GDG.

limite

(Obligatoire) Spécifie le nombre de générations actives, ou de biais. Pour un cluster de base GDG, le maximum est de 255.

Disposition des rouleaux

(Facultatif) Spécifie comment gérer les ensembles de données de génération lorsque le maximum est atteint ou dépassé. Les valeurs possibles sont `No Scratch and No Empty`, `Scratch and No Empty`, `Scratch and Empty` ou `No Scratch and Empty`. L'argument par défaut est `Scratch and No Empty`.

Exemple de format de demande d'ensemble de données pour les générations PS ou GDG

- AWS Moteur d'exécution Blu Age : non pris en charge.
- Moteur d'exécution Rocket Software : pris en charge.

Si vous importez des ensembles de données de générations PS ou GDG, spécifiez `ps` comme. `datasetOrg` Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PS.FB",
    "relativePath": "DATA",
    "datasetOrg": {
        "ps": {
            "format": "FB",
            "encoding": "A"
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.PS.LSEQ"
}
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données des générations PS ou GDG.

format

(Obligatoire) Spécifie le format des enregistrements de l'ensemble de données. Les valeurs possibles sont `FFA`,`FB`,`FBA`,`FBM`,`FBS`,`FM`,`FS`,`LSEQ`,`U`,`V`,`VA`,`VB`,`VBA`,`VBM`,`VBS`,`VM`, et`VS`.

encoding

(Obligatoire) Spécifie le codage du jeu de caractères du jeu de données. Les valeurs possibles sont ASCII (`A`), EBCDIC () et `E` Unknown () ?

Durée de l'enregistrement

(Obligatoire) Spécifie la longueur d'un enregistrement. Vous devez spécifier à la fois la longueur minimale (`min`) et la longueur maximale (`max`) de l'enregistrement. Pour les formats d'enregistrement de longueur fixe, ces valeurs doivent correspondre.

Emplacement externe

(Obligatoire) Spécifie l'emplacement source : c'est-à-dire le compartiment Amazon S3 dans lequel vous avez chargé l'ensemble de données.

Exemple de format de demande d'ensemble de données pour PO

Si vous importez des ensembles de données de bons de commande, spécifiez po comme`datasetOrg`. Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PO.PROC",
    "relativePath": "DATA",
    "datasetOrg": {
        "po": {
            "format": "LSEQ",
            "encoding": "A",
            "memberFileExtensions": ["PRC"]
        }
    },
    "recordLength": {
        "min": 80,
        "max": 80
```

```
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/source/proc/"
}
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données PO.

format

(Obligatoire) Spécifie le format des enregistrements de l'ensemble de données. Les valeurs possibles sont `FFA`,`FB`,`FBA`,`FBM`,`FBS`,`FM`,`FS`,`LSEQ`,`U`,`V`,`VA`,`VB`,`VBA`,`VBM`,`VBS`,`VM`, et`VS`.

encoding

(Obligatoire) Spécifie le codage du jeu de caractères du jeu de données. Les valeurs possibles sont ASCII (`A`), EBCDIC () et `E` Unknown (). `?`

memberFileExtensions

(Obligatoire) Spécifie un tableau contenant une ou plusieurs extensions de nom de fichier, ce qui vous permet de spécifier les fichiers à inclure en tant que membre PDS.

Durée de l'enregistrement

(Facultatif) Spécifie la longueur d'un enregistrement. La longueur minimale (`min`) et maximale (`max`) de l'enregistrement est facultative. Pour les formats d'enregistrement de longueur fixe, ces valeurs doivent correspondre.

Emplacement externe

(Obligatoire) Spécifie l'emplacement source : c'est-à-dire le compartiment Amazon S3 dans lequel vous avez chargé l'ensemble de données.

> **Note**
>
> L'implémentation actuelle du moteur d'exécution Rocket Software ajoute des entrées PDS sous forme d'ensembles de données dynamiques.

Environnements d'exécution gérés dans le cadre de la modernisation des AWS mainframes

Si vous débutez dans le domaine de la modernisation des AWS mainframes, consultez les rubriques suivantes pour commencer :

- Qu'est-ce que la modernisation AWS du mainframe ?
- Configuration pour la modernisation du AWS mainframe
- Commencez à moderniser votre AWS mainframe
- Tutoriel : Configuration d'un environnement d'exécution géré pour AWS Blu Age
- Tutoriel : Configuration du runtime géré pour Rocket Software (anciennement Micro Focus)

Dans AWS Mainframe Modernization, un environnement d'exécution est une combinaison nommée de ressources de AWS calcul, d'un moteur d'exécution et des détails de configuration que vous spécifiez. L'environnement d'exécution héberge une ou plusieurs applications. Dans le cadre de la modernisation AWS du mainframe, les applications contiennent des charges de travail du mainframe migrées. Vous pouvez choisir le moteur d'exécution pour les environnements que vous créez. Choisissez AWS Blu Age si vous utilisez le modèle de refactorisation automatique, et Rocket Software (anciennement Micro Focus) si vous utilisez le modèle de replateforme. Vous pouvez également choisir la quantité de ressources de calcul adaptée à votre application et éventuellement associer du stockage aux environnements d'exécution. AWS La modernisation du mainframe permet à Amazon d'utiliser les CloudWatch métriques et la journalisation pour vous permettre de surveiller votre environnement d'exécution.

Rubriques

- Création d'un environnement d'exécution pour la modernisation du AWS mainframe
- Mettre à jour un environnement d'exécution de modernisation du AWS mainframe
- Arrêter un environnement d'exécution de modernisation du AWS mainframe
- Redémarrer un environnement d'exécution de modernisation du AWS mainframe
- Supprimer un environnement d'exécution de modernisation du AWS mainframe

Création d'un environnement d'exécution pour la modernisation du AWS mainframe

Utilisez la console de modernisation du AWS mainframe pour créer un environnement de modernisation AWS du mainframe.

Ces instructions supposent que vous avez effectué les étapes décrites dansConfiguration pour la modernisation du AWS mainframe.

Création d'un environnement d'exécution

Pour créer un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'environnement.
3. Sur la page Environnements, choisissez Create environment.
4. Sur la page Spécifier les informations de base, fournissez les informations suivantes :
    a. Dans la section Nom et description, entrez le nom de l'environnement.
    b. (Facultatif) Dans le champ Description de l'environnement, entrez une description de l'environnement. Cette description peut vous aider, ainsi que les autres utilisateurs, à identifier l'objectif de l'environnement d'exécution.
    c. Dans la section des options du moteur, choisissez Blu Age pour le refactoring automatique ou Micro Focus (Rocket) pour le replatforming.
    d. Choisissez une version pour le moteur que vous avez sélectionné.
    e. (Facultatif) Dans la section Balises, choisissez Ajouter une nouvelle balise pour ajouter une ou plusieurs balises d'environnement à votre environnement. Une balise d'environnement est une étiquette d'attribut personnalisée qui vous aide à organiser et à gérer vos AWS ressources.
    f. Choisissez Next (Suivant).
5. Sur la page Spécifier les configurations, fournissez les informations suivantes :
    a. Dans la section Disponibilité, choisissez Environnement d'exécution autonome ou Cluster à haute disponibilité.

Le modèle de disponibilité détermine le niveau de disponibilité de votre application lors de son exécution. Le mode autonome convient parfaitement à des fins de développement. La haute disponibilité concerne les applications qui doivent être disponibles à tout moment.

b. Dans Ressources, choisissez un type d'instance et la capacité souhaitée.

Ces ressources sont les EC2 instances Amazon gérées par AWS Mainframe Modernization qui hébergeront votre environnement d'exécution. Les environnements d'exécution autonomes offrent deux choix de type d'instance et n'autorisent qu'une seule instance. Les environnements d'exécution à haute disponibilité offrent deux choix de type d'instance et autorisent jusqu'à deux instances.

Pour plus d'informations, consultez Amazon EC2 Instance Types et contactez un spécialiste des AWS mainframes pour obtenir des conseils.

6. Dans la section Sécurité et réseau, procédez comme suit :

a. Si vous souhaitez que les applications soient accessibles au public, choisissez Autoriser les applications déployées dans cet environnement à être accessibles au public.

b. Choisissez le type de réseau. Si vous le souhaitez IPv4, les applications de l'environnement AWS Mainframe Modernization répondent uniquement IPv4 aux demandes. En mode double pile, les applications répondront à la fois aux IPv6 demandes IPv4 et aux demandes. Si vous choisissez le mode double pile, assurez-vous qu'il existe au moins un VPC IPv6 avec des sous-réseaux activés.

c. Choisissez un Virtual Private Cloud (VPC).

d. Si vous utilisez le modèle de haute disponibilité, choisissez deux sous-réseaux ou plus. Si vous utilisez le modèle autonome avec le moteur AWS Blu Age, choisissez deux sous-réseaux ou plus. Si vous utilisez le modèle autonome avec le moteur Rocket Software, vous pouvez spécifier un sous-réseau.

e. Choisissez un groupe de sécurité pour le VPC que vous avez sélectionné.

> **Note**
>
> AWS La modernisation du mainframe crée un Network Load Balancer qui vous permet de distribuer les connexions à votre environnement d'exécution. Assurez-vous que les règles entrantes et sortantes de votre groupe de sécurité autorisent l'accès depuis une adresse IP au port que vous avez spécifié dans la `the section called “Écouteur (s) - obligatoire”` propriété de la définition de

l'application. Pour plus d'informations, voir Mettre à jour les groupes de sécurité pour votre Network Load Balancer dans le Guide de l'utilisateur pour les Network Load Balancers.

f. Dans le champ Clé KMS, choisissez Personnaliser les paramètres de chiffrement si vous souhaitez utiliser un système géré par le client AWS KMS key. Pour de plus amples informations, veuillez consulter Chiffrement des données arrêté pour le service de modernisation des AWS mainframes.

> **Note**
>
> Par défaut, AWS Mainframe Modernization chiffre vos données avec des données AWS KMS key que AWS Mainframe Modernization possède et gère pour vous. Cependant, vous pouvez choisir d'utiliser un service géré par le client AWS KMS key.

g. (Facultatif) Choisissez un nom AWS KMS key d'utilisateur ou un Amazon Resource Name (ARN). Vous pouvez également choisir Create an AWS KMS key pour accéder à la AWS KMS console et en créer un nouveau AWS KMS key.

h. Choisissez Next (Suivant).

7. (Facultatif) Sur la page Joindre un stockage, choisissez un ou plusieurs systèmes de FSx fichiers Amazon EFS ou Amazon.

   Le système de fichiers monté sur un environnement de modernisation du AWS mainframe doit appartenir à un utilisateur approprié pour être utilisé par vos applications exécutées dans la console AWS Mainframe Modernization.

   Pour configurer ces paramètres utilisateur, vous pouvez associer le lecteur à une EC2 instance Amazon Linux. Créez ensuite un groupe avec ID `101` et un utilisateur avec ID`3001`. Assurez-vous également que le dossier de données souhaité qui sera utilisé par vos applications doit appartenir à cet utilisateur.

   Par exemple, le `myFiles` dossier peut être utilisé par vos applications de modernisation du AWS mainframe exécutées dans AWS Mainframe Modernization Managed.

```
groupadd -g 101 mygroup
useradd -M -g mygroup -p mypassword -u 3001 myuser
mkdir myFiles
```

```
chown myuser:mygroup myFiles
```

> **Note**
>
> Pour permettre l'accès au système de fichiers, les règles de groupes de sécurité suivantes doivent être configurées pour établir la connectivité réseau entre l'EFS et l'instance d'environnement M2 :
>
> - Groupe de sécurité de l'environnement M2 — Incluez une règle sortante qui autorise le trafic sur le port NFS 2049.
> - Le montage du système de fichiers cible le groupe de sécurité : incluez une règle entrante qui autorise le trafic sur le port NFS 2049 à partir du groupe de sécurité d'instance (répertorié ci-dessus) et une règle sortante qui autorise le trafic sur le port NFS 2049.

8. Choisissez Next (Suivant).
9. Dans la section Fenêtre de maintenance, choisissez le moment où vous souhaitez appliquer les modifications en attente à l'environnement.

   - Si vous choisissez Aucune préférence, AWS Mainframe Modernization choisit une fenêtre de maintenance optimisée pour vous.
   - Pour spécifier une fenêtre de maintenance particulière, choisissez Sélectionner une nouvelle fenêtre de maintenance. Choisissez ensuite un jour de la semaine, une heure de début et une durée pour la fenêtre de maintenance.

   Pour plus d'informations sur la fenêtre de maintenance, consultezAWS Fenêtre de maintenance de la modernisation du mainframe.

   Choisissez Next (Suivant).
10. Sur la page Réviser et créer, passez en revue les informations que vous avez saisies, puis choisissez Créer un environnement.

Mettre à jour un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour mettre à jour un environnement d'exécution AWS Mainframe Modernization. Vous pouvez mettre à jour la version secondaire du moteur d'exécution ou le type d'instance qui héberge l'environnement d'exécution. Vous pouvez choisir d'appliquer les mises à jour immédiatement ou pendant la période de maintenance préférée.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Mettre à jour un environnement d'exécution

Pour mettre à jour un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez mettre à jour a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez mettre à jour.
4. Sur la page de détails de l'environnement, choisissez Actions, puis Modifier l'environnement.
5. Effectuez une ou plusieurs des modifications suivantes :

   - Dans la section Options du moteur, choisissez la version du moteur que vous souhaitez.
   - Dans la section Ressources, choisissez le type d'instance que vous souhaitez.
   - Dans la section Fenêtre de maintenance, choisissez le jour, l'heure et la durée souhaités.

   > **Note**
   >
   > Les seules modifications que vous pouvez choisir d'appliquer pendant la période de maintenance concernent la version du moteur. Vous devez appliquer immédiatement toutes les autres modifications.

6. Choisissez Next (Suivant).
7. Dans Quand appliquer ces modifications, choisissez Immédiatement ou Pendant la fenêtre de maintenance suivante. Choisissez ensuite l'environnement de mise à jour.

Si vous choisissez Immédiatement, un message s'affiche lorsque la mise à jour de l'environnement est terminée.

AWS Fenêtre de maintenance de la modernisation du mainframe

Chaque environnement d'exécution dispose d'une fenêtre de maintenance hebdomadaire de deux heures. Toutes les modifications du système sont appliquées pendant cette période. La fenêtre de maintenance vous permet de contrôler les modifications, les logiciels et les correctifs de sécurité. Si un événement de maintenance est prévu pour une semaine donnée, il commence pendant cette fenêtre de maintenance de deux heures. La plupart des événements de maintenance se terminent également pendant la fenêtre de maintenance de deux heures, bien que les événements de maintenance plus importants puissent prendre plus de deux heures.

La fenêtre de maintenance de deux heures est sélectionnée au hasard parmi une tranche de 8 heures par région. Si vous ne spécifiez pas de fenêtre de maintenance lorsque vous créez un environnement d'exécution, AWS Mainframe Modernization attribue une fenêtre de maintenance de 2 heures un jour de la semaine sélectionné au hasard.

AWS La modernisation du mainframe consomme certaines des ressources de votre instance d'environnement pendant la maintenance. Il est possible que vous observiez un effet minime sur les performances ou des interruptions dans les applications pendant la maintenance.

Arrêter un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour arrêter un environnement d'exécution AWS Mainframe Modernization. Lorsque vous arrêtez un environnement, les déploiements d'applications en cours sont conservés et vous ne serez pas facturé pour l'environnement tant que celui-ci n'est pas redémarré.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Arrêter un environnement d'exécution

Si vous devez arrêter un environnement d'exécution AWS Mainframe Modernization, vous devez suivre des étapes similaires à celles décrites dans la section relative à l'environnement de mise à jour.

Utilisez la console AWS Mainframe Modernization pour arrêter un environnement d'exécution AWS Mainframe Modernization. Lorsque vous arrêtez un environnement, les déploiements d'applications en cours sont conservés et vous ne serez pas facturé pour l'environnement tant que celui-ci n'est pas redémarré.

> **Note**
>
> Vous devez arrêter toutes les applications avant d'arrêter l'environnement.

Pour arrêter un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez arrêter a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez arrêter.
4. Sur la page de détails de l'environnement, choisissez Actions, puis Modifier l'environnement.
5. Sur la page Modifier l'environnement, recherchez la section Ressources et mettez à jour la capacité souhaitée à zéro.

   > **Note**
   >
   > Pour arrêter un environnement, vous pouvez uniquement choisir de l'arrêter immédiatement.

6. Choisissez Next (Suivant).
7. Dans Quand appliquer ces modifications, sélectionnez Immédiatement. Choisissez ensuite l'environnement de mise à jour.

Un message s'affiche lorsque la capacité de l'environnement est mise à jour.

Redémarrer un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour redémarrer un environnement d'exécution AWS Mainframe Modernization. Lorsque vous redémarrez un environnement d'exécution, la facturation de l'environnement reprend.

Redémarrer un environnement d'exécution

Pour redémarrer un environnement d'exécution AWS Mainframe Modernization, vous devez suivre des étapes similaires à celles décrites dans la section relative à l'environnement d'arrêt.

Pour redémarrer un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez redémarrer a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez redémarrer.
4. Sur la page de détails de l'environnement, choisissez Actions, puis Modifier l'environnement.

   > **Note**
   >
   > La capacité souhaitée pour un environnement autonome ne peut être mise à jour qu'à 1. Pour redémarrer un environnement d'exécution, vous pouvez uniquement choisir de le redémarrer immédiatement.

5. Sur la page Modifier l'environnement, recherchez la section Ressources et mettez à jour la capacité souhaitée de zéro à la capacité requise.
6. Choisissez Next (Suivant).
7. Dans Quand appliquer ces modifications, sélectionnez Immédiatement. Choisissez ensuite l'environnement de mise à jour.

Un message s'affiche lorsque la capacité de l'environnement est mise à jour et que l'environnement est redémarré.

Supprimer un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour supprimer un environnement d'exécution AWS Mainframe Modernization.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration pour la modernisation du AWS mainframe.

Supprimer un environnement d'exécution

Si vous devez supprimer un environnement d'exécution AWS Mainframe Modernization, assurez-vous d'abord de supprimer toutes les applications déployées de l'environnement. Vous ne pouvez pas supprimer un environnement d'exécution dans lequel des applications sont déployées.

Pour supprimer un d'environnement

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez supprimer a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez supprimer, puis sélectionnez Actions et Supprimer l'environnement.
4. Dans la fenêtre Supprimer l'environnement, entrez `delete` pour confirmer que vous souhaitez supprimer l'environnement d'exécution, puis choisissez Supprimer.

Tests d'applications dans le cadre de la modernisation des AWS mainframes

AWS Les tests d'applications de modernisation du mainframe fournissent des tests d'équivalence fonctionnelle automatisés pour vos projets de migration. AWS Les tests d'applications de modernisation du mainframe accélèrent les projets de migration en tirant parti de l'élasticité du cloud. Vous pouvez exécuter des suites de tests indépendantes sur autant d'environnements parallèles que nécessaire, réduisant ainsi les délais de test. Les principaux avantages des tests d'applications incluent l'accélération et l'agilité des tests, des degrés élevés de répétabilité des tests, une évolutivité et une élasticité intégrées, une automatisation à grande échelle, une rentabilité et une intégration parfaite AWS CloudFormation pour créer des environnements de test cibles.

Rubriques

- Qu'est-ce que le test des applications de modernisation des AWS mainframes ?
- AWS Concepts de test des applications de modernisation du mainframe
- AWS Conditions préalables aux tests des applications de modernisation du mainframe
- Workflows de la console de test des applications
- Tutoriel : Configuration de l' CardDemo exemple d'application dans le cadre des tests d'applications de modernisation AWS du mainframe
- Tutoriel : Rejouer et comparer dans le cadre des tests d'applications de modernisation AWS du mainframe à l'aide CardDemo de AWS Blu Age déployée sur Amazon EC2
- AWS Modernisation du mainframe, tests d'applications, ensembles de données pris en charge, pages de code
- Protection des données dans le cadre des tests d'applications de modernisation des AWS mainframes
- Comment fonctionnent les tests d'applications de modernisation des AWS mainframes avec IAM

Qu'est-ce que le test des applications de modernisation des AWS mainframes ?

Les tests ont un impact significatif sur les projets de modernisation. AWS Les tests d'applications, une fonctionnalité de la modernisation du AWS mainframe, fournissent des tests d'équivalence

fonctionnelle automatisés pour vos applications migrées. Les tests d'équivalence fonctionnelle vous aident à valider que vos applications sur le AWS Cloud sont équivalentes à celles de votre mainframe. AWS Les tests d'applications comparent automatiquement les modifications apportées aux ensembles de données, aux enregistrements de base de données et aux écrans 3270 en ligne entre votre ordinateur central et. AWS En outre, les tests d'applications permettent des tests reproductibles, ce qui vous permet d'exécuter vos scénarios de test de nombreuses fois au fur et à mesure que vous mettez à jour l'architecture cible, que vous résolvez des problèmes et que vous progressez vers une application entièrement migrée. Après la migration, vous pouvez continuer à utiliser les tests d'applications pour les tests de régression, afin de vous assurer que les mises à jour des moteurs d'exécution ou d'autres composants n'entraînent pas de régressions. Les tests d'applications sont rentables : les environnements de test cibles sont créés à l'aide des CloudFormation modèles fournis par l'utilisateur, en s'appuyant sur les concepts Infrastructure-as-Code (IaC). Les tests d'applications accélèrent les projets de migration en utilisant l'élasticité du cloud. Vous pouvez exécuter des suites de tests indépendantes sur autant d'environnements parallèles que nécessaire, réduisant ainsi les délais de test.

Rubriques

- Utilisez-vous les tests d'applications pour la première fois ?
- Avantages des tests d'applications
- Intégration avec AWS CloudFormation
- Comment fonctionnent les tests d'applications
- Services connexes
- Accès aux tests d'applications
- Tarification des tests d'applications

Utilisez-vous les tests d'applications pour la première fois ?

Si vous utilisez les tests d'applications pour la première fois, nous vous recommandons de commencer par lire les sections suivantes :

- Concepts de test d'applications
- Tutoriel : Configuration de CardDemo l'application dans le cadre des tests d'applications
- the section called “Tutoriel : Rejouez et comparez sur AWS Blu Age en utilisant CardDemo”

Avantages des tests d'applications

Les tests d'applications offrent plusieurs avantages pour vous aider dans votre processus de migration :

- Tester l'accélération, l'agilité et la flexibilité.
- Concepts de test « Enregistrer une fois sur mainframe, rejouer plusieurs fois dans AWS ».
- Création iAc d'environnements cibles à l'aide de CloudFormation modèles fournis par l'utilisateur.
- Hauts degrés de répétabilité des tests.
- Conçu pour le cloud, dans un souci d'évolutivité et d'élasticité.
- Tests à grande échelle avec un haut degré d'automatisation.
- Rentabilité.

Intégration avec AWS CloudFormation

Les tests d'applications utilisent l'infrastructure sous forme de code avec AWS CloudFormation. Ce choix de conception simplifie et améliore votre expérience de test. AWS CloudFormation vous donne l'autonomie et l'indépendance nécessaires pour définir l'infrastructure la mieux adaptée à vos besoins. Vous pouvez sélectionner ou définir de nombreux paramètres (taille de l'instance, instance RDS, groupe de sécurité optimal) indépendamment. Vous pouvez ajouter des ressources, telles qu'une file d'attente Amazon SQS, dont vous avez besoin pour que votre application fonctionne correctement dans des conditions de test.

Dans les AWS CloudFormation modèles fournis au téléchargement, vous remarquerez certaines caractéristiques communes :

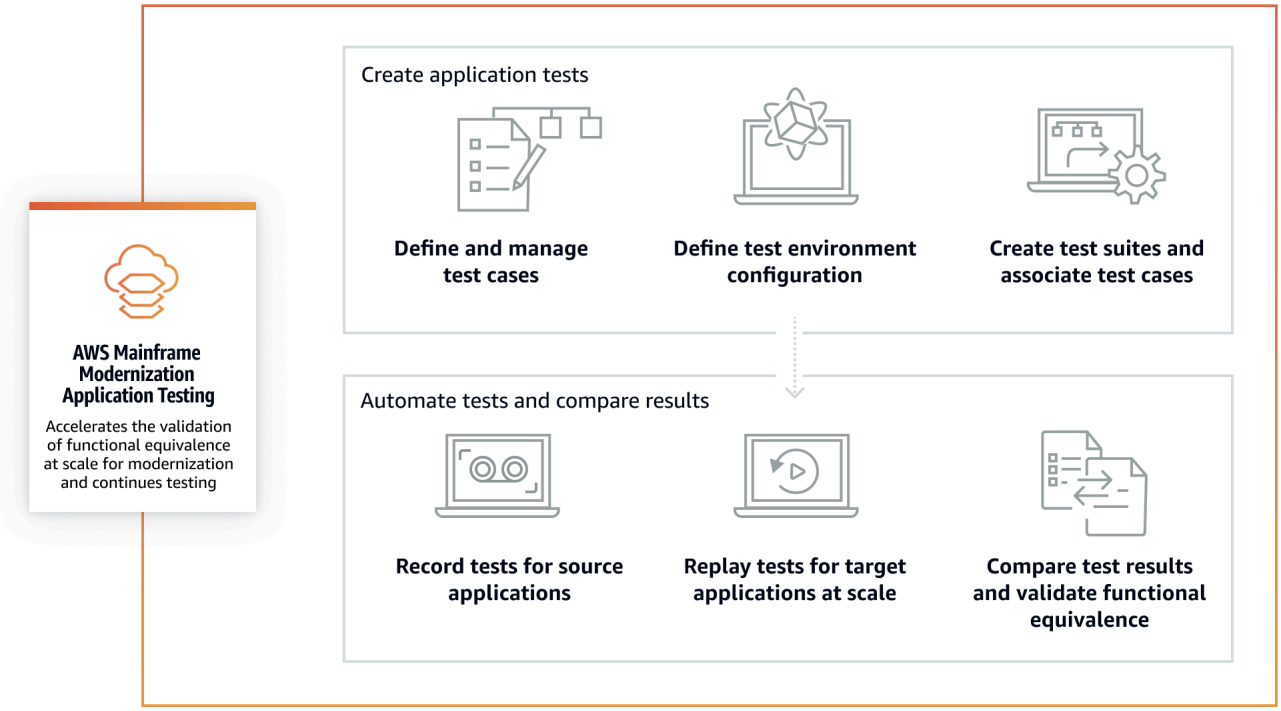
- Les tests d'applications créent une pile totalement isolée, comprenant un environnement d'exécution et une application de modernisation du AWS mainframe, dotés de leurs propres définitions de réseau et de sécurité. Cette pile isolée apporte de la résilience, car les autres acteurs de la même pile Compte AWS ne peuvent pas interférer avec les activités de test. Cela permet également d'éviter les situations dans lesquelles les opérateurs système modifient le VPC ou le groupe de sécurité par défaut, ce qui peut entraîner l'échec des activités de test.
- Le groupe de sécurité vous permet également de contrôler l'accès externe aux ressources utilisées lors des tests. Par exemple, une base de données peut contenir des données confidentielles.
- L'isolation complète empêche les autres acteurs partageant le VPC d'espionner le trafic.

- Elle améliore les performances. Par exemple, la communication entre l'application AWS Mainframe Modernization créée par le modèle et sa base de données Amazon RDS s'effectue sur un réseau distinct (un VPC privé), ce qui évite aux autres acteurs de ralentir le trafic.

Nous vous recommandons d'implémenter également ces fonctionnalités dans les AWS CloudFormation modèles que vous créez.

Comment fonctionnent les tests d'applications

La figure suivante donne un aperçu du fonctionnement des tests d'applications.



- Vous pouvez transférer les données d'entrée de la source à l' AWS utilisation Transfert de fichiers ou à vos outils préférés pour le transfert de données sur le mainframe.
- Vous exécutez la même logique métier à la fois sur la source et sur la cible.
- Les tests d'applications comparent automatiquement les données de sortie (ensembles de données, modifications de bases de données relationnelles, écrans 3270 en ligne et interactions utilisateur) à la fois de la source et de la cible. Après avoir exécuté votre scénario de test sur le mainframe, vous capturez les données de sortie, vous les transférez vers AWS, puis vous rejouez le scénario de test sur la cible. Les tests d'applications comparent automatiquement les données de sortie du test AWS avec les données de sortie de la source. Vous pouvez voir en un coup d'œil quels enregistrements sont identiques, équivalents, différents ou manquants. En outre, vous

pouvez définir des règles d'équivalence afin que les enregistrements qui ne sont pas identiques mais qui ont la même signification commerciale soient considérés comme équivalents.

Le flux de travail que vous suivez dans le cadre des tests d'applications comprend les étapes suivantes :

1. Création de cas de test : les cas de test sont la plus petite unité d'actions de test. Lorsque vous créez un scénario de test, vous identifiez également les types de données à comparer qui représentent le mieux l'équivalence fonctionnelle entre la source et la cible.
2. Définir la configuration de l'environnement de test : Spécifiez la configuration de votre environnement en spécifiant AWS CloudFormation un modèle et des attributs supplémentaires.
3. Création de suites de tests : les suites de tests sont un ensemble de cas de test.
4. Chargez des ensembles de données sur la source et rediffusez-les sur la cible : capturez les ensembles de données d'entrée et de sortie sur le mainframe, puis chargez-les sur. AWS Rejouez ensuite le scénario de test sur AWS.
5. Comparez les ensembles de données source et cible : les tests d'applications comparent automatiquement les ensembles de données de sortie provenant de la source et de la cible, afin que vous puissiez voir en un coup d'œil ce qui est correct et ce qui ne l'est pas.

L'action finale d'un scénario de test et l'objectif de l'ensemble du processus sont d'identifier les écarts entre les essais source et cible. Les tests d'applications comparent la version source et la version cible pour les données capturées sur tous les canaux d'interaction pendant le test. Il compare également les états finaux des données pertinentes (tels que définis dans les scénarios de test).

Services connexes

Les tests d'applications sont une fonctionnalité de la modernisation des AWS mainframes. Il utilise également l'infrastructure comme code AWS CloudFormation pour garantir la répétabilité, l'automatisation et la rentabilité des tests. Pour plus d'informations, consultez :

- AWS Modernisation du mainframe
- AWS CloudFormation

Accès aux tests d'applications

Vous pouvez accéder à la console de test d'applications depuis https://console.aws.amazon.com/apptest/ ou depuis la console de modernisation du AWS mainframe en choisissant Application Testing dans le volet de navigation de gauche.

Tarification des tests d'applications

Les tarifs des tests d'applications sont disponibles sur le site AWS Mainframe Modernization Pricing.

AWS Concepts de test des applications de modernisation du mainframe

AWS Les tests d'applications utilisent des termes que d'autres services de test ou progiciels peuvent utiliser avec une signification légèrement différente. Les sections suivantes expliquent comment les tests d'applications de modernisation des AWS mainframes utilisent cette terminologie.

Rubriques

- Cas de test
- Suite de tests
- Configuration de l'environnement de test
- Charger
- Relire
- Compare
- Comparaison de bases de données
- Comparaisons de jeux de
- État de la comparaison
- Règles d'équivalence
- Comparaison des ensembles de données sur l'état final
- Comparaisons de bases de données sur l'état
- Equivalence fonctionnelle (FE)
- Comparaisons d'écrans 3270 en ligne

- Rejouer les données
- Données de référence
- Téléchargez, rejouez et comparez
- Différences
- Équivalences
- Application source
- Application cible

Cas de test

Un scénario de test est l'unité d'action la plus atomique de votre flux de travail de test. Généralement, un scénario de test est utilisé pour représenter une unité indépendante de logique métier qui modifie les données. Des comparaisons seront effectuées pour chaque cas de test. Les cas de test sont ajoutés à une suite de tests. Les cas de test contiennent des métadonnées sur les artefacts de données (ensembles de données, bases de données) que le scénario de test modifie et sur les fonctions métier déclenchées lors de l'exécution du scénario de test : tâches par lots, boîtes de dialogue interactives 3270, etc. Par exemple, les noms et les pages de code des ensembles de données.

Données d'entrée → Scénario de test → Données de sortie

Les cas de test peuvent être en ligne ou par lots :

- Les cas de test d'écran 3270 en ligne sont des cas de test dans lesquels l'utilisateur exécute des boîtes de dialogue d'écran interactives (3270) pour lire, modifier ou produire de nouvelles données commerciales (bases de données et/ou enregistrements d'ensembles de données).
- Les cas de test par lots sont des cas de test nécessitant la soumission d'un lot pour lire, traiter et modifier ou produire de nouvelles données commerciales (ensembles de données et/ou enregistrements de base de données).

Suite de tests

Les suites de tests contiennent un ensemble de scénarios de test exécutés dans un ordre séquentiel, un par un. La rediffusion se fait au niveau de la suite de tests. Tous les scénarios de test de la suite de tests sont exécutés sur l'environnement de test cible lorsqu'une suite de tests est rejouée. S'il

existe des différences après avoir comparé les artefacts des tests de référence et de rediffusion, les différences seront affichées au niveau du scénario de test.

Par exemple, Test Suite A :

Cas de test 1, cas de test 2, cas de test 3, etc.

Configuration de l'environnement de test

La configuration de l'environnement de test vous permet de configurer l'ensemble initial de données et de paramètres de configuration (ou ressources) CloudFormation dont vous avez besoin pour rendre le test répétable.

Charger

Les téléchargements sont effectués au niveau de la suite de tests. Lors du chargement, vous devez fournir un emplacement Amazon S3 contenant les artefacts, les ensembles de données et les journaux CDC de la base de données relationnelle provenant du mainframe source à comparer. Elles seront considérées comme des données de référence provenant de l'ordinateur central source. Pendant la rediffusion, les données de rediffusion générées seront comparées aux données de référence téléchargées afin de garantir l'équivalence des applications.

Relire

Les rediffusions sont effectuées au niveau de la suite de tests. Pendant la rediffusion, les tests d'applications de modernisation du AWS mainframe utilisent le CloudFormation script pour créer l'environnement de test cible et exécuter l'application. Les ensembles de données et les enregistrements de base de données modifiés pendant la rediffusion sont capturés et comparés aux données de référence du mainframe. Généralement, vous effectuez le téléchargement sur le mainframe une seule fois, puis vous le rejouez plusieurs fois, jusqu'à ce que l'équivalence fonctionnelle soit atteinte.

Compare

Les comparaisons sont effectuées automatiquement une fois la rediffusion terminée avec succès. Lors des comparaisons, les données référencées que vous avez téléchargées et capturées pendant la phase de téléchargement sont comparées aux données de rediffusion générées pendant la phase de rediffusion. Les comparaisons sont effectuées séparément au niveau d'un scénario de test individuel pour les ensembles de données, les enregistrements de base de données et les écrans en ligne.

Comparaison de bases de données

Les tests d'applications utilisent une fonctionnalité de mise en correspondance de l'état de progression lors de la comparaison des modifications des enregistrements de base de données entre les applications source et cible. La correspondance par état et progression compare les différences entre chaque instruction INSERT, UPDATE et DELETE exécutée, contrairement à la comparaison des lignes du tableau à la fin du processus. La mise en correspondance de l'état d'avancement est plus efficace que les alternatives, car elle permet des comparaisons plus rapides et plus précises en comparant uniquement les données modifiées et en détectant les erreurs auto-corrigées dans le flux de transactions. En utilisant la technologie CDC (Changed Data Capture), les tests d'applications peuvent détecter les modifications de base de données de relations individuelles et les comparer entre la source et la cible.

Les modifications de la base de données relationnelle sont générées sur la source et la cible par le code de l'application testée à l'aide d'instructions DML (Data Modification Language) telles que SQL INSERT, UPDATE ou DELETE, mais également indirectement lorsque l'application utilise des procédures stockées, ou lorsque des déclencheurs de base de données sont définis sur certaines tables, ou lorsque CASCADE DELETE est utilisé pour garantir l'intégrité référentielle, déclenchant automatiquement des suppressions supplémentaires.

Comparaisons de jeux de

Les tests d'application comparent automatiquement les ensembles de données de référence et de réexécution produits sur les systèmes source (enregistrement) et cible (rediffusion).

Pour comparer des ensembles de données :

1. Commencez avec les mêmes données d'entrée (ensembles de données, base de données) à la fois sur la source et sur la cible.
2. Exécutez vos scénarios de test sur le système source (mainframe).
3. Capturez les ensembles de données produits et chargez-les dans un compartiment Amazon S3. Vous pouvez transférer des ensembles de données d'entrée de la source vers AWS des journaux, des écrans et des ensembles de données du CDC.
4. Spécifiez l'emplacement du compartiment Amazon S3 dans lequel les ensembles de données du mainframe ont été téléchargés lorsque vous avez chargé le scénario de test.

Une fois la rediffusion terminée, les tests d'application comparent automatiquement les ensembles de données de référence et de destination en sortie, en indiquant si les enregistrements sont identiques,

équivalents, différents ou manquants. Par exemple, les champs de date relatifs au moment de l'exécution de la charge de travail (jour +1, fin du mois en cours, etc.) sont automatiquement considérés comme équivalents. En outre, vous pouvez éventuellement définir des règles d'équivalence afin que les enregistrements qui ne sont pas identiques aient toujours la même signification commerciale et soient marqués comme équivalents.

État de la comparaison

Les tests d'applications utilisent les états de comparaison suivants : IDENTIQUE, ÉQUIVALENT et DIFFÉRENT.

IDENTIQUE

Les données source et cible sont exactement les mêmes.

ÉQUIVALENT

Les données source et cible contiennent de fausses différences considérées comme des équivalences, telles que des dates ou des horodatages qui n'affectent pas l'équivalence fonctionnelle lorsqu'elles sont relatives au moment de l'exécution de la charge de travail. Vous pouvez définir des règles d'équivalence pour identifier ces différences. Lorsque toutes les suites de tests rejouées par rapport à leurs suites de tests de référence affichent le statut IDENTIQUE ou ÉQUIVALENT, votre suite de tests ne montre aucune différence.

DIFFÉRENT

Les données source et cible présentent des différences, telles qu'un nombre différent d'enregistrements dans un ensemble de données ou des valeurs différentes dans le même enregistrement.

Règles d'équivalence

Ensemble de règles permettant d'identifier les fausses différences qui peuvent être considérées comme des résultats équivalents. Les tests d'équivalence fonctionnelle hors ligne (OFET) entraînent inévitablement des différences pour certains résultats entre les systèmes source et cible. Par exemple, les horodatages des mises à jour sont conçus différemment. Les règles d'équivalence expliquent comment ajuster ces différences et éviter les faux positifs au moment de la comparaison. Par exemple, si une date correspond à une durée d'exécution de plus de 2 jours dans une colonne de données donnée, la règle d'équivalence la décrit et accepte une durée sur le système cible

correspondant à une durée d'exécution de plus de 2 jours au lieu d'une valeur strictement égale à la même colonne dans le téléchargement de référence.

Comparaison des ensembles de données sur l'état final

État final des ensembles de données créés ou modifiés, y compris toutes les modifications ou mises à jour apportées aux ensembles de données par rapport à leur état initial. Pour les ensembles de données, Application Testing examine les enregistrements contenus dans ces ensembles de données à la fin de l'exécution d'un scénario de test et compare les résultats.

Comparaisons de bases de données sur l'état

Comparaisons des modifications apportées aux enregistrements de base de données sous la forme d'une séquence d'instructions DML (Delete, Update, Insert) individuelles. Les tests d'applications comparent les modifications individuelles (insertion, mise à jour ou suppression d'une ligne de table) entre la base de données source et la base de données cible, et identifient les différences pour chaque modification individuelle. Par exemple, une instruction INSERT individuelle peut être utilisée pour insérer dans une table une ligne avec des valeurs différentes dans la base de données source par rapport à la base de données cible.

Equivalence fonctionnelle (FE)

Deux systèmes sont considérés comme fonctionnellement équivalents s'ils produisent les mêmes résultats sur toutes les opérations observables, avec les mêmes données d'entrée. Par exemple, deux applications sont considérées comme fonctionnellement équivalentes si les mêmes données d'entrée produisent des données de sortie identiques (par le biais d'écrans, de modifications d'ensembles de données ou de modifications de base de données).

Comparaisons d'écrans 3270 en ligne

Compare la sortie des écrans 3270 du mainframe avec la sortie des écrans Web des applications modernisées lorsque le système cible s'exécute sous le runtime AWS Blu Age dans le. AWS Cloud Et il compare la sortie des écrans 3270 du mainframe à celle des 3270 écrans de l'application réhébergée lorsque le système cible s'exécute sous le runtime Rocket Software (anciennement Micro Focus) dans le. AWS Cloud

Rejouer les données

Les données de rediffusion sont utilisées pour décrire les données générées par la réexécution d'une suite de tests sur l'environnement de test cible. Par exemple, les données de rediffusion sont générées lorsqu'une suite de tests est exécutée sur une application de service de modernisation AWS du mainframe. Les données de rediffusion sont ensuite comparées aux données de référence capturées à partir de la source. Chaque fois que vous rejouez la charge de travail dans l'environnement cible, une nouvelle génération de données de rediffusion est générée.

Données de référence

Les données de référence sont utilisées pour décrire les données capturées sur le mainframe source. Il s'agit de la référence à laquelle les données générées par le replay (cible) seront comparées. En général, pour chaque enregistrement du mainframe qui crée des données de référence, de nombreuses rediffusions sont effectuées. Cela est dû au fait que les utilisateurs capturent généralement l'état correct de l'application sur le mainframe et rejouent les scénarios de test sur l'application modernisée cible pour valider l'équivalence. Si des bogues sont détectés, ils sont corrigés et les scénarios de test sont rejoués. Souvent, plusieurs cycles de rediffusion, de correction de bogues et de rediffusion pour valider l'occurrence. C'est ce qu'on appelle le paradigme des tests « capture une fois, rejouer plusieurs fois ».

Téléchargez, rejouez et comparez

Les tests d'applications se déroulent en trois étapes :

- Téléchargement : capture les données référencées créées sur le mainframe pour chaque scénario de test d'un scénario de test. Il peut s'agir de 3 270 écrans en ligne, d'ensembles de données et d'enregistrements de base de données.
  - Pour les écrans 3270 en ligne, vous devez utiliser l'émulateur de terminal Blu Insights pour capturer votre charge de travail source. Pour plus d'informations, consultez la documentation Blu Insights.
  - Pour les ensembles de données, vous devrez capturer les ensembles de données produits par chaque scénario de test sur le mainframe à l'aide d'outils courants, tels que le FTP ou le service de transfert de jeux de données intégré à la modernisation du AWS mainframe.
  - Pour les modifications de base de données, vous utilisez la documentation AWS Mainframe Modernization Data Replication with Precisely pour capturer et générer des journaux CDC contenant les modifications.

- Rediffusion : La suite de tests est rejouée dans l'environnement cible. Tous les cas de test spécifiés dans la suite de tests sont exécutés. Les types de données spécifiques créés par les scénarios de test individuels, tels que les ensembles de données, les modifications de bases de données relationnelles ou les écrans 3270, seront capturés automatiquement. Ces données sont appelées données de rediffusion et seront comparées aux données de référence capturées lors de la phase de téléchargement.

  > **Note**
  >
  > Les modifications apportées à la base de données relationnelle nécessiteront des options de configuration spécifiques au DMS dans votre modèle de condition initial. CloudFormation

- Comparaison : la source des tests, les données de référence et les données de rediffusion cibles sont comparées, et les résultats vous seront présentés sous forme de données identiques, différentes, équivalentes ou manquantes.

Différences

Indique que des différences ont été détectées entre les ensembles de données de référence et de rediffusion par comparaison des données. Par exemple, un champ d'un écran 3270 en ligne qui affiche des valeurs différentes du point de vue de la logique métier entre le mainframe source et l'application modernisée cible sera considéré comme une différence. Un autre exemple est un téléchargement dans un ensemble de données qui n'est pas identique entre les applications source et cible.

Équivalences

Les enregistrements équivalents sont des enregistrements différents entre les ensembles de données de référence et de réexécution, mais ne doivent pas être traités comme différents du point de vue de la logique métier. Par exemple, un enregistrement contenant l'horodatage de la date de production de l'ensemble de données (heure d'exécution de la charge de travail). À l'aide de règles d'équivalence personnalisables, vous pouvez demander à Application Testing de traiter une telle différence faussement positive comme une équivalence, même si elle indique des valeurs différentes entre les données de référence et les données de rediffusion.

Application source

L'application mainframe source à laquelle la comparaison doit être effectuée.

Application cible

Application nouvelle ou modifiée sur laquelle les tests sont effectués et qui sera comparée à l'application source afin de détecter tout défaut et d'obtenir une équivalence fonctionnelle entre les applications source et cible. L'application cible s'exécute généralement dans le AWS cloud.

AWS Conditions préalables aux tests des applications de modernisation du mainframe

AWS La fonctionnalité de test des applications de modernisation du AWS mainframe vous permet d'effectuer des tests d'équivalence fonctionnelle automatisés pour vos projets de migration. Pour préparer l'utilisation des tests d'applications dans la console de modernisation du AWS mainframe, procédez comme suit :

1. Définissez les scénarios de test : définissez les unités de test de base que vous souhaitez exécuter et rejouer dans un ordre spécifique, pour votre application cible. Pour plus d'informations sur la façon de créer des scénarios de test, consultezthe section called "Création de cas de test dans Application Testing".
2. Préparer le CloudFormation modèle et les données d'entrée : créez un CloudFormation modèle qui sera utilisé pour approvisionner l'environnement de test cible. Les variables de ce modèle seront utilisées pour ajouter des données d'entrée et des noms de variables de sortie dans votre application de modernisation AWS du mainframe. Pour plus d'informations, consultez la section Utilisation du AWS CloudFormation modèle dans le Guide de AWS CloudFormation l'utilisateur.
3. Garantir l'accès au mainframe et la capture des données : Vérifiez que vous avez accès au mainframe source. Cela vous permettra également de capturer et de télécharger les données sources générées par les applications exécutées sur le mainframe.

Workflows de la console de test des applications

AWS La console de test des applications de modernisation du mainframe vous permet de créer des scénarios de test, des suites de tests et des configurations d'environnement de test.

Rubriques

- Création de cas de test dans le cadre des tests d'applications de modernisation du AWS mainframe
- Création de suites de tests dans le cadre des tests d'applications de modernisation des AWS mainframes
- Création de configurations d'environnement de test dans le cadre des tests d'applications de modernisation du AWS mainframe

Création de cas de test dans le cadre des tests d'applications de modernisation du AWS mainframe

Un cas de test est une unité atomique qui représente une certaine action dans votre flux de travail. Pour plus d'informations sur les différents concepts, voir???.

> **Important**
>
> Vous devez d'abord créer au moins une configuration d'environnement de test avant d'exécuter des scénarios de test. Pour créer votre première configuration d'environnement, consultezthe section called "Création de configurations d'environnement de test dans Application Testing".

Rubriques

- Création d'un cas de test Batch
- Création d'un scénario de test d'écran 3270 en ligne

Création d'un cas de test Batch

Les cas de test par lots vous permettent de soumettre un lot pour lire, traiter et modifier ou produire de nouvelles données commerciales (bases de données et/ou enregistrements d'ensembles de données).

Pour créer un scénario de test Batch

1. Ouvrez la console de test des applications de modernisation du AWS mainframe à https://console.aws.amazon.com/apptest/ l'adresse.

2. Dans le Région AWS sélecteur, choisissez la région dans laquelle les tests d'applications sont disponibles.

> **Note**
>
> Les tests d'applications sont actuellement disponibles uniquement dans les régions de l'est des États-Unis (Virginie du Nord), de l'Asie-Pacifique (Sydney), de l'Europe (Francfort) et de l'Amérique du Sud (São Paulo).

3. Dans le volet de navigation de gauche, sélectionnez Test cases.
4. Dans Définir un scénario de test, entrez le nom de votre scénario de test et une description facultative. Choisissez Batch sous Type de scénario de test.
5. Choisissez Next (Suivant).
6. (Facultatif) Sur la page Spécifier les paramètres JCL par lots, ajoutez le nom JCL (langage de contrôle des tâches) et les paramètres de votre tâche (noms et valeurs).
7. Choisissez Next (Suivant).
8. Sur la page Source de données à capturer, vous pouvez choisir soit les modifications de base de données relationnelle, soit les ensembles de données, soit les deux.

   - Choisissez Relational database changes lorsque vous souhaitez que le scénario de test modifie les enregistrements de base de données.
   - Choisissez Ensembles de données lorsque vous souhaitez que le scénario de test modifie des ensembles de données. Sous Ensembles de données de sortie, ajoutez le nom de votre ensemble de données de sortie.

   > **Note**
   >
   > Vous pouvez ajouter plusieurs ensembles de données.

9. Choisissez Next (Suivant).
10. Sur la page Réviser et créer, passez en revue toutes les informations et choisissez Créer un scénario de test.

Création d'un scénario de test d'écran 3270 en ligne

Les scénarios de test d'écran 3270 en ligne vous permettent d'exécuter des boîtes de dialogue d'écran interactives (3270) pour lire, modifier ou produire de nouvelles données commerciales (bases de données et/ou enregistrements d'ensembles de données).

Pour créer un scénario de test d'écran 3270 en ligne

1. Ouvrez la console de test des applications de modernisation du AWS mainframe à https://console.aws.amazon.com/apptest/ l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle les tests d'applications sont disponibles.

   > **Note**
   >
   > Les tests d'applications sont actuellement disponibles uniquement dans les régions de l'est des États-Unis (Virginie du Nord), de l'Asie-Pacifique (Sydney), de l'Europe (Francfort) et de l'Amérique du Sud (São Paulo).

3. Dans le volet de navigation de gauche, sélectionnez Test cases.
4. Dans Définir un scénario de test, entrez le nom de votre scénario de test et une description facultative. Choisissez Online 3270 screens sous Type de scénario de test.
5. Choisissez Next (Suivant).

   > **Note**
   >
   > L'écran 3270 en ligne ne vous oblige pas à spécifier les paramètres JCL.

6. Choisissez Next (Suivant).
7. Sur la page Source de données à capturer, la sélection par défaut est Online 3270 screens. En outre, vous pouvez choisir les modifications de base de données relationnelles et les ensembles de données.

   - Choisissez Relational database changes lorsque vous souhaitez que le scénario de test modifie les enregistrements de base de données.
   - Choisissez Ensembles de données lorsque vous souhaitez que le scénario de test modifie des ensembles de données. Sous Ensembles de données de sortie, ajoutez le nom de votre ensemble de données de sortie.

> **Note**
>
> Vous pouvez ajouter plusieurs ensembles de données.

8. Choisissez Next (Suivant).
9. Sur la page Réviser et créer, passez en revue toutes les informations et choisissez Créer un scénario de test.

Création de suites de tests dans le cadre des tests d'applications de modernisation des AWS mainframes

Les suites de tests sont des séries de scénarios de test exécutés dans un ordre séquentiel. Les suites de tests sont importantes pour rejouer les scénarios de test.

> **Important**
>
> Avant de créer des suites de tests, vous devez disposer d'au moins un scénario de test. Vous pouvez créer votre premier cas de test en utilisant,the section called "Création de cas de test dans Application Testing".

Pour plus d'informations sur les différents concepts, voirthe section called "Concepts de test d'applications".

Rubriques

- Création d'une suite de tests
- Charger les données de référence
- Rejouez et comparez

Création d'une suite de tests

Les suites de tests vous permettent d'exécuter différents scénarios de test, de les rejouer et de les comparer ultérieurement.

Pour créer une suite de tests

1. Ouvrez la console de test des applications de modernisation du AWS mainframe à https://console.aws.amazon.com/apptest/ l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle les tests d'applications sont disponibles.

   > **Note**
   >
   > Les tests d'applications sont actuellement disponibles uniquement dans les régions de l'est des États-Unis (Virginie du Nord), de l'Asie-Pacifique (Sydney), de l'Europe (Francfort) et de l'Amérique du Sud (São Paulo).

3. Dans le volet de navigation de gauche, sélectionnez Test cases.
4. Choisissez Créer des suites de tests.
5. Dans la section Créer des suites de tests, recherchez des cas de test dans la bibliothèque de cas de test, puis choisissez Ajouter des cas de test sélectionnés.

   > **Note**
   >
   > Vous pouvez ajouter jusqu'à 20 scénarios de test dans une suite de tests.

6. Dans le volet de la suite de tests, entrez le nom de votre suite de tests et une description facultative. Vous pouvez également choisir entre un environnement d'exécution géré ou un environnement d'exécution non géré, qui définira la manière dont la suite de tests configure et déconfigure une AWS application de modernisation du mainframe. Ajoutez éventuellement l'URI JSON S3 du jeu de données d'importation AWS Mainframe Modernization.
7. Dans la section Ajout de scénarios de test, empilez vos scénarios de test dans l'ordre dans lequel vous souhaitez les télécharger et rejouez-les.
8. Choisissez Créer une suite de tests.

Charger les données de référence

Téléchargez les données de référence du mainframe dans AWS Application Testing. Il vous suffit d'enregistrer les données de référence téléchargées pour la première fois. Le service de test peut réutiliser les résultats téléchargés depuis la source et les comparer consécutivement aux résultats rejoués sur la cible.

Pour télécharger des données de référence

1. Dans la section Suites de tests, choisissez la suite de tests pour télécharger les données de référence.
2. Choisissez Charger.
3. Sur la page Charger les données de référence, sélectionnez les scénarios de test que vous souhaitez rejouer. Complétez les champs pour la date de capture des données, l'emplacement S3 du journal des modifications de base de données, l'emplacement S3 des ensembles de données et Choose Upload.

Rejouez et comparez

Le processus de rediffusion et de comparaison associe votre scénario de test à l'environnement de test cible et exécute l'application. Vous devez télécharger les données avant de lancer le processus de rediffusion.

Pour rejouer et comparer

1. Dans la section Suites de tests, choisissez la suite de tests à rejouer.
2. Choisissez Replay et comparez.
3. Sur la page de présentation de la rediffusion et de la comparaison, sélectionnez la configuration de votre environnement de test et consultez les informations. La fonction d'édition vous permet de modifier tous les champs de configuration de l'environnement de test. Vous pouvez également trouver des AWS CloudFormation paramètres.
4. Dans la section Cas de test à rejouer, choisissez les cas de test et placez-les dans l'ordre dans lequel vous souhaitez les rejouer.
5. Choisissez Replay et comparez.

Création de configurations d'environnement de test dans le cadre des tests d'applications de modernisation du AWS mainframe

Les configurations de l'environnement de test vous permettent de configurer l'ensemble initial de données et de paramètres de configuration (ou ressources) AWS CloudFormation dont vous avez besoin pour rendre le test reproductible.

Pour plus d'informations sur les différents concepts, voirthe section called "Concepts de test d'applications".

Création d'une configuration d'environnement de test

Configurez votre environnement de test pour rejouer et comparer les cas de test dans Application Testing.

Configuration des configurations de l'environnement de test

1. Ouvrez la console de test des applications de modernisation du AWS mainframe à https://console.aws.amazon.com/apptest/ l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle les tests d'applications sont disponibles.

   > **Note**
   >
   > Les tests d'applications sont actuellement disponibles uniquement dans les régions de l'est des États-Unis (Virginie du Nord), de l'Asie-Pacifique (Sydney), de l'Europe (Francfort) et de l'Amérique du Sud (São Paulo).

3. Dans le volet de navigation de gauche, sélectionnez Tester les configurations de l'environnement.
4. Choisissez Créer une configuration d'environnement de test.
5. Dans le volet Créer une configuration d'environnement de test, entrez le nom et la description. Ajoutez également votre compartiment Amazon S3 qui contient le CloudFormation modèle pour les tests d'applications. De plus, vous pouvez ajouter les paramètres CloudFormation d'entrée qui seront utilisés lors de la création de la CloudFormation pile.
6. Spécifiez votre application de modernisation du mainframe AWS qui sera affectée par cette configuration de test. Ajoutez le nom de la variable de sortie pour l'ID de l'application AWS Mainframe Modernization, le moteur d'exécution (AWS Blu Age non géré ou géré par Rocket Software (anciennement Micro Focus)).

> **Note**
>
> Le nom de la variable de sortie pour l'ID d'application AWS Mainframe Modernization doit correspondre au nom de la variable de sortie du CloudFormation modèle pour la création de la pile.

> **Important**
>
> L'environnement d'exécution non géré AWS Blu Age vous oblige également à spécifier le nom de la variable de sortie pour l'ID de service du point de terminaison VPC, le nom de la variable de sortie pour le port du récepteur et le nom de la variable de sortie pour le nom. WebApp Ces noms doivent correspondre aux noms des variables de sortie du CloudFormation modèle.

7. (Facultatif) Un attribut supplémentaire tel que le nom de la variable de sortie peut être défini pour la tâche Amazon Resource Name (ARN) du Database Migration Service (DMS), qui est utilisée pour capturer les modifications de base de données relationnelles. Un autre attribut est l'URI DDL S3 de la base de données source.

> **Important**
>
> Le nom de la variable de sortie doit correspondre au nom de la variable du CloudFormation modèle.

8. (Facultatif) Personnalisez votre clé du service de gestion des clés (KMS). Pour plus d'informations, consultez Gestion de l'accès aux clés gérées par le client dans le Guide du développeur AWS Key Management Service .
9. Choisissez Créer une configuration d'environnement de test.

Tutoriel : Configuration de l' CardDemo exemple d'application dans le cadre des tests d'applications de modernisation AWS du mainframe

Dans le cadre de ce didacticiel, vous allez créer une AWS CloudFormation pile qui vous aide à configurer l'CardDemo exemple d'application pour le replatforming avec le service géré Micro Focus on AWS Mainframe Modernization et des fonctionnalités telles que les tests d'applications de modernisation des AWS mainframes. Le didacticiel décrit un exemple de AWS CloudFormation modèle que vous pouvez utiliser pour créer la pile. Nous fournissons également un fichier compressé contenant les artefacts d'application nécessaires. L'exemple de modèle fournit une base de données, un environnement d'exécution, une application et un environnement réseau totalement isolé.

Ce modèle crée plusieurs AWS ressources. Ils vous seront facturés si vous créez une pile à partir de ce modèle.

Prérequis

- Téléchargez et décompressez le fichier `IC3-card-demo-zip` et. `datasets_Mainframe_ebcdic.zip` Ces fichiers contiennent les CardDemo exemples et les ensembles de données à utiliser pour les tests d' AWS applications.
- Créez un compartiment Amazon S3 pour contenir les CardDemo fichiers et autres artefacts. Par exemple, `my-carddemo-bucket`.

Étape 1 : Préparation de la configuration CardDemo

Téléchargez les fichiers CardDemo d'exemple et modifiez le AWS CloudFormation modèle qui créera l' CardDemoapplication.

1. Téléchargez les `IC3-card-demo` dossiers `datasets_Mainframe_ebcdic` et que vous avez décompressés précédemment dans votre bucket.
2. Téléchargez le `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation modèle depuis votre bucket. Il se trouve dans le `IC3-card-demo` dossier.
3. Modifiez le `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation modèle comme suit :
    - Remplacez le `BucketName` paramètre par le nom du compartiment que vous avez défini précédemment, par exemple`my-carddemo-bucket`.

- Remplacez `ImportJsonPath` le fichier par l'emplacement du `mf-carddemo-datasets-import.json` fichier dans votre compartiment. Par exemple, la `s3://my-carddemo-bucket/IC3-card-demo/mf-carddemo-datasets-import.json` mise à jour de cette valeur garantit que la sortie `M2ImportJson` contient la bonne valeur.
- (Facultatif) Adaptez les `InstanceType` paramètres `EngineVersion` et en fonction de vos normes.

> **Note**
>
> Ne modifiez pas les `M2ApplicationId` sorties `M2EnvironmentId` et. Les tests d'applications utilisent ces valeurs pour localiser les ressources avec lesquelles elles interagiront.

Étape 2 : créer toutes les ressources nécessaires

Exécutez votre AWS CloudFormation modèle personnalisé pour créer toutes les ressources dont vous avez besoin pour terminer ce didacticiel avec succès. Ce modèle configure l' CardDemo application afin que vous puissiez l'utiliser lors des tests.

1. Connectez-vous à la AWS CloudFormation console et choisissez Create stack, puis choisissez With new resources (standard).
2. Dans Prérequis - Préparer le modèle, sélectionnez Le modèle est prêt.
3. Dans Spécifier le modèle, choisissez Charger un fichier modèle, puis choisissez Choisir un fichier.
4. Naviguez jusqu'à l'endroit où vous avez téléchargé `aws-m2-math-mf-carddemo.yaml` le fichier, puis choisissez Next.
5. Dans Spécifier les détails de la pile, donnez un nom à la pile afin que vous puissiez la retrouver facilement dans une liste, puis choisissez Suivant.
6. Dans Configurer les options de pile, conservez les valeurs par défaut et choisissez Next.
7. Dans Révision, vérifiez ce qui AWS CloudFormation est créé pour vous, puis choisissez Soumettre.

La création de la pile prend environ 10 AWS CloudFormation à 15 minutes.

> **Note**
>
> Le modèle est configuré pour ajouter un suffixe unique aux noms des ressources qu'il crée. Cela signifie que vous pouvez créer plusieurs instances de ce modèle de pile en parallèle, une fonctionnalité clé pour les tests d'applications qui vous permet d'exécuter plusieurs suites de tests en même temps.

Étape 3 : Déployer et démarrer l'application

Déployez l' CardDemo application AWS CloudFormation créée pour vous et assurez-vous qu'elle est en cours d'exécution.

1. Ouvrez la console AWS Mainframe Modernization et sélectionnez Applications dans le menu de navigation de gauche.
2. Choisissez l' CardDemo application, qui porte un nom comme`aws-m2-math-mf-carddemo-abc1d2e3`.
3. Choisissez Actions, puis choisissez Déployer l'application.
4. Dans Environnements, choisissez l'environnement d'exécution correspondant à l'application. Le même identifiant unique sera ajouté à la fin du nom. Par exemple, `aws-m2-math-mf-carddemo-abc1d2e3`.
5. Choisissez Déployer. Attendez que l'application soit déployée avec succès et qu'elle soit à l'état Prêt.
6. Choisissez l'application, puis sélectionnez Actions et Démarrer l'application. Attendez que l'application soit en cours d'exécution.
7. Sur la page des détails de l'application, copiez le port et le nom d'hôte DNS dont vous avez besoin pour vous connecter à l'application en cours d'exécution.

Étape 4 : Importer les données initiales

Pour utiliser l' CardDemo exemple d'application, vous devez importer un premier ensemble de données. Procédez comme suit.

1. Téléchargez le fichier `mf-carddemo-datasets-import.json`.
2. Modifiez le fichier dans votre éditeur de texte préféré.

3. Localisez le `s3Location` paramètre et mettez à jour la valeur pour qu'elle pointe vers le compartiment Amazon S3 que vous avez créé.
4. Effectuez la même modification pour toutes les occurrences des`s3Location`, puis enregistrez le fichier.
5. Connectez-vous à la console Amazon S3 et accédez au compartiment que vous avez créé précédemment.
6. Téléchargez le `mf-carddemo-datasets-import.json` fichier personnalisé.
7. Ouvrez la console AWS Mainframe Modernization et sélectionnez Applications dans le menu de navigation de gauche.
8. Choisissez l' CardDemo application.
9. Choisissez Ensembles de données, puis Importer.
10. Accédez à l'emplacement dans Amazon S3 où vous avez chargé le fichier JSON personnalisé et choisissez Soumettre.

Cette tâche importe 23 ensembles de données. Pour contrôler le résultat de la tâche d'importation, consultez la console. Lorsque tous les ensembles de données sont correctement importés, connectez-vous à l'application.

> **Note**
>
> Lorsque vous utilisez ce modèle dans le cadre de tests d'applications, la sortie gère `M2ImportJson` automatiquement le processus d'importation.

Étape 5 : Connectez-vous à l' CardDemoapplication

Connectez-vous à l' CardDemo exemple d'application à l'aide de l'émulateur 3270 de votre choix.

- Lorsque l'application est en cours d'exécution, utilisez votre émulateur 3270 pour vous connecter à l'application, en spécifiant le nom d'hôte DNS et le nom de port, si nécessaire.

  Par exemple, si vous utilisez l'émulateur open source c3270, votre commande ressemble à ceci :

  ```
  c3270 -port port-number DNS-hostname
  ```

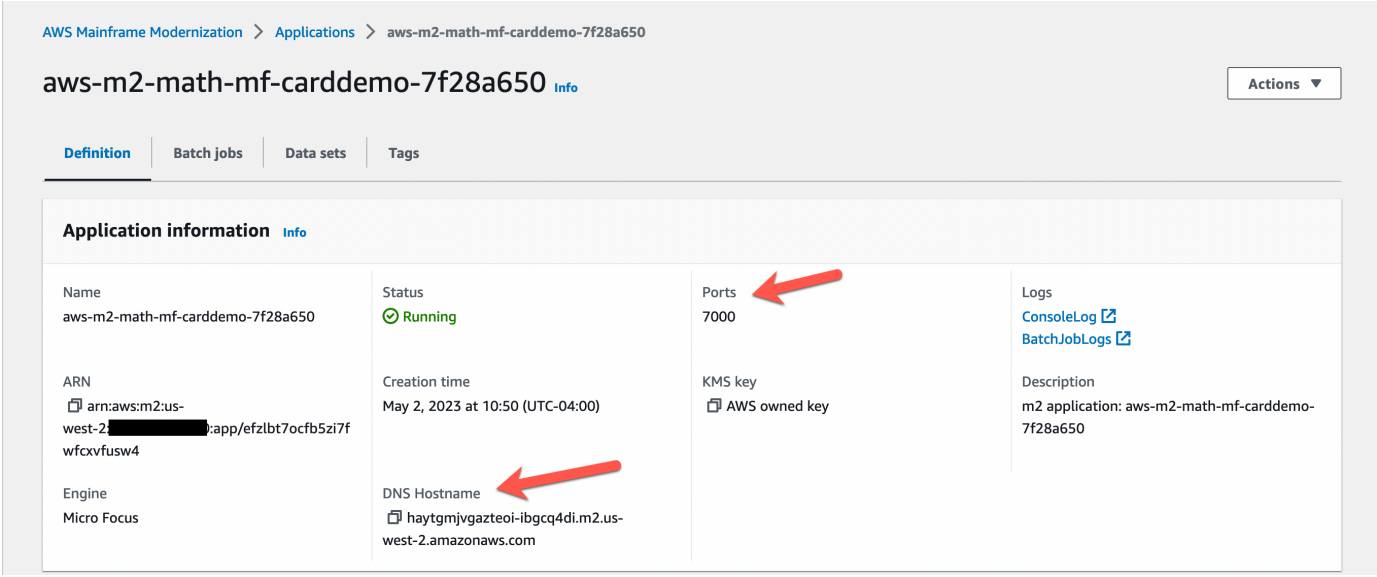
port

Port spécifié sur la page détaillée de l'application. Par exemple, 6000.

Hostname

Le nom d'hôte DNS spécifié sur la page détaillée de l'application.

La figure suivante indique où trouver le port et le nom d'hôte DSN.



Tutoriel : Rejouer et comparer dans le cadre des tests d'applications de modernisation AWS du mainframe à l'aide CardDemo de AWS Blu Age déployée sur Amazon EC2

Dans ce didacticiel, vous allez suivre les étapes requises pour rejouer et comparer les charges de travail de test avec l' CardDemo application exécutée sur AWS Blu Age déployée sur Amazon. EC2

Étape 1 : Obtenir une image de machine Amazon EC2 Amazon AWS Blu Age (AMI)

Suivez les instructions du didacticiel de configuration de AWS Blu Age Runtime (sur Amazon EC2) pour connaître les étapes d'intégration requises pour accéder à l' EC2 AMI AWS Blu Age sur Amazon.

Étape 2 : démarrer une EC2 instance Amazon à l'aide de l'AMI AWS Blu Age

1. Configurez vos AWS informations d'identification.
2. Identifiez l'emplacement du fichier binaire Amazon EC2 AMI 3.5.0 (CLI seulement/version AWS Blu Age) dans le compartiment Amazon S3 :

```
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/3.5.0/AMI/
```

> **Note**
>
> La fonctionnalité de test d'application n'est disponible que dans 4 régions de prod (us-east-1, sa-east-1, eu-central-1 et ap-southeast-2).

3. Restaurez l'AMI dans votre compte à l'aide de la commande suivante :

```
aws ec2 create-restore-image-task --object-key 3.5.0/AMI/ami-0182ffe3b9d63925b.bin
 --bucket aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1 --region eu-west-1 --name
 "AWS BLUAGE RUNTIME AMI"
```

> **Note**
>
> Remplacez le nom du fichier bin de l'AMI et la région dans laquelle vous souhaitez créer l'AMI.

4. Après avoir créé une EC2 instance Amazon, vous pouvez trouver l'ID d'AMI correct qui a été restauré à partir du compartiment Amazon S3 dans le catalogue EC2 d'images Amazon.

> **Note**
>
> Dans ce didacticiel, l'ID de l'AMI est ami-0d0fafcc636fd1e6d, et vous devez remplacer cet identifiant dans les différents fichiers de configuration par celui qui vous a été fourni.

1. Si aws ec2 create-restore-image-task échoue, vérifiez votre version de Python et de la CLI à l'aide de la commande suivante :

```
aws --version
```

> **Note**
>
> La version Python doit être >= 3 et la version CLI doit être >= 2.

2. Si ces versions sont obsolètes, la CLI doit être mise à jour. Pour mettre à jour la CLI :
   a. Suivez les instructions de la section Installer ou mettre à jour la dernière version de l'AWS CLI.
   b. Supprimez la CLI v1 à l'aide de la commande suivante :

```
sudo yum remove awscli
```

   c. Et installez CLI v2 avec les commandes suivantes :

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

   d. Enfin, vérifiez la version de Python et de la CLI avec la commande suivante :

```
aws --version
```

3. Vous pouvez ensuite refaire le aws create-restore-image-task ec2.

Étape 3 : télécharger les fichiers CardDemo dépendants sur S3

Copiez le contenu des dossiers, des bases de données, du système de fichiers et des données utilisateur. Téléchargez et décompressez les CardDemo applications. Ces trois dossiers doivent être copiés dans l'un de vos compartiments appelé amzn-s3-demo-bucket dans cette documentation.

Étape 4 : Chargement des bases de données et initialisation de l'application CardDemo

Créez une EC2 instance Amazon temporaire que vous utiliserez comme ressource de calcul pour générer les instantanés de base de données requis pour l' CardDemoapplication. Cette EC2 instance

n'exécutera pas l' CardDemo application elle-même, mais générera les instantanés de base de données qui seront utilisés ultérieurement.

Commencez par modifier le CloudFormation modèle fourni nommé « load-and-create-ba - snapshots.yml ». Il s'agit du CloudFormation modèle utilisé pour créer l' EC2 instance Amazon utilisée pour générer les instantanés de base de données.

1. Générez et fournissez EC2 la paire de clés qui sera utilisée pour l' EC2 instance. Pour plus d'informations, consultez la section Création de paires de clés.

   Exemple :

   ```
   Ec2KeyPair:
       Description: 'ec2 key pair'
       Default: 'm2-tests-us-west-2'
       Type: String
   ```

2. Spécifiez le chemin Amazon S3 de votre dossier dans lequel vous avez placé le dossier de base de données à l'étape précédente :

   ```
   S3DBScriptsPath:
       Description: 'S3 DB scripts folder path'
       Type: String
       Default: 's3://amzn-s3-demo-bucket/databases'
   ```

3. Spécifiez le chemin Amazon S3 de votre dossier dans lequel vous avez placé le dossier du système de fichiers de l'étape précédente :

   ```
   S3ApplicationFilesPath:
       Description: 'S3 application files folder path'
       Type: String
       Default: 's3://amzn-s3-demo-bucket/file-system'
   ```

4. Spécifiez le chemin Amazon S3 de votre dossier dans lequel vous avez placé le dossier userdata de l'étape précédente :

   ```
   S3UserDataPath:
       Description: 'S3 userdata folder path'
       Type: String
       Default: 's3://amzn-s3-demo-bucket/userdata'
   ```

5. Spécifiez également un chemin Amazon S3 dans lequel vous enregistrerez les fichiers de résultats à utiliser à l'étape suivante.

```
S3SaveProducedFilesPath:
    Description: 'S3 path folder to save produced files'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/post-produced-files'
```

6. Modifiez l'ID de l'AMI par le bon ID obtenu précédemment dans ce didacticiel à l'aide du modèle suivant :

```
  BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0bd41245734fd20d9'
    Type: String
```

- Vous pouvez éventuellement modifier le nom des trois instantanés qui seront créés lors de l'exécution des bases de données de chargement avec CloudFormation. Ils seront visibles dans la CloudFormation pile lors de sa création et seront utilisés ultérieurement dans ce didacticiel. N'oubliez pas de noter les noms utilisés pour les instantanés de base de données.

```
SnapshotPrimary:
    Description: 'Snapshot Name DB BA Primary'
    Type: String
    Default: 'snapshot-primary'

  SnapshotBluesam:
    Description: 'Snapshot Name DB BA Bluesam'
    Type: String
    Default: 'snapshot-bluesam'

  SnapshotJics:
    Description: 'Snapshot Name DB BA Jics'
    Type: String
    Default: 'snapshot-jics'
```

> **Note**
>
> Dans ce document, nous partons du principe que le nom des instantanés reste cohérent.

7. Exécutez la CLI ou AWS la console CloudFormation avec le bouton Create Stack et l'assistant. À la fin du processus, vous devriez voir apparaître trois instantanés dans la console RDS portant le nom que vous avez choisi suivi d'un identifiant unique. Vous aurez besoin de ces noms à l'étape suivante.

   > **Note**
   >
   > RDS ajoutera des suffixes aux noms des instantanés définis dans le modèle. AWS CloudFormation Assurez-vous d'obtenir le nom complet du snapshot auprès de RDS avant de passer à l'étape suivante.

   Exemple de commande CLI-

   ```
   aws cloudformation create-stack --stack-name load-and-create-ba-snapshots --
   template-url https://your-apptest-bucket.s3.us-west-2.amazonaws.com/load-and-
   create-ba-snapshots.yml --capabilities CAPABILITY_NAMED_IAM
   ```

   Vous pouvez également vérifier dans le chemin Amazon S3 que vous avez fourni pour S3 SaveProducedFilesPath que les ensembles de données ont été correctement créés.

Étape 5 : Lancez le moteur d'exécution AWS Blu Age CloudFormation

CloudFormation À utiliser pour exécuter l' EC2 instance Amazon avec l'application CardDemo AWS Blu Age. Vous devez remplacer certaines variables du CloudFormation nom en `m2-with-ba-using-snapshots-https-authentication.yml` éditant le fichier YAML ou en modifiant les valeurs dans la console lors du lancement du CFN.

1. Modifiez le `AllowedVpcEndpointPrincipals` pour spécifier quel compte atteindra le point de terminaison VPC pour accéder au moteur d'exécution AWS Blu Age, à l'aide des commandes suivantes :

```
AllowedVpcEndpointPrincipals:
    Description: 'comma-separated list of IAM users, IAM roles, or AWS accounts'
    Default: 'apptest.amazonaws.com'
    Type: String
```

2. Modifiez la valeur SnapshotPrimaryDb des SnapshotBlusamDb variables et SnapshotJicsDb le nom des instantanés. Obtenez également les noms des instantanés auprès de RDS après leur création à l'étape précédente.

```
SnapshotPrimary:
    Description: 'Snapshot DB cluster for DB Primary'
    Type: String
    Default: 'snapshot-primary87d067b0'

  SnapshotBluesam:
    Description: 'Snapshot DB cluster for DB Bluesam'
    Type: String
    Default: 'snapshot-bluesam87d067b0'

  SnapshotJics:
    Description: 'Snapshot DB cluster for DB Jics'
    Type: String
    Default: 'snapshot-jics87d067b0'
```

> **Note**
>
> RDS ajoutera son propre suffixe aux noms des instantanés.

3. Fournissez votre paire de EC2 clés Amazon pour l' EC2 instance à l'aide de cette commande :

```
Ec2KeyPair:
    Description: 'ec2 key pair'
    Default: 'm2-tests-us-west-2'
    Type: String
```

4. Fournissez l'ID d'AMI que vous avez obtenu lors du processus d'enregistrement de l'AMI pour la variable BaaAmiId, en utilisant :

```
BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
```

```
    Default: 'ami-0d0fafcc636fd1e6d'
    Type: String
```

5. Indiquez le chemin du dossier Amazon S3 que vous avez utilisé à l'étape précédente pour enregistrer les fichiers produits, à l'aide de la commande suivante :

```
S3ApplicationFilesPath:
    Description: 'bucket name'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/post-produced-files'
```

6. Enfin, indiquez le chemin du dossier du s3- userdata-folder-path :

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/userdata'
```

- (Facultatif) Vous pouvez activer le mode HTTPS et l'authentification HTTP de base pour Tomcat. Bien que les paramètres par défaut fonctionneraient également.

> **Note**
>
> Par défaut, le mode HTTPS est désactivé et défini sur le mode HTTP dans le paramètre BacHttpsMode:

Par exemple :

```
BacHttpsMode:
    Description: 'http or https for Blue Age Runtime connection mode '
    Default: 'http'
    Type: String
    AllowedValues: [http, https]
```

- (Facultatif) Pour activer le mode HTTPS, vous devez changer la valeur en HTTPS et fournir l'ARN de votre certificat ACM en modifiant la valeur de la variable ACMCertArn :

```
ACMCertArn:
    Type: String
    Description: 'ACM certificate ARN'
```

```
    Default: 'your arn certificate'
```

- (Facultatif) L'authentification de base est désactivée par défaut avec le paramètre WithBacBasicAuthenticationdéfini sur false. Vous pouvez l'activer en définissant la valeur sur true.

```
WithBacBasicAuthentication:
    Description: 'false or true for Blue Age Runtime Basic Authentication '
    Default: false
    Type: String
    AllowedValues: [true, false]
```

7. Lorsque vous avez terminé la configuration, vous pouvez créer la pile à l'aide du CloudFormation modèle modifié.

Étape 6 : Test de l' EC2 instance Amazon AWS Blu Age

Exécutez manuellement le CloudFormation modèle pour créer l' EC2 instance Amazon AWS Blu Age de l' CardDemo application afin de vous assurer qu'elle démarre sans erreur. Ceci est fait pour vérifier que le CloudFormation modèle et tous les prérequis sont valides, avant d'utiliser le CloudFormation modèle avec la fonctionnalité de test d'application. Vous pouvez ensuite utiliser les tests d'applications pour créer automatiquement l' EC2 instance Amazon AWS Blu Age cible lors de la rediffusion et de la comparaison.

1. Exécutez la commande CloudFormation create stack pour créer l' EC2 instance Amazon AWS Blu Age, en fournissant le CloudFormation modèle m2- with-ba-using-snapshots -https-authentication.yml que vous avez modifié à l'étape précédente :

```
aws cloudformation create-stack --stack-name load-and-create-ba-snapshots —-
template-url https://apptest-ba-demo.s3.us-west-2.amazonaws.com/m2-with-ba-using-
snapshots-https-authentication.yml --capabilities CAPABILITY_NAMED_IAM --region us-
west-2
```

> **Note**
>
> N'oubliez pas de spécifier la bonne région dans laquelle l'AMI AWS Blu Age a été restaurée.

2. Assurez-vous que tout fonctionne correctement en recherchant dans la console l' EC2 instance Amazon en cours d'exécution. Connectez-vous-y à l'aide du gestionnaire de session.
3. Une fois connecté à l' EC2 instance Amazon, utilisez les commandes suivantes :

```
sudo su
cd /m2-anywhere/tomcat.gapwalk/velocity/logs
cat catalina.log
```

4. Assurez-vous qu'il n'y a aucune exception ou erreur dans le journal.
5. Vérifiez ensuite que l'application répond à l'aide de cette commande :

```
curl http://localhost:8080/gapwalk-application/
```

Vous verrez le message « L'application Jics est en cours d'exécution ».

Étape 7 : Valider que les étapes précédentes ont été effectuées correctement

Au cours des prochaines étapes, nous testerons les applications de modernisation AWS du mainframe pour rejouer et comparer les ensembles de données créés par l'application. CardDemo Ces étapes dépendent de la réussite de toutes les étapes précédentes de ce didacticiel. Validez les points suivants avant de continuer :

1. Vous avez créé avec succès l' EC2 instance AWS Blu Age on Amazon via le AWS CloudFormation modèle.
2. Le service Tomcat sur le AWS Blu Age sur Amazon EC2 est opérationnel, sans exception.

Lorsque l' EC2 instance est exécutée avec l' CardDemo application, effectuez les étapes suivantes sur la console de test des applications pour effectuer une réexécution et une comparaison des ensembles de données par lots.

Étape 8 : Création du scénario de test

Au cours de cette étape, vous créez le scénario de test qui sera utilisé pour comparer les ensembles de données créés dans l'application Card Demo.

1. Créez un nouveau scénario de test. Donnez-lui un nom et une description.

2. Spécifiez `CREASTMT.JCL` comme nom JCL.
3. Ajoutez les ensembles de données suivants à la définition du cas de test :

Nom	CCSID	RecordFormat	RecordLength
AWS.M2.CARDDEMO.STATEMNT.PS	« 037 »	FB	80
AWS.M2.CARDDEMO.STATEMENT.HTML	« 037 »	FB	100

> **Note**
>
> Le nom de votre JCL et les détails de votre jeu de données doivent correspondre.

Étape 9 : Création d'une suite de tests

1. Créez une nouvelle suite de tests et donnez-lui un nom et une description.
2. Ajoutez le scénario de test que vous avez créé à l'étape précédente à votre suite de tests.
3. Une fois la suite de tests créée, capturez les cas de test sur le mainframe et téléchargez les données de référence du mainframe dans AWS Application Testing.
4. Choisissez Créer une suite de tests.

Étape 10 : Création d'une configuration d'environnement de test

1. Créez une nouvelle configuration d'environnement de test et donnez-lui un nom et une description.
2. Ajoutez votre CloudFormation modèle. Vous pouvez également ajouter le nom et la valeur du paramètre d'entrée à partir de votre CloudFormation modèle.
3. Choisissez le service de modernisation AWS du mainframe AWS Blu Age non géré comme environnement d'exécution.

4. Ajoutez le nom de la variable de sortie pour le nom de l'ID de l'application AWS Mainframe Modernization, le nom de la variable de sortie pour l'ID de service du point de terminaison VPC, le nom de la variable de sortie pour le port du récepteur et le nom de la variable de sortie pour le nom. WebApp

   > **Note**
   >
   > Les noms de ces champs doivent correspondre aux noms des variables de sortie du CloudFormation modèle qui sera renvoyé par AWS Mainframe Modernization lors de la création de la pile.

5. (Facultatif) Choisissez le nom de la variable de sortie pour l'ARN de la tâche DMS (Database Migration Service) et l'emplacement de l'URI S3 DDL (Database Definition Language) de la base de données source.
6. (Facultatif) Personnalisez votre clé de service de gestion des clés (KMS). Pour plus d'informations, consultez Gestion de l'accès aux clés gérées par le client dans le Guide du développeur AWS Key Management Service .
7. Choisissez Créer une configuration d'environnement de test.

Étape 11 : Téléchargez vos données d'entrée dans la suite de tests

Au cours de cette étape, vous exécutez des scénarios de test sur la source. Pour ce faire :

1. Téléchargez et exécutez les ensembles de données issus de l'exécution de l'application sur le CardDemo mainframe.
2. Téléchargez le dossier décompressé dans votre compartiment Amazon S3. Ce compartiment Amazon S3 doit se trouver dans la même région que vos autres ressources de test d'applications.

   > **Note**
   >
   > Il doit y avoir deux fichiers dont les noms correspondent aux noms des ensembles de données passés dans le scénario de test précédent.

3. Sur la page de présentation de la suite de tests, cliquez sur le bouton Télécharger.
4. Sur la page Charger les données de référence, spécifiez l'emplacement Amazon S3 où vous avez chargé les ensembles de données obtenus à partir du mainframe source.

5. Choisissez Upload pour démarrer le processus de téléchargement.

> **Note**
>
> Attendez que l'enregistrement soit terminé avant de procéder à la rediffusion et à la comparaison.

Étape 12 : Rejouer et comparer

Exécutez la suite de tests et les scénarios de test dans l' EC2 environnement AWS AWS Blu Age on Amazon cible. Les tests d'applications captureront les ensembles de données produits par rediffusion et les compareront aux ensembles de données de référence enregistrés sur le mainframe.

1. Choisissez Replay et comparez. La création de la CloudFormation pile et l'exécution de la comparaison devraient prendre environ trois minutes.

Une fois que tout est terminé, vous devriez avoir des résultats de comparaison avec quelques différences créées intentionnellement dans le cadre de cette démonstration.

AWS Modernisation du mainframe, tests d'applications, ensembles de données pris en charge, pages de code

Utilisez le tableau suivant pour déterminer si l'identifiant de jeu de caractères codé (CCSID) de vos données est pris en charge lors des tests AWS d'applications. Si vos données utilisent un CCSID non pris en charge, nous vous recommandons de le convertir en un CCSID compatible ou de nous contacter pour obtenir de l'aide.

CCSID	Jeu de caractères	Description
37	IBM037, IBM-037, Cp037	Pays hôte : États-Unis, Canada (ESA), Pays-Bas, Portugal, Brésil, Australie, Nouvelle Zélande
273	IBM273, IBM-273, Cp273	Pays hôte : Autriche, Allemagne

CCSID	Jeu de caractères	Description
277	IBM277, IBM-277, Cp277	Pays hôte : Danemark, Norvège
278	IBM278, IBM-278, Cp278	Pays hôte : Finlande, Suède
280	IBM280, IBM-280, Cp280	Pays hôte : Italie
284	IBM284, IBM-284, Cp284	Pays hôte : Espagne, Amérique latine (espagnol)
285	IBM285, IBM-285, Cp285	Hôte : Royaume-Uni
297	IBM297, IBM-297, Cp297	Hôte : France
300	IBM-300	JAPAN DB EBCDIC
301	IBM-301	Données PC : base de données du Japon
437	IBM437, IBM-437, US-ASCII, ASCII, Cp437, US-ASCII	Données PC : PC Base USA, de nombreux autres pays
500	IBM500, IBM-500, Cp500	Hôte : Belgique, Canada (AS/400), Suisse, International Latin-1
720	IBM-720	MSDOS ARABE
737	IBM-737, x- IBM737	MDOS GREC
775	IBM775, IBM-775	MSDOS BALTIQUE
808	IBM-808	Données informatiques : cyrillique, Russie, avec euro
813	ISO-8859-7, _7 ISO8859	ISO 8859-7 : Grèce
819	ISO-8859-1, _1 ISO8859	ISO 8859-1 : Pays latin-1

CCSID	Jeu de caractères	Description
833	IBM-833	EBDIC CORÉEN
834	IBM-834, x- IBM834	DB EBCDIC CORÉEN
835	IBM-835	T-CHINESE DB EBCD
836	IBM-836	EBDIC S-CHINOIS
837	IBM-837	EBDIC S-CHINOIS
850	IBM850, IBM-850, Cp850	Données informatiques : pays latin-1
855	IBM855, IBM-855, Cp855	Données du PC : cyrillique
856	IBM-856, x-, Cp856 IBM856	Données du PC : hébreu
858	IBM00858, IBM-858, Cp858	Données informatiques : pays latin-1, avec euro
859	IBM-859	Données du PC : LATIN-9
860	IBM860, IBM-860	Données du PC : portugais
861	IBM861, IBM-861	Données informatiques : Islande
862	IBM862, IBM-862, Cp862	Données du PC : hébreu (migration)
863	IBM863, IBM-863	Données informatiques : Canada
865	IBM865, IBM-865, Cp865	Données informatiques : Den/ Norvège
866	IBM866, IBM-866, Cp866	Données informatiques : cyrillique, Russie

CCSID	Jeu de caractères	Description
867	IBM-867	Données informatiques : hébreu avec euro
870	IBM870, IBM-870, Cp870	Hébergeur : Latin-2 multilingue
871	IBM871, IBM-871, Cp871	Pays hôte : Islande
874	x- IBM874	Données du PC : thaï
875	IBM-875, x-, Cp875 IBM875	Pays hôte : Grèce
897	IBM-897	Données PC : Japon SB
912	ISO-8859-2, _2 ISO8859	ISO 8859-2 : Latin-2 multiling ue
915	ISO-8859-5, _5 ISO8859	ISO 8859-5 : cyrillique
916	ISO-8859-8, _8 ISO8859	ISO 8859-8 : Hébreu
918	IBM918, IBM-918, Cp918	Hôte : Urdu
920	ISO-8859-9, _9 ISO8859	ISO 8859-9 : Latin-5 (ECMA-128, Turquie TS-5881)
921	IBM-921, x-, Cp921 IBM921	Données informatiques : Lettonie, Lituanie
922	IBM-922, x-, Cp922 IBM922	Données informatiques : Estonie
923	ISO-8859-15, Cp923, _15_FDIS ISO8859	ISO 8859-15 : Latin-9
924	IBM-924	ISO 8859-15 : Latin-9
927	IBM-927	Données du PC : T-Chinese

CCSID	Jeu de caractères	Description
930	IBM-930, IBM93 x-0, Cp930	Katakana Host : SBCS étendu. Kanji Host : DBCS comprenant 4 370 caractères définis par l'utilisateur
932	IBM-932	Données du PC : Japan Mix
933	IBM-933, x-, Cp933 IBM933	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisa teur et 11 172 caractères Hangul complets
935	IBM-935, x-, Cp935 IBM935	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisa teur.
937	IBM-937, x-, Cp937 IBM937	Hôte : Extended SBCS. Hôte : DBCS comprenant 6204 caractères définis par l'utilisa teur
939	IBM-939, x-, Cp939 IBM939	Latin Host : SBCS étendu. Kanji Host : DBCS comprenan t 4 370 caractères définis par l'utilisateur.
942	IBM-942, IBM-942C, x-, x-IBM942 C, Cp942, Cp942C IBM942	Données du PC : SBCS étendu. Données PC : DBCS comprenant 1880 caractères définis par l'utilisateur

CCSID	Jeu de caractères	Description
943	IBM-943, IBM-943C, Shift_JIS , Windows-31j, Windows-9 32, x-, x-C, Cp943, Cp943C, IBM943 IBM943 MS932	Données du PC : SBCS. Données PC : DBCS pour environnement ouvert comprenant 1880 caractères IBM définis par l'utilisateur
947	IBM-947	T-CHINESE BIG-5
948	IBM-948, x-, Cp948 IBM948	Données du PC : SBCS étendu. Données PC : DBCS comprenant 6204 caractères définis par l'utilisateur
949	IBM-949, IBM-949C, x-, x-IBM949 C, Cp949, Cp949C IBM949	Code IBM KS - Données du PC : SBCS. Code IBM KS - Données PC : DBCS comprenant 1880 caractères définis par l'utilisateur
950	Grand 5, IBM-950, x-0, Cp950 IBM95	Données du PC : SBCS (IBM BIG5). Données PC : DBCS comprenant 13493 CNS, 566 caractères sélectionnés par IBM, 6204 caractères définis par l'utilisateur
951	IBM-951	Données du PC : IBM KS
954	EUC-JP, IBM-954, IBM-954C	G0 : JIS X201 Roman. G1 : EST X208-1990. G1 : JIS X201 Katakana. G1 : JIS X212
964	EUC-TW, IBM-964, x-, Cp964 IBM964	G0 : ASCII. G1 : plan CNS 11643 1. G1 : plan CNS 11643 2.

CCSID	Jeu de caractères	Description
970	EUC-KR, IBM97 x-0, Cp970	G0 : ASCII. G1 : KSC X5601-1989 comprenant 1880 caractères définis par l'utilisateur
971	IBM-971	EUC CORÉEN
1006	IBM-1006, x-006, Cp1006 IBM1	ISO-8 : ourdou
1025	IBM-1025, x-025, Cp1025 IBM1	Hébergeur : cyrillique multilingue
1026	IBM1026, IBM-1026, Cp1026	Hôte : Latin-5 (Turquie)
1027	IBM-1027	JAPON LATIN EBCD
1041	IBM-1041	Données PC : Japon
1043	IBM-1043	Données du PC : T-Chinese
1046	IBM-1046, IBM-1046S, x-046 IBM1	ARABE - PC
1047	IBM1047, IBM-1047	Hôte : Latin-1
1051	hp-roman 8	ÉMULATION HP
1088	IBM-1088	Données PC : Korea KS
1089	ISO-8859-6, _6 ISO8859	ISO 8859-6 : arabe
1097	IBM-1097, x-097, Cp1097 IBM1	Hôte : Farsi
1098	IBM-1098, x-098, Cp1098 IBM1	Données du PC : Farsi

CCSID	Jeu de caractères	Description
1112	IBM-1112, x-, Cp1112 IBM1112	Pays hôte : Lettonie, Lituanie
1114	IBM-1114	Données du PC : T-CH SB
1115	IBM-1115	Données du PC : S-CH Go
1122	IBM-1122, x-, Cp1122 IBM1122	Pays hôte : Estonie
1123	IBM-1123, x-, Cp1123 IBM1123	Hôte : Ukraine cyrillique
1124	IBM-1124, x-, Cp1124 IBM1124	8 bits : cyrillique, Biélorussie
1140	IBM01140, IBM-1140, Cp1140	Pays d'accueil : États-Unis, Canada (ESA), Pays-Bas, Portugal, Brésil, Australie, Nouvelle Zélande, avec euro
1141	IBM01141, IBM-1141, Cp1141	Pays hôte : Autriche, Allemagne, avec l'euro
1142	IBM01142, IBM-1142, Cp1142	Pays hôte : Danemark, Norvège, avec l'euro
1143	IBM01143, IBM-1143, Cp1143	Pays hôte : Finlande, Suède, avec l'euro
1144	IBM01144, IBM-1144, Cp1144	Pays hôte : Italie, avec euro
1145	IBM01145, IBM-1145, Cp1145	Pays hôte : Espagne, Amérique latine (espagnol), avec l'euro
1146	IBM01146, IBM-1146, Cp1146	Pays hôte : Royaume-Uni, avec euro

CCSID	Jeu de caractères	Description
1147	IBM01147, IBM-1147, Cp1147	Pays hôte : France, avec euro
1148	IBM01148, IBM-1148, Cp1148	Pays hôte : Belgique, Canada (AS/400), Suisse, International Latin-1, avec euro
1149	IBM01149, IBM-1149, Cp1149	Pays hôte : Islande, avec l'euro
1200	UTF-16BE	Unicode avec jeu de caractères 65535. En l'absence d'une marque d'ordre des octets (BOM), elle est supposée être UTF-16 BE (big-endian).
1202	UTF-16LE	UTF-16 LE avec IBM PUA
1204	UTF-16	UTF-16 avec IBM PUA
1208	UTF-8, UTF-8J, UTF8	Unicode avec jeu de caractères 65535. UTF-8.
1232	UTF-32BE	UTF-32 BE avec IBM PUA
1234	UTF-32LE	UTF-32 LE avec IBM PUA
1236	UTF-32	UTF-32 avec IBM PUA
1351	IBM-1351	OPEN DU JAPON
1362	IBM-1362	MS-WIN CORÉEN
1363	IBM-1363, IBM-1363C, fenêtre-949, MS949	Données du PC : MS Windows Korean SBCS. Données du PC : MS Windows Koran DBCS, y compris 11172 Hangul complet

CCSID	Jeu de caractères	Description
1364	IBM-1364	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisateur et 11 172 caractères Hangul complets
1370	IBM-1370	Données PC : SBCS étendu, avec euro. Données PC : DBCS comprenant 6204 caractères définis par l'utilisateur, avec euro
1371	IBM-1371	Hôte : SBCS étendu, avec euro. Hôte : DBCS comprenant 6204 caractères définis par l'utilisateur, avec euro
1375	Big 5-HKSC	Mixed Big-5 Ext pour HKSCS
1380	IBM-1380	Données du PC : S-CH Go
1381	IBM-1381, x-, Cp1381 IBM1381	Données du PC : SBCS étendu (IBM Go). Données PC : DBCS (IBM GB) comprenant 31 caractères sélectionnés par IBM, 1880 caractères définis par l'utilisateur
1382	IBM-1382	EUC CHINOIS
1383	EUC-CN, IBM-1383 GB2312, x-, Cp1383 IBM1383	G0 : ASCII. G1 : ensemble GB 2312-80
1385	IBM-1385	Données du PC : S-CH GBK

CCSID	Jeu de caractères	Description
1386	GBK, IBM-1386, Fenêtre-936, MS936	Données PC : S-Chinese GBK et T-Chinese IBM BIG-5. Données du PC : S-Chinese GBK
1388	IBM-1388	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisa teur
1390	IBM-1390	Katakana Host : SBCS étendu, avec euro. Kanji Host : DBCS comprenant 6205 caractères définis par l'utilisa teur
1399	IBM-1399	Latin Host : SBCS étendu, avec euro. Kanji Host : DBCS comprenant 4 370 caractère s définis par l'utilisateur, avec euro
5050	JIS0201, JIS0208, JIS0212, JIS0201, JIS0208, JIS0212	G0 : JIS X201 Roman. G1 : EST X208-1990. G1 : JIS X201 Katakana. G1 : JIS X212
5054	ISO-282 JP	TCP JAPONAIS
5346	Windows-1250, Cp1250	MS Windows : Latin-2, version 2 avec euro
5347	Windows-1251, Cp1251	MS Windows : cyrillique, version 2 avec euro
5348	Windows-1252, Cp1252	MS Windows : pays Latin-1, version 2 avec euro

CCSID	Jeu de caractères	Description
5349	Windows-1253, Cp1253	MS Windows : Grèce, version 2 avec euro
5350	Windows-1254, Cp1254	MS Windows : Turquie, version 2 avec euro
5351	Windows-1255, Cp1255	MS Windows : Hébreu, version 2 avec euro
5352	Windows-1256, Windows-1256s, Cp1256	MS Windows : arabe, version 2 avec euro
5353	Windows-1257, Cp1257	MS Windows : Baltic Rim, version 2 avec euro
5354	Windows-1258, Cp1258	MS Windows : vietnamien, version 2 avec euro
5488	GB18030	GB18030, données à 1 octet 030, données à 2 octets GB18 030, données GB18 à 4 octets
9030	IBM-838, Cp838	Hôte : Thai extended SBCS
9066	IBM-874, Cp874	Données du PC : SBCS étendu thaïlandais
9400	CESU-8	CESU-8 avec IBM PUA
25546	ISO-282 KR	TCP CORÉEN
33722	IBM-33722, IBM-33722C	IBMeucJP

Protection des données dans le cadre des tests d'applications de modernisation des AWS mainframes

Le modèle de responsabilité AWS partagée Le modèle de s'applique à la protection des données dans le AWS cadre des tests d'applications de modernisation des ordinateurs centraux. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

Nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer des utilisateurs individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Par conséquent, chaque utilisateur ne reçoit que les autorisations nécessaires à l'accomplissement de ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- Utilisez le protocole SSL/TLS pour communiquer avec les ressources. AWS Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-2 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS (Federal Information Processing Standard) disponibles, consultez Federal Information Processing Standard (FIPS) 140-2 (Normes de traitement de l'information fédérale).

Nous vous recommandons d'éviter d'utiliser des informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libres (par exemple, le champ Nom). Cela inclut lorsque vous travaillez avec des tests d'applications de modernisation du AWS mainframe ou autre Services AWS à l'aide de la console, de l'API ou AWS SDKs. AWS CLI Toutes les données que vous saisissez dans les balises ou les champs de texte libre utilisés pour les noms peuvent être utilisées pour les journaux de facturation ou de diagnostic. Si vous fournissez une URL vers un serveur externe, évitez d'utiliser les informations d'identification contenues dans l'URL pour valider votre demande auprès de ce serveur.

Données collectées lors des tests de l'application de modernisation AWS du mainframe

AWS Les tests d'applications de modernisation du mainframe collectent plusieurs types de données auprès de vous :

- `Resource definition`: La définition des ressources indique les données transmises aux tests d'applications lorsque vous créez ou mettez à jour une ressource de type scénario de test, suite de tests ou configuration de test.
- `Scripts for replay`: il s'agit de scripts transmis aux tests d'applications par rapport à votre application de modernisation AWS du mainframe.
- `Data for comparison`: Il s'agit d'ensembles de données ou de fichiers CDC (Database Change Data Capture) transmis à Application Testing à des fins de comparaison.

AWS Les tests d'applications de modernisation du mainframe stockent ces données de manière native dans. AWS Les données que nous collectons auprès de vous sont stockées dans un bucket Amazon S3 géré par les tests d'applications de modernisation du AWS mainframe. Lorsque vous supprimez une ressource, les données associées sont supprimées du compartiment Amazon S3.

Lorsque vous lancez un test pour effectuer une rediffusion afin de tester des charges de travail interactives, AWS Mainframe Modernization Application Testing télécharge le script dans un conteneur Fargate géré par Amazon ECS basé sur un stockage éphémère pour effectuer la rediffusion. Le fichier de script est supprimé une fois la rediffusion terminée et le fichier de sortie généré par le script est stocké dans le compartiment Amazon S3 géré par les tests d'applications sur votre compte. Le fichier de sortie de rediffusion est supprimé du compartiment Amazon S3 lorsque vous supprimez le test.

De même, lorsque vous lancez un test pour comparer des fichiers (ensembles de données ou modifications de base de données), AWS Mainframe Modernization Application Testing télécharge les fichiers dans un conteneur Fargate géré par Amazon ECS basé sur un stockage éphémère pour effectuer la comparaison. Les fichiers téléchargés sont supprimés dès que l'opération de comparaison est terminée. Les données de sortie de comparaison sont stockées dans le compartiment Amazon S3 géré par les tests d'applications dans votre compte. Les données de sortie sont supprimées du compartiment S3 lorsque vous supprimez le test.

Vous pouvez utiliser toutes les options de chiffrement Amazon S3 disponibles pour sécuriser vos données lorsque vous les placez dans le compartiment Amazon S3 utilisé par AWS Mainframe Modernization Application Testing pour comparer les fichiers.

Chiffrement des données au repos pour les tests d'applications de modernisation AWS du mainframe

AWS Les tests d'applications de modernisation du mainframe s'intègrent à AWS Key Management Service (KMS) pour fournir un chiffrement transparent côté serveur (SSE) sur toutes les ressources dépendantes qui stockent les données en permanence. Les exemples de ressources incluent Amazon Simple Storage Service, Amazon DynamoDB et Amazon Elastic Block Store. AWS Les tests d'applications de modernisation du mainframe créent et gèrent des AWS KMS clés de chiffrement symétriques pour vous. AWS KMS

Le chiffrement des données au repos par défaut permet de réduire les frais opérationnels et la complexité liés à la protection des données sensibles. Dans le même temps, il vous permet de tester des applications qui nécessitent une conformité stricte en matière de chiffrement et des exigences réglementaires.

Vous ne pouvez pas désactiver cette couche de chiffrement ni sélectionner un autre type de chiffrement lorsque vous créez des scénarios de test, des suites de tests ou des configurations de test.

Vous pouvez utiliser votre propre clé gérée par le client pour les fichiers de comparaison et les AWS CloudFormation modèles afin de chiffrer Amazon S3. Vous pouvez utiliser cette clé pour chiffrer toutes les ressources créées pour les tests dans le cadre des tests d'applications.

> **Note**
>
> Les ressources DynamoDB sont toujours chiffrées à l'aide d' Clé gérée par AWS un compte de service intégré aux tests d'applications. Vous ne pouvez pas chiffrer les ressources DynamoDB à l'aide d'une clé gérée par le client.

AWS Les tests des applications de modernisation du mainframe utilisent la clé gérée par le client pour les tâches suivantes :

- Exportation d'ensembles de données depuis Application Testing vers Amazon S3.
- Chargement des fichiers de sortie de comparaison vers Amazon S3.

Pour plus d'informations, consultez Clés gérées par le client dans le Guide du développeur AWS Key Management Service (langue française non garantie).

Création d'une clé gérée par le client

Vous pouvez créer une clé symétrique gérée par le client en utilisant le AWS Management Console ou le AWS KMS APIs.

Pour créer une clé symétrique gérée par le client

Suivez les étapes de la rubrique Création d'une clé symétrique gérée par le client dans le Guide du développeur AWS Key Management Service .

Stratégie de clé

Les stratégies de clé contrôlent l'accès à votre clé gérée par le client. Chaque clé gérée par le client doit avoir exactement une stratégie de clé, qui contient des instructions qui déterminent les personnes pouvant utiliser la clé et comment elles peuvent l'utiliser. Lorsque vous créez votre clé gérée par le client, vous pouvez spécifier une stratégie de clé.

Vous trouverez ci-dessous un exemple de politique clé délimitant l'accès ViaService qui permet aux tests d'applications d'enregistrer des données générées par des rediffusions et des comparaisons dans votre compte. Vous devez associer cette politique au rôle IAM lorsque vous appelez `StartTestRun` l'API.

Example

```
{
    "Sid": "TestRunKmsPolicy",
    "Action": ["kms:Decrypt", "kms:GenerateDataKey"],
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/TestRunRole"
    },
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": ["s3.amazonaws.com"]
        },
        "ForAnyValue:StringEquals": {
            "kms:EncryptionContextKeys": "aws:apptest:testrun"
        }
    }
}
```

Pour plus d'informations, consultez Gestion de l'accès aux clés gérées par le client dans le Guide du développeur AWS Key Management Service .

Pour plus d'informations sur le dépannage des clés d'accès, consultez le Guide du développeur AWS Key Management Service .

Spécification d'une clé gérée par le client pour les tests AWS d'applications de modernisation du mainframe

Lorsque vous créez une configuration de test, vous pouvez spécifier une clé gérée par le client en saisissant un ID de clé. Application Testing permet de chiffrer les données chargées dans le compartiment Amazon S3 pendant l'exécution du test.

- KEY ID — Identifiant de clé pour une clé gérée par le client. Saisissez un ID de clé, un ARN de clé, un nom d'alias ou un ARN d'alias.

Pour ajouter votre clé gérée par le client lorsque vous créez une configuration de test avec le AWS CLI, spécifiez le `kmsKeyId` paramètre comme suit :

```
create-test-configuration --name test \
--resources '[{
```

```
    "name": "TestApplication",
    "type": {
        "m2ManagedApplication": {
            "applicationId": "wqju4m2dcz3rhny5fpdozrsdd4",
            "runtime": "MicroFocus"
        }
    }
}]' \
--service-settings '{
    "kmsKeyId": "arn:aws:kms:us-west-2:111122223333:key/05d467z6-c42d-40ad-
b4b7-274e68b14013"
}'
```

AWS Modernisation du mainframe | Test des applications | Contexte de chiffrement

Un contexte de chiffrement est un ensemble facultatif de paires clé-valeur qui contient des informations contextuelles supplémentaires sur les données.

AWS KMS utilise le contexte de chiffrement comme données authentifiées supplémentaires pour prendre en charge le chiffrement authentifié. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, AWS KMS lie le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez inclure le même contexte de chiffrement dans la demande.

AWS Modernisation du mainframe | Test des applications | Contexte de chiffrement

AWS Les tests d'applications de modernisation du mainframe utilisent le même contexte de chiffrement dans toutes les opérations AWS KMS cryptographiques liées à un test, où la clé `aws:apptest:testrun` et la valeur sont l'identifiant unique du test.

Example

```
"encryptionContext": {
        "aws:apptest:testrun": "u3qd7uhdandgdkhhi44qv77iwq"
}
```

Utilisation du contexte de chiffrement pour la surveillance

Lorsque vous utilisez une clé symétrique gérée par le client pour chiffrer votre test, vous pouvez également utiliser le contexte de chiffrement dans les enregistrements d'audit et les journaux pour

identifier la manière dont la clé gérée par le client est utilisée lors du téléchargement de données sur Amazon S3.

Surveillance de vos clés de chiffrement pour les tests d'applications de modernisation du AWS mainframe

Lorsque vous utilisez une clé gérée par le AWS KMS client avec vos ressources de test d'applications de modernisation du AWS mainframe, vous pouvez l'utiliser AWS CloudTrailpour suivre les demandes que les tests d'applications de modernisation du AWS mainframe envoient à Amazon S3 lors du téléchargement d'objets.

Chiffrement en transit

Pour les cas de test qui définissent les étapes à suivre pour tester les charges de travail transactionnelles, les échanges de données entre l'émulateur de terminal géré Application Testing exécutant vos scripts Selenium et les points de terminaison de l'application AWS Mainframe Modernization ne sont pas chiffrés pendant le transit. AWS Les tests d'applications de modernisation du mainframe permettent AWS PrivateLink de se connecter au point de terminaison de votre application afin d'échanger des données en privé sans exposer le trafic sur l'Internet public.

AWS Les tests d'applications de modernisation du mainframe utilisent le protocole HTTPS pour chiffrer le service. APIs Toutes les autres communications dans le AWS cadre des tests d'applications de modernisation des mainframes sont protégées par le VPC de service ou le groupe de sécurité, ainsi que par le protocole HTTPS.

Le chiffrement de base en transit est configuré par défaut, mais ne s'applique pas aux tests de charge de travail interactifs basés sur le `TN3270` protocole.

Comment fonctionnent les tests d'applications de modernisation des AWS mainframes avec IAM

Avant d'utiliser IAM pour gérer l'accès aux tests d'applications de modernisation des AWS mainframes, découvrez quelles fonctionnalités IAM peuvent être utilisées dans le cadre des tests des applications de modernisation des AWS mainframes.

Fonctionnalités IAM que vous pouvez utiliser dans le cadre des tests d'applications de modernisation AWS du mainframe

Fonctionnalité IAM	AWS Soutien aux tests d'applications de modernisation du mainframe
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Non
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition de politique	Oui
ACLs	Non
ABAC (étiquettes dans les politiques)	Oui
Informations d'identification temporaires	Oui
Transmission des sessions d'accès (FAS)	Oui
Fonctions du service	Non
Rôles liés à un service	Non

Pour obtenir une vue d'ensemble de la façon dont les tests d'applications de modernisation du AWS mainframe et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez les AWS services compatibles avec IAM dans le guide de l'utilisateur IAM.

Politiques basées sur l'identité pour les tests d'applications de AWS modernisation des mainframes

Prend en charge les politiques basées sur l'identité : oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles

ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité, car celle-ci s'applique à l'utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples de politiques basées sur l'identité pour les tests d'applications de AWS modernisation des mainframes

Pour consulter des exemples de politiques basées sur l'identité destinées à tester des applications de modernisation du AWS mainframe, consultez. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Politiques basées sur les ressources dans le cadre des tests d'applications de modernisation des AWS mainframes

Prend en charge les politiques basées sur les ressources : non

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal intercompte à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont différents Comptes AWS, un administrateur IAM du compte sécurisé doit également accorder à l'entité

principale (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions stratégiques pour les tests d'applications de modernisation des AWS ordinateurs centraux

Prend en charge les actions de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de stratégie portent généralement le même nom que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour consulter la liste des actions de test des applications de modernisation du AWS mainframe, voir Actions définies par les tests des applications de modernisation du AWS mainframe dans la référence d'autorisation de service.

Les actions de stratégie dans le AWS cadre des tests d'applications de modernisation des mainframes utilisent le préfixe suivant avant l'action :

```
apptest
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "apptest:CreateTestCase",
```

```
      "apptest:StartTestRun"
]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `List`, incluez l'action suivante :

```
"Action": "apptest:List*"
```

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Ressources relatives aux politiques pour les AWS tests d'applications de modernisation des ordinateurs centraux

Prend en charge les ressources de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Vous pouvez restreindre l'accès à des ressources spécifiques de test des applications de modernisation des AWS mainframes en les utilisant ARNs pour identifier la ressource à laquelle s'applique la politique IAM. Pour plus d'informations sur le format de ARNs, consultez Amazon Resource Names (ARNs) dans le Références générales AWS.

Par exemple, un scénario de test d'une application de modernisation d'un AWS mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:apptest:regionId:accountId:testcase/service-generated-unique-identifier"
```

Une configuration de test d'application de modernisation d'un AWS mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:apptest:regionId:accountId:testconfiguration/service-generated-unique-identifier"
```

Une suite de tests d'applications de modernisation d'un AWS mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:apptest:regionId:accountId:testsuite/service-generated-unique-identifier"
```

Un test d'application de modernisation d'un AWS mainframe exécuté possède l'ARN suivant.

```
"Resource": "arn:aws:apptest:regionId:accountId:testrun/service-generated-unique-identifier"
```

Les actions de test des applications de modernisation AWS du mainframe ne prennent pas toutes en charge les autorisations au niveau des ressources. Pour les actions qui ne prennent pas en charge les autorisations au niveau des ressources, vous devez utiliser le caractère générique (*).

Les actions de test des applications de modernisation AWS du mainframe suivantes ne prennent pas en charge les autorisations au niveau des ressources.

```
ListTestCases
ListTestConfigurations
ListTestRuns
ListTestSuites
ListTagsForResource
```

Pour consulter la liste des types de ressources de test des applications de modernisation du AWS mainframe et leurs caractéristiques ARNs, consultez la section Ressources définies par les tests d'applications de modernisation du mainframe AWS dans la référence d'autorisation de service. Pour

savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par les tests d'applications de modernisation du mainframe AWS.

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Clés de condition des politiques pour les tests AWS d'applications de modernisation des ordinateurs centraux

Prend en charge les clés de condition de politique spécifiques au service : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR` opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Pour consulter la liste des clés de condition relatives aux tests d'applications de modernisation des AWS ordinateurs centraux, consultez la section Clés de condition pour les tests des applications de modernisation des AWS ordinateurs centraux dans la référence d'autorisation de service. Pour

savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez Actions définies par les tests d'applications de modernisation du mainframe AWS.

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Listes de contrôle d'accès (ACLs) dans les tests AWS d'applications de modernisation des mainframes

Supports ACLs : Non

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Contrôle d'accès basé sur les attributs (ABAC) avec tests d'applications de modernisation du AWS mainframe

Prise en charge d'ABAC (balises dans les politiques) : Oui

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux entités IAM (utilisateurs ou rôles) et à de nombreuses AWS ressources. L'étiquetage des entités et des ressources est la première étape d'ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l'identification du principal correspond à celle de la ressource à laquelle il tente d'accéder.

L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur ABAC, consultez Définition d'autorisations avec l'autorisation ABAC dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

Utilisation d'informations d'identification temporaires dans le cadre des AWS tests d'applications de modernisation du mainframe

Prend en charge les informations d'identification temporaires : oui

Certains Services AWS ne fonctionnent pas lorsque vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, y compris celles qui Services AWS fonctionnent avec des informations d'identification temporaires, consultez Services AWS la section relative à l'utilisation d'IAM dans le guide de l'utilisateur d'IAM.

Vous utilisez des informations d'identification temporaires si vous vous connectez à l' AWS Management Console aide d'une méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS l'aide du lien d'authentification unique (SSO) de votre entreprise, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Passage d'un rôle utilisateur à un rôle IAM (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide de l' AWS API AWS CLI or. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour y accéder AWS. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Sessions d'accès direct pour les tests AWS d'applications de modernisation du mainframe

Prend en charge les sessions d'accès direct (FAS) : oui

Lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal

appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

> **Important**
>
> Ces jetons permettent à AWS Mainframe Modernization Application Testing d'accéder aux données des clients sans votre accord explicite ; par exemple, AWS Mainframe Modernization Application Testing fournit les résultats des tests au bucket Amazon S3 d'un client sans obtenir l'autorisation explicite du client. Vous devrez peut-être mettre à jour toute documentation de conformité en conséquence.

Rôles de service pour les AWS tests d'applications de modernisation des mainframes

Prend en charge les rôles de service : Non

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

Rôles liés aux services pour les tests d'applications de AWS modernisation du mainframe

Prend en charge les rôles liés à un service : non

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés au service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Refactorisation automatique des applications avec Blu Age AWS

Le refactoring automatisé avec AWS Blu Age fournit une end-to-end solution pour la migration et la modernisation de vos applications mainframe. Les étapes du processus de refactorisation sont les suivantes :

- Analyser l'inventaire
- Analyser les dépendances
- Transformez automatiquement le code
- Capturez et gérez les scénarios de test

Vous pouvez effectuer les étapes précédentes dans l'outil Blu Insights, disponible via l'authentification unique depuis la console AWS Mainframe Modernization. Pour plus d'informations sur Blu Insights, consultez la documentation Blu Insights.

Lorsque vous êtes satisfait du code source transformé, il est temps de passer à AWS, où vous allez effectuer les étapes suivantes :

- Créez et déployez l'application refactorisée.
- Déployez et surveillez votre application dans AWS Mainframe Modernization.

AWS Blu Age propose différentes options d'exécution adaptées aux différents scénarios de déploiement et aux préférences opérationnelles. Il s'agit notamment des environnements d'exécution gérés et non gérés, chacun ayant son propre ensemble de fonctionnalités et de cibles de déploiement.

Pour une présentation complète des options AWS Blu Age Runtime disponibles, y compris les versions gérées et non gérées, les cibles de déploiement et leurs caractéristiques respectives, consultez the section called “AWS Options d'exécution de Blu Age” la documentation.

Ce guide vous aidera à comprendre les différences entre les options d'exécution et à choisir celle qui convient le mieux à votre projet de modernisation.

Rubriques

- AWS Sorties de Blu Age

- AWS Concepts de Blu Age Runtime
- Configurer la configuration pour AWS Blu Age Runtime
- AWS Blue Age Runtime APIs
- Configurer AWS Blu Age Runtime (non géré)
- Modifiez le code source avec Blu Age Developer IDE
- AWS FAQ sur Blu Age

AWS Sorties de Blu Age

AWS Le moteur Blu Age propose plusieurs versions parmi lesquelles vous pouvez choisir. Cette page présente le fonctionnement de la gestion des versions de AWS Blu Age, les modifications apportées à chaque version, les instructions de mise à niveau des versions, la manière dont les mises à jour de AWS Blu Age sont communiquées aux clients et le cycle de vie de ces versions.

La the section called "AWS Versionnage de Blu Age" page fournit des informations détaillées sur les versions et explique comment chaque version peut être identifiée par versions majeures et mineures. La the section called "AWS Notes de publication de Blu Age" page contient des notes de publication détaillées pour chaque version majeure et mineure. the section called "AWS Failles de sécurité dans Blu Age"Cette page explique comment AWS Blu Age gère les vulnérabilités et expositions courantes (CVE). the section called "Mise à niveau de AWS Blue Age"détaille les instructions de mise à niveau pour les versions de AWS Blu Age. Et the section called "AWS Cycle de vie de Blu Age" inclut tous les détails concernant les dates de fin de vie (EOL) des versions majeures de AWS Blu Age Runtime.

Rubriques

- AWS Versionnage de Blu Age
- AWS Options d'exécution de Blu Age
- AWS Notes de publication de Blu Age
- AWS Failles de sécurité dans Blu Age
- Instructions de mise à niveau pour AWS Blu Age
- AWS Cycle de vie de Blu Age

AWS Versionnage de Blu Age

Les produits AWS Blu Age Transformation et Runtime sont versionnés à l'aide d'un schéma conforme au semver (versionnement sémantique). Pour déployer votre application, vous devez utiliser la

version d'exécution correspondante compatible avec votre code modernisé. Si vous avez des questions sur la version à utiliser, contactez votre responsable de livraison AWS Blu Age.

Versions

Chaque version est identifiée par un **`[Major].[Minor].[Patch]`**modèle. Par exemple, avec la version AWS Blu Age Runtime`4.1.0`, la version principale est 4, la version mineure est 1 et la version patch est 0.

Nous avons l'intention de publier de nouvelles versions mineures de AWS Blu Age Runtime tous les mois, ainsi que de nouvelles versions majeures lorsque des modifications importantes sont apportées au produit ou à ses dépendances.

Pour plus de détails sur les nouvelles fonctionnalités disponibles dans chaque version, consultezthe section called "AWS Notes de publication de Blu Age".

Pré-versions alpha

Chaque pré-version alpha est identifiée par un **`[Major].[Next_Minor].0-alpha.[pre-release]`**schéma. Par exemple, dans la version préliminaire`4.2.0-alpha.1`, les modifications disponibles dans `alpha.1` seront publiées dans la prochaine version `4.2.0` mineure.

Les pré-versions alpha sont fréquemment des versions de courte durée destinées et disponibles pour une itération rapide au cours des projets de modernisation. Il n'existe pas de cadence de publication fixe pour les nouvelles versions préliminaires d'Alpha, et elles sont mises à disposition au fur et à mesure de leur développement et de leurs tests.

Pour plus d'informations sur le contrôle de version, les mises à niveau et le support, consultezCycle de vie des composants.

> **Important**
>
> Les versions préliminaires d'Alpha ne doivent être utilisées que pendant la phase du projet de modernisation et non pour la production ou les charges de travail critiques.

AWS Options d'exécution de Blu Age

AWS Blu Age propose trois types d'options d'exécution pour répondre aux différentes étapes de votre parcours de modernisation et à vos besoins opérationnels. Cette page décrit chaque option, ses caractéristiques, ses cas d'utilisation et la manière d'y accéder.

Runtime non géré

Avec AWS Blu Age Runtime (non géré), vous pouvez déployer vous-même votre application modernisée Compte AWS , ce qui vous permet de gérer votre propre infrastructure. Cette option fournit à la fois des versions préliminaires et préliminaires, ce qui vous donne la flexibilité d'utiliser tous les composants techniques nécessaires pour exécuter votre application modernisée comme vous le souhaitez. Vous pouvez choisir entre des versions stables pour les environnements de production ou des versions préliminaires à des fins de test et de développement.

L'environnement d'exécution non géré est déployé et géré par le client, ce qui permet de mieux contrôler l'environnement d'exécution. Il fournit des fonctionnalités de refactorisation automatique et convient aux scénarios de déploiement personnalisés.

Utilisation

Le runtime non géré convient aux environnements de test et de production, et est particulièrement utile lorsqu'une personnalisation spécifique de l'environnement d'exécution est requise.

Comment accéder

Pour demander l'accès aux artefacts d'exécution non gérés, consultez la section Intégration de AWS Blu Age Runtime.

Déploiement

AWS Blu Age Runtime (non géré) peut être déployé sur :

- Amazon EC2
- Amazon ECS sur Amazon EC2
- Amazon EKS sur Amazon EC2
- Amazon ECS géré par AWS Fargate

Le déploiement sur Amazon EC2 peut être effectué directement dans l'instance ou via une application conteneurisée Docker, ce qui est la méthode préférée lorsque vous utilisez Amazon ECS ou Amazon EKS.

Pour obtenir des instructions de déploiement détaillées, consultez la documentation relative à la configuration d'AWS Blu Age Runtime (non géré).

Runtime géré

Grâce à la gestion de AWS Blu Age Runtime, vous pouvez déployer votre application modernisée dans un environnement géré par AWS qui simplifie votre expérience. Vous n'avez donc pas besoin de gérer l'infrastructure sous-jacente qui exécute votre application modernisée.

Le runtime géré comprend une infrastructure gérée par AWS, avec des mises à jour et des correctifs automatiques. Il permet des opérations et une maintenance simplifiées. Seules les versions finales sont disponibles dans le runtime géré, ce qui garantit la stabilité et la fiabilité de vos environnements de production.

Utilisation

L'environnement d'exécution géré est idéal pour les environnements de production et convient particulièrement aux situations dans lesquelles vous préférez une approche directe de la gestion du temps d'exécution.

Comment accéder

Pour accéder à la version gérée de AWS Blu Age Runtime, il vous suffit d'accéder aux AWS services suivants :

- Amazon S3
- AWS Modernisation du mainframe

Avec les autorisations de compte appropriées, vous pouvez interagir de manière fluide avec l'environnement d'exécution géré. Cet accès rationalisé vous permet de tirer parti de toutes les fonctionnalités de Blu Age Runtime tout en bénéficiant AWS des services gérés.

Déploiement

Pour savoir comment configurer et utiliser le Managed Runtime, consultez la documentation Configurer le Managed Runtime pour AWS Blu Age.

Developer Runtime (BluInsights boîte à outils)

Le Developer Runtime est accessible depuis BluInsights Toolbox. Il est conçu pour les phases de développement et de test et est fréquemment mis à jour avec les dernières fonctionnalités. Ce runtime inclut à la fois les versions Release et les versions préliminaires d'Alpha, permettant aux développeurs d'accéder à des versions stables ainsi qu'à des fonctionnalités de pointe. Son objectif

principal est de prendre en charge des activités de test ou de développement spécifiques dans un environnement de développement local, généralement à partir d'un IDE. Il est important de noter que ce temps d'exécution est limité à 2 heures d'utilisation, ce qui le rend adapté aux sessions de développement ciblées plutôt qu'aux déploiements à long terme ou en production.

Utilisation

Le Developer Runtime est idéal pour les projets de développement et de modernisation initiaux, ainsi que pour les itérations rapides et les tests d'applications modernisées.

Comment accéder

L'accès à la BluInsights boîte à outils est fourni dans le cadre de votre engagement dans le cadre de votre projet AWS Blu Age. Le Developer Runtime est disponible via les requêtes de la boîte à outils AWS Blu Age. Une fois approuvé, vous aurez accès à des compartiments S3 spécifiques : `s3://toolbox-dev-runtime-<region>`

Ces buckets sont disponibles dans les régions us-east-1 et us-east-2. Pour utiliser le compartiment disponible, ajoutez la région au nom du compartiment (par exemple,`s3://toolbox-dev-runtime-us-east-1`).

Pour obtenir des instructions détaillées sur la façon de demander l'accès et de configurer les autorisations nécessaires, consultez la documentation AWS Blu Age Runtimes dédiée au développement et à la configuration spéciale.

Déploiement

Pour déployer le Developer Runtime, procédez comme suit :

1. Répertoriez les versions disponibles en utilisant AWS CLI :

```
aws s3 ls s3://toolbox-dev-runtime
```

2. Choisissez une version spécifique et listez son contenu :

```
aws s3 ls s3://toolbox-dev-runtime/[version]/
```

3. Téléchargez l'artefact d'exécution :

```
aws s3 cp s3://toolbox-dev-runtime/[version]/gapwalk-[version]-dev.tar.gz
```

4. Extrayez et configurez le runtime en fonction des exigences de votre projet.

Pour obtenir des instructions de déploiement et des directives d'utilisation plus détaillées, consultez la documentation relative aux environnements d'exécution AWS Blu Age Runtimes destinés aux développeurs et aux applications spéciales.

> **Note**
>
> Assurez-vous de disposer des autorisations de lecture S3 nécessaires Compte AWS pour accéder à ces compartiments. Vous trouverez un exemple de politique IAM dans la documentation AWS Blu Age Runtimes destinée aux développeurs et aux applications spéciales.

AWS Notes de publication de Blu Age

Cette section contient les notes de publication de AWS Blu Age Runtime and Modernization Tools à partir de la version 3.5.0, les plus récentes en premier, organisées par numéro de version.

> **Note**
>
> Pour les notes de publication antérieures à ce document, contactez les services de livraison de AWS Blu Age. Pour plus d'informations sur les dernières fonctionnalités de Blu Insights, consultez les versions de Blu Insights.

Rubriques

- Notes de mise à jour 4.8.0
- Runtime version 4.8.0
- Moteur de transformation AWS Blu Age 4.8.0
- Notes de mise à jour 4.7.0
- Runtime version 4.7.0
- Moteur de transformation AWS Blu Age 4.7.0
- Notes de mise à jour 4.6.0
- Runtime version 4.6.0
- AWS Moteur de transformation Blu Age 4.6.0
- Notes de mise à jour 4.5.0
- Runtime version 4.5.0

- AWS Moteur de transformation Blu Age 4.5.0
- Notes de mise à jour 4.4.0
- Runtime version 4.4.0
- AWS Moteur de transformation Blu Age 4.4.0
- Notes de mise à jour 4.3.0
- Runtime version 4.3.0
- Outils de modernisation, version 4.3.0
- Notes de mise à jour 4.2.0
- Runtime version 4.2.0
- Outils de modernisation, version 4.2.0
- Notes de mise à jour 4.1.0
- Runtime version 4.1.0
- Outils de modernisation, version 4.1.0
- Notes de mise à jour 4.0.0
- Runtime version 4.0.0
- Outils de modernisation, version 4.0.0
- Notes de mise à jour 3.10.0
- Runtime version 3.10.0
- Outils de modernisation, version 3.10.0
- Notes de mise à jour 3.9.0
- Runtime version 3.9.0
- Outils de modernisation, version 3.9.0
- Notes de mise à jour 3.8.0
- Runtime version 3.8.0
- Outils de modernisation, version 3.8.0
- Notes de mise à jour 3.7.0
- Runtime version 3.7.0
- Outils de modernisation, version 3.7.0
- Notes de mise à jour 3.6.0
- Runtime version 3.6.0

- Outils de modernisation, version 3.6.0
- Notes de mise à jour 3.5.0
- Runtime version 3.5.0
- Outils de modernisation, version 3.5.0

Notes de mise à jour 4.8.0

Date de sortie : 23 avril 2025

Cette version de AWS Blu Age Runtime and Transformation Engines introduit des mises à jour clés pour améliorer les capacités et les performances de transformation des bases de données :

- Transformation des bases de données IBM IMS — Ajout de la prise en charge de la transformation des bases de données IBM IMS en notre solution Blu Age personnalisée JHDB.
- Fonction de cache en mémoire — Ajout d'une fonctionnalité de cache en mémoire qui permet aux utilisateurs de mettre en cache des données en lecture seule dans la mémoire, améliorant ainsi les performances des programmes gourmands en données.

Nous avons testé cette version du AWS Blu Age Runtime avec le stack suivant.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 22.11.0
	Npm 10.9.0
	Angulaire 19.1.3
couche de service	Spring Boot 3.4.2
	Spring Core 6.2.2
	Session de printemps 3.4.1
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 15.10

	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Runtime version 4.8.0

ZoS

Nouvelles fonctionnalités

- Ajout de la prise en charge de la définition d'en-têtes personnalisés pour les transactions sécurisées via la propriété YML. `gapwalk-application.security.customAllowedHeaders` Cette propriété est associée à une `gapwalk-application.identity` propriété ayant une valeur`oauth`.
- Ajout d'une fonctionnalité permettant aux clients de reconstruire les métadonnées de Blusam en fonction des enregistrements réels de la table de base de données Blusam
- Connexions JHDB sécurisées grâce à l'intégration AWS Secrets Manager

Améliorations

- COBOL
  - Prise en charge améliorée de la condition AT END sur l'instruction de fichier READ
  - Prise en charge améliorée du transfert des littéraux numériques signés vers des champs alphanumériques sans déplacer le signe
  - Prise en charge améliorée de l'instruction UNSTRING pour gérer les conversions de champs non numériques en champs numériques et conserver les faibles valeurs lorsque le délimiteur est introuvable et qu'aucune valeur n'est déplacée vers d'autres variables
  - Amélioration de l'opération de liaison d'adresses pour les entités de liaison dynamique
  - Opérations de concaténation de chaînes améliorées grâce à la prise en charge des octets Shift In/Out (SOSI)
  - Support amélioré pour l'instruction COBOL ACCEPT avec DAY-OF-WEEK le code source
  - Ajout du support pour SYSIN et SYSOUT en tant que valeurs SPECIAL NAMES
  - Amélioration de la prise en charge de la concaténation de chaînes pour gérer le caractère d'alimentation du formulaire
- JCL - IDCAMS

  - Ajout de la prise en charge de la suppression des générations GDG à l'aide d'un nom de base sans le paramètre GDG
  - Support amélioré pour la gestion du paramètre VOLUME en tant que variante du paramètre VOLUMES standard pour l'instruction DEFINE CLUSTER
- JCL - SPORT
  - Ajout du support pour SYMNAMES avec la commande JOINKEYS
- JCL - IEBGENER
  - Amélioration de la prise en charge par IEBGENER du contenu SYSIN non valide en introduisant des contrôles de validation et un retour automatique à ICEGENER, garantissant ainsi un fonctionnement continu même avec des instructions de contrôle incorrectes
- JCL - INFUTILB
  - Gestion améliorée des clauses unload INTO et FROM
- JCL - DSNUTILB
  - Ajout du support pour le type de données DOUBLE PRECISION
- JCL - Divers
  - Amélioration des configurations de fichiers de remplacement utilisant les noms DD dans la procédure enfant issue de la tâche
  - DSNTIAUL/INFUTILB - Personnalisation améliorée des octets de remplissage SYSREC pour l'utilitaire de déchargement
  - Ajout de la prise en charge de l'option OUTDD pour la redirection de l'instruction DISPLAY
  - Support amélioré pour l'écriture de fichiers séquentiels par blocs fixes. Lorsque le LRECL du programme COBOL est différent du LRECL JCL, le LRECL JCL a priorité.
  - Ajout de la prise en charge des attributs DCB du catalogue pour le fichier SORTOUT, mais uniquement pour les ensembles de données permanents
  - Support amélioré pour la transformation conditionnelle par étapes JCL
  - Support amélioré pour ACCEPT avec SYSIN et les lignes dépassant la taille cible avec des espaces de fin
  - Prise en charge améliorée des instructions SET pour gérer les commandes sans inclure le nom de variable, la commande définie à l'intérieur inclut un fichier membre qui devrait être accessible dans la JCL et la commande contient des commentaires en ligne
  - Ajout du support pour conserver les paramètres « ScriptContext » au niveau de la tâche dans les métadonnées du point de contrôle JCL et le contexte de la tâche à utiliser au redémarrage

- Blusam
  - Amélioration de l'insertion d'enregistrements sur les grands ensembles de données indexés de Blusam lors de l'écriture après le chargement d'un ensemble de données vide
  - Performances améliorées pour les grands ensembles de données en activant un mécanisme de préchauffage et en introduisant une fenêtre de prélecture optionnelle pour les enregistrements, ainsi qu'un stockage local pour les index et les pages
  - Prise en charge améliorée de l'ensemble de données d'exportation pour gérer un ensemble de données plus important avec AWS Key Management Service
- SQL
  - Support amélioré pour le type de données JDBC BLOB
- Intégration des files d'attente de messages
  - Ajout de la prise en charge de l'identifiant utilisateur sur la commande MQ PUT lorsque XA est activé
  - Ajout de la prise en charge des connexions MQ simultanées lorsque XA est activé
- IMS
  - Ajout d'une propriété YML `jhdb.keepParent` pour contrôler si les appels IMS ISRT remplissent les champs _parent et _logicalparent dans la base de données lorsqu'ils sont définis sur true
  - Robustesse améliorée en supprimant la file de messages en cas de code de transaction non valide

AS400

Nouvelles fonctionnalités

- Ajout d'une fonctionnalité de cache en mémoire qui permet aux utilisateurs de mettre en cache des données en lecture seule dans la mémoire, améliorant ainsi les performances des programmes gourmands en données.

Améliorations

- CL
  - Amélioration de la prise en charge du paramètre QRYSLT dans la commande OPNQRYF pour gérer le mot clé RANGE, pour analyser la valeur lorsque les colonnes de chaînes ont des valeurs purement numériques et pour gérer les valeurs de chaîne vides

- Ajout du support pour la commande FTP SENDPASV
- Support amélioré pour la formation de requêtes dans la commande DSPDBR exécutée à l'aide de Qcmdexc
- Amélioration de la prise en charge de l'exception MONMSG pour la propager correctement dans la pile d'exécution
- Copie d'enregistrements améliorée entre les tables avec différents nombres de colonnes et types de données sur CPYF
- Priorité de surveillance des messages améliorée : les messages de niveau commande ont désormais correctement la priorité sur ceux de niveau procédure
- Support amélioré pour la commande CPYF afin de prendre en compte la bibliothèque définie par IVLIB BLU4
- Support amélioré de la MonMsgs commande CHKOBJ pour gérer le cas où le fichier n'existe pas dans la bibliothèque

- Screen
  - Gestion améliorée des BigDecimal valeurs de plus de 15 chiffres en construisant la valeur manuellement puis en la transformant en un nombre à afficher au lieu de laisser le nombre être arrondi
- Misc
  - Support amélioré pour l'option UserSpace « Replace as NO » lorsqu'elle existe UserSpace déjà
  - Fonctionnalité BLU4 IVLIB améliorée pour prendre en charge à la fois la récupération et l'écriture d'enregistrements dans des bibliothèques remplacées
  - Prise en charge améliorée des champs de tableaux multidimensionnels à l'aide des opérations DIM et SURCERS
  - Prise en charge améliorée des faibles valeurs transmises en tant que valeurs de champ clés dans le fonctionnement du DAO
  - Suppression du conflit sur les instructions MONMSG pour les messages pouvant être traités par un moniteur local et un moniteur global
  - Paramètre amélioré FileStatus sur Duplicate Record Insert
  - Réduction des conflits de verrouillage de l'espace utilisateur et du risque d'échec de création en utilisant d'abord le registre pour vérifier si l'espace utilisateur existe
  - Support amélioré pour readPriorEqual () sauter des enregistrements après suppression ()
  - Mécanisme de verrouillage des dossiers amélioré sous contrôle des engagements

- Gestion améliorée de l'instruction RPG RETURN dans les sous-programmes
- Amélioration de la prise en charge des informations utilisateur dans SharedContext

Moteur de transformation AWS Blu Age 4.8.0

ZoS

Améliorations

- COBOL
  - Génération améliorée du mode imprimante pour plusieurs fichiers.
  - Support amélioré pour SPECIAL-NAMES afin de gérer les points de fin facultatifs et les points multiples.
  - Support amélioré de COPY avec les instructions relatives path /REPLACE OFF/COPY - REPLACING.
  - Ajout du support pour COL en tant que nom de champ valide.
  - Analyse améliorée pour gérer les espaces après SPECIAL-NAMES, X-COMPUTER et DECLARATIVES.
  - Ajout du support pour S01 à S05 et C01 à C12 en tant que noms de paragraphes valides.
  - Ajout du support pour la fonction LENG en tant qu'alias de la fonction LENGTH.
  - Amélioration de l'analyse des copybooks lorsque la valeur de configuration de la cobolMaxCode largeur diffère de celle de leur inclueur COBOL
  - Support amélioré pour l'analyse des expressions avec virgule et les opérations, par exemple MIN (A * B, (3 + 1)/2, 3 + 4).
  - Ajout du support pour IS INITIAL PROGRAM dans la clause PROGRAM ID.
  - Ajout de la prise en charge de la condition de niveau 88 sur l'instruction SEARCH ALL.
  - Ajout du support pour la section REPORT.
- SQL
  - Ajout du support pour la transformation DDL à partir de la base de données IBM IMS.

AS400

Améliorations

- CL

  - Support amélioré de CL en définissant la transformation en JAVA comme support par défaut
  - Amélioration de la transformation en script groovy en ajoutant l'utilisation d'un métamodèle
  - Les mots clés CONTINUE et FOR ont été ajoutés à la liste des mots clés réservés
- DDS
  - Support amélioré des champs DSPF définis par rapport au précédent
  - Support amélioré pour le champ de caractères DDS avec le mot-clé VARLEN
  - Transformation/génération améliorée de l'entité LF DAOs en générant le réglage explicite de champs renommés dans les méthodes de conversion et de mise à jour JPAFrom Vo.
- RPG
  - Support amélioré du paramètre de mot-clé PREFIX pour gérer les cas où les valeurs n'ont pas de guillemets
  - Gestion améliorée des appels de méthodes imbriqués dans les index de tableau
- COBOL400
  - Amélioration de la génération d'entités d'enregistrement importées depuis un DSPF avec la directive COPY DDS-ALL-FORMATS
- SQL
  - Amélioration de la transformation des clauses SQL WHERE avec des conditions OR non incluses.

Notes de mise à jour 4.7.0

Date de sortie : 12 mars 2025

Cette version de AWS Blu Age Runtime and Transformation Engines apporte des mises à jour clés aux dépendances principales : AWS Blu Age Runtime a été mis à niveau pour utiliser la dernière version 3.4.2 de Spring Boot, et les applications frontales modernisées utilisent désormais la dernière version 19 d'Angular.

Nous avons testé cette version du AWS Blu Age Runtime avec la pile suivante. D'autres versions peuvent également être compatibles.

Composant	Version testée
Java	Java 17

Couche de présentation	Nœud JS 22.11.0
	Npm 10.9.0
	Angulaire 19.1.3
couche de service	Spring Boot 3.4.2
	Spring Core 6.2.2
	Session de printemps 3.4.1
Couche de persistance	Spring State Machine 4.0.0
	Moteur PostgreSQL 14
Oracle 21c	
Serveur d'application	Apache Tomcat 10.1.17

Runtime version 4.7.0

ZoS

Nouvelles fonctionnalités

- Ajout d'une nouvelle propriété YML `dataSimplifier.doubleFloatingType` pour définir le format à utiliser lors du codage et du décodage des nombres à virgule flottante - Les valeurs autorisées sont `IEEE_754` (par défaut)/`HFP`(pour le format hexadécimal à virgule flottante)
- Ajout du support de la position de mise à l'échelle décimale P sur l'image numérique

Améliorations

- COBOL
  - Support amélioré pour l'instruction INSPECT avec les clauses AFTER INITIAL et BEFORE INITIAL
  - Prise en charge améliorée des types numériques édités avec conversion d'octets optimisée améliorant l'efficacité globale du traitement des données
  - Ajout du support pour FUNCTION MIN & MAX avec comparaison alphanumérique

  - Support amélioré pour FUNCTION INTEGER pour les valeurs négatives
- PL/I
  - Support amélioré de l'instruction PUT avec le jeu de données FBA/VBA SYSPRINT
- JCL - SPORT
  - Ajout du support de l'option OUTREC SQZ avec les sous-paramètres SHIFT, LEAD, TRAIL, MID, PREBLANK et PAIR
  - Attributs DCB de catalogage améliorés pour le fichier SORTOUT
- JCL - DSNUTILB
  - Support amélioré pour le type de données DECIMAL EXTERNAL
  - Support amélioré pour la condition NULLIF liée aux valeurs/positions d'une autre colonne
  - Support amélioré pour les colonnes BYTEA
- JCL - INFUTILB
  - Support amélioré pour les mots clés CURRENT [DATE/TIME/TIMESTAMP] dans l'opération UNLOAD
- JCL - Support amélioré pour le code de retour qualifié de l'appel PROCs
- Traitement des données
  - Logique de comparaison numérique améliorée lors du traitement de champs à valeur vide redéfinis de types alphanumériques à des types numériques, garantissant une évaluation correcte par rapport à zéro
- Blusam
  - Prise en charge améliorée de la commande START pour gérer les recherches partielles par clé à l'aide de segments de la clé primaire, offrant ainsi des capacités de récupération d'enregistrements plus flexibles
- SQL
  - Ajout du support pour le paramètre d'entrée transmis dans la clause ON
  - Ajout du support pour la fonction REPEAT
  - Support amélioré pour l'instruction UPDATE avec une affectation impliquant un nom de colonne
  - Gestion améliorée de SQLCODE et SQLSTATE lors de l'extraction de curseurs sur plusieurs lignes
  - Ajout du support de la fonction DB2 XML pour la cible DB2 de base de données
- Intégration des files d'attente de messages

  - Gestion des erreurs améliorée dans les opérations MQ GET, fournissant une indication d'état plus claire MQCC_FAILED pour les corps de message vides
  - Support amélioré pour les opérations MQ PUT grâce à la gestion des MQRFH2 en-têtes dans le contexte des transactions XA
  - Prise en charge améliorée des secrets d'AWS Blu Age Runtime afin de gérer des propriétés supplémentaires pour JMS MQ
  - Capacité de configuration du pool de connexions JMS MQ améliorée
- Misc
  - Intégration du système - Ajout de la prise en charge de l'environnement de schéma dans les appels de programme DFSRRC00
  - Compatibilité ajoutée pour les transactions IMS lorsque Blusam est désactivé
  - Prise en charge améliorée de la end-of-file condition sur un fichier séquentiel afin de l'aligner sur l'ancienne règle « L'EOF est considéré comme une lecture infructueuse »

AS400

Nouvelles fonctionnalités

- Ajout de la prise en charge de la commande RUNSQLSTM pour exécuter des requêtes SQL stockées dans un fichier source externe

Améliorations

- CL
  - Ajout d'un mécanisme de verrouillage des fichiers via les commandes ALCOBJ/DLCOBJ
  - Amélioration de la fonctionnalité de commande CPYF afin de préserver l'intégrité de la séquence d'enregistrements lors de la copie de fichiers
  - Commande CPYF améliorée pour prendre en charge la sortie sous forme de fichier plat
  - Gestion améliorée des paramètres pour ajuster automatiquement la longueur des données lors des appels de programmes
  - Prise en charge améliorée des paramètres pour gérer la référence de plage
  - Ajout du support pour MONMSG CPF21 05 sur la commande DLTDTAARA
  - Ajout du support de la bibliothèque QTEMP pour la commande CHKOBJ sur DATAAREA

  - Support amélioré pour la commande ADDLFM, améliorant les capacités de gestion des membres de fichiers logiques
  - Ajout de la prise en charge des noms de zones de données génériques sur la commande DLTDTAARA
  - Ajout du support à SBMJOB pour gérer le passage des commutateurs via le paramètre SWS
  - Amélioration de la logique d'écriture de la zone de données en plaçant les appels de lecture et d'écriture dans un bloc synchronisé
  - Gestion améliorée de la simultanéité des zones de données pour empêcher les écritures simultanées grâce à la mise en œuvre d'un mécanisme de verrouillage granulaire, éliminant ainsi les erreurs lors des opérations parallèles
  - Ajout de la prise en charge du mot clé *ALL dans le paramètre DTAARA pour la commande RTVDTAARA
- Screen
  - Amélioration de la couleur d'un attribut blanc dans le front-end
- RPG
  - Support amélioré pour le fonctionnement de TESTB afin de mieux gérer les différents codages
  - Support amélioré pour le fonctionnement d'EDITC avec les codes d'édition « Y » et « P »
  - Support amélioré pour EDITCODE pour les codes K, Q et Z
  - Implémenter la prise en charge du support des ExtProc mots clés Prototype sur la procédure locale avec des paramètres
  - Paramètres gérés transmis par valeur pour les procédures externes
- COBOL
  - Instruction SORT améliorée pour gérer la bibliothèque ciblée
  - Tronqué une partition lorsque le DAO associé est ouvert en mode OUTPUT
  - Ajout du support pour le mot clé PREFIX
- DataQueue
  - Ajout d'options de configuration de keystore et de truststore pour les données RabbitMQ et les files d'attente de messages hybrides, permettant une messagerie sécurisée lors des déploiements de production
  - Performances de redémarrage du serveur optimisées en éliminant la vérification de l'existence des files d'attente de données redondantes

  - Architecture rationalisée grâce à la suppression DataQueueRegistry de DataQueueConnectionProvider composants
  - Suppression simplifiée des files d'attente de données grâce à un traitement direct sans validation du registre
  - Processus de création de files d'attente amélioré en garantissant une création et une queue-to-exchange liaison d'échange appropriées
- Misc
  - Ajout de la validation de la taille des demandes de transaction pour éviter la surcharge du système liée à des demandes de plage de valeurs excessives
  - Gestion améliorée des serrures lors du nettoyage des ressources
  - Amélioration du mécanisme de verrouillage DAO pour empêcher le verrouillage d'un enregistrement lorsque le fichier est ouvert en mode INPUT
  - Performances optimisées pour le mappage dynamique des entités
  - Gestion améliorée du contexte de persistance JPA pour améliorer les performances des opérations DAO
  - Support amélioré des zones de données en ajoutant des colonnes d'horodatage pour gérer les métadonnées des enregistrements
  - Fiabilité améliorée de la zone de données grâce à la mise en œuvre d'opérations de lecture-écriture synchronisées pour éviter les problèmes d'accès simultanés
  - Support amélioré pour DateHelper #moveDate afin de soutenir date-to-timestamp les conversions
- DDS
  - Prise en charge améliorée des conditions Select/Omit combinée à des opérations synthétiques telles que SST ou CONCAT
- SQL
  - Optimisation du SQLExecutor générateur pour gérer plus efficacement la récupération partielle des enregistrements, en particulier lors de la récupération de plusieurs lignes avec des limites d'extraction supérieures à une

Moteur de transformation AWS Blu Age 4.7.0

ZoS

Améliorations

- COBOL
  - Ajout du support pour l'instruction CODE-SET
  - Ajout du support pour l'option NOT END OF PAGE from WRITE statement
  - Ajout du support pour une nouvelle syntaxe sur ALTER
  - Ajout du support du remplacement de plusieurs lignes pour l'instruction COPY REPLACING
  - Analyse améliorée du mot clé CHARACTERS impliqué dans une définition de clause différente
  - Amélioration de la prise en charge des clés secondaires impliquées dans la section d'entrée-sortie lorsque les noms des clés sont dupliqués mais ont des parents différents
  - Prise en charge améliorée de la condition IF grâce à la comparaison de grands champs par rapport aux espaces
  - Amélioration de la résolution statique des références aux fichiers logiques en la rendant insensible aux majuscules et minuscules
  - Transformation de code améliorée pour générer automatiquement des champs implicites LINAGE-COUNTER pour chaque entrée FD contenant une clause LINAGE
  - Support amélioré pour la définition du cluster IDCAMS avec un nom entouré de simples guillemets
  - Support amélioré de l'instruction PERFORM pour le format de base, avec le format de phrase TIMES et le format impliquant la qualification de la section
  - Activation de la transformation de l'instruction SORT avec plusieurs fichiers GIVING et ajout d'une exception pour suivre ce cas lors de son exécution
  - Amélioration de la transformation des RENAMES lorsqu'ils sont spécifiés sur un groupe à l'aide de la clause REDEFINES
- PL/1
  - Support ajouté pour l'option DATA ou LIST pour PUT STRING
  - Ajout de la prise en charge du programme à entrées multiples

AS400

Améliorations

- CL
  - Amélioration de l'analyse des paramètres pour que QCMDEXC puisse exécuter ALCOBJ/DLCOBJ via QCMDEXC

  - Ajout du support pour la ElseIf déclaration sur CL
  - Ajout de la prise en charge de la fonction intégrée %SWITCH pour configurer les commutateurs de tâches.
- DDS
  - Support amélioré de l'enregistrement DSPF décrit par le programme pour gérer les enregistrements fictifs dans les spécifications d'entrée
  - Amélioration des conditions de la clause WHERE de génération DAO dans le cas d'un fichier CREATE INDEX SQLDDS
- RPG
  - Ajout du support pour l'initialisation de ZEROES INZ (*ZEROS) sur un entier non signé
  - Support amélioré de la commande SAUF avec un type de fichier de type Workstation et un index SQL
  - Support amélioré du type de données interne RPG pour les champs numériques définis en externe
  - Support amélioré du %ERROR intégré pour obtenir le statut d'erreur actuel après une opération sur un fichier
  - Support amélioré du mot clé de fichier COMMIT pour gérer les fichiers logiques multiformats
  - Traitement SELECT/OMIT amélioré pour les fichiers logiques multiformats
  - Amélioration de la gestion MOVEA pour définir un tableau avec des espaces, déplacer des tableaux fixes de type compressé, zoné et binaire vers un autre tableau du même type.
  - Gestion améliorée des tableaux pour Z-ADD *ZEROS et MOVEA *ZEROS
  - Opérations MOVE/MOVEL améliorées lors du déplacement à partir de combinaisons numériques et de caractères
  - Implémenter la prise en charge du support des ExtProc mots clés du prototype RPGLE sur la procédure locale avec des paramètres
  - Gestion améliorée de la constante figurative *HIVAL dans les instructions d'affectation
  - Amélioration de la prise en charge du mot clé PREFIX pour gérer le remplacement des caractères
- COBOL400
  - Support amélioré pour les opérations de REWRITE et de MISE À JOUR en mode d'accès dynamique
  - Ajout du support pour la clause ACCEPT Statement FOR dans CBLLE

  - Support amélioré du %STATUS intégré pour obtenir l'état actuel d'un fichier après une opération
- Misc
  - Transformation améliorée des programmes qui définissent des structures de données portant le même nom que l'un des champs du PF/LF spécifié
  - Génération améliorée de l'accès aux tableaux sur les variables kanji

Notes de mise à jour 4.6.0

Date de sortie : 24 janvier 2025

Nous avons testé cette version du AWS Blu Age Runtime avec la pile suivante. D'autres versions peuvent également être compatibles.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 22.11.0
	Npm 10.9.0
	Angulaire 18
couche de service	Spring Boot 3.3.5
	Spring Core 6.1.14
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Runtime version 4.6.0

ZoS

Améliorations

- COBOL
  - `WRITE ADVANCING`Fonctionnalités améliorées avec une précision améliorée pour l'écriture séquentielle de lignes de fichiers, prise en charge de contextes multiples (`BEFORE>`,`AFTER`, et utilisations implicites) et mise en œuvre complète des `PAGE` instructions
  - Support amélioré `FILLER` pour les cas où un remplisseur de table imbriqué est utilisé en groupe avec une table en tant qu'enfant
  - Accès amélioré à l'enfant dont les parents sont ambigus au sein d'un segment
  - Ajout du support pour le type d'édition numérique avec picture='-----'
  - Gestion améliorée de l'affichage des données de type BINAIRE
- PL/I
  - Conversion améliorée des valeurs littérales binaires dans les instructions d'affectation
- JCL — TRIER
  - Prise en charge améliorée `OVERLAY` des paramètres consécutifs dans la même `OUTFIL` instruction
- JCL — DSNUTILB
  - Mécanismes de chargement optimisés, permettant d'accélérer de 25 % les temps de récupération des données
  - Support amélioré pour les transactions XA pour les sources de données commerciales externes
- JCL — INFUTILB
  - UNLOAD - Ajout de la prise en charge du type de FLOAT8 données
- JCL — IDCAMS
  - Gestion optimisée des codes de retour pour `IDCAMS` les commandes
  - Ajout du support pour supprimer toutes les générations GDG en fonction du nom de base GDG
  - Ajout du support pour la suppression de fichiers sans `NONVSAM` paramètre
- JCL — Divers
  - Gestion améliorée des métadonnées de redémarrage par lots pour améliorer la gestion de l'état du flux de travail en mode redémarrage

- Blusam
  - Ajout du support du TTL pour le cache Blusam dans les implémentations Ehcache et Redis
  - Support amélioré pour `DEPENDING ON` le champ sur la description du fichier COBOL `FD` pour le fichier Blusam KSDS
  - Sécurité des threads améliorée dans les opérations de lecture de Redis Blusam pour l'exécution simultanée de plusieurs tâches
  - Création améliorée du schéma Blusam pour une meilleure robustesse en ce qui concerne les privilèges des utilisateurs de base de données
  - Rembourrage amélioré vers la droite sur le jeu de données d'entrée concaténé par blocs variables `READ`
- BAC
  - Ajout de la prise en charge de la création de jeux de données en mode multi-schémas, y compris une nouvelle colonne « Schéma » pour indiquer l'association de schémas pour chaque ensemble de données
- MFS
  - Propagation améliorée des informations utilisateur depuis le front-end vers le contexte partagé, garantissant une propagation correcte vers le contexte JHDB
  - Ajout de la prise en charge de l'en-tête d'information IBM MQ IMS sur les transactions XA
- SQL
  - `SQLCODE`Gestion améliorée pour définir 305 lors de la récupération du curseur lorsque toutes les valeurs de colonne sont NULL
  - Ajout de la prise en charge de `IN` la clause impliquant `OCCURS` un paramètre pour `WHERE` les conditions
  - Ajout de la prise en charge des instructions de table `DECLARE GLOBAL` temporaires
  - Support DB2 SQL étendu pour un format d'horodatage DB2 spécifique à minuit et 24 heures grâce à des conversions dédiées lors de l'exécution conformément au moteur de base de données ciblé
- Misc
  - Jeu de caractères IBM93 0 amélioré pour permettre aux caractères Unicode U+2014 et U+2015 de correspondre à X'44x4a' dans EBCDIC
  - TDQUEUE - Implémentation SQS refactorisée pour prendre en charge le multithreading

- Résolution du nom du jeu de données GDG améliorée pour permettre au client d'archiver des fichiers avec le même préfixe GDG (par exemple, `A.B.C.G0002V00` il s'agit du fichier actuel et `A.B.C.G0001V00.1236` d'un fichier d'archive)
- Amélioré `SQLConverter::toPgmDate/Time/Timestamp` pour aligner le calcul de la date en fonction de l'ancien format

AS400

Nouvelles fonctionnalités

- Ajout de la prise en charge des tables AS4 00 créées dynamiquement pour les fichiers plats et les entités dupliquées, permettant l'accès aux tables créées via des commandes CL telles que CRTPF, CRTDUPOBJ et CPYF
- Ajout d'un service pour prendre en charge la liste des bibliothèques via un registre qui gère la bibliothèque par défaut pour chaque table

Améliorations

- CL
  - CLRPFM - Amélioration de la gestion des membres lorsque la commande est appelée pour la bibliothèque QTEMP
  - SMBJOB - Support amélioré des paramètres `PARM` pour gérer les arguments construits dynamiquement
  - CPYFRMIMPF - Ajout du support pour les paramètres, et `TIMFMT ERRRCDFILE ERRRCDOPT`
  - CPYFRMIMPF - Amélioration de la prise en charge des valeurs alphanumériques de base de données contenant des guillemets simples
  - CPYF - Amélioration de la construction des requêtes de commande pour les fichiers à plusieurs `FROM` membres avec `TOMBR(*ALL)`
  - CPYF - Support amélioré pour gérer `FMTOPT` les paramètres pour `MAP DROP`
  - CPYTOIMPF - Amélioration de la prise en charge des paramètres par rapport `FROMFILE` à la gestion de la table `MEMBER`
  - RTVUSRPRF - Ajout du support pour les paramètres `RTNUSRPRF`

  - DSPDBR - Refonte de la commande pour qu'elle corresponde au comportement habituel attendu, qui consiste à imprimer des informations sur les vues existant sur une table, ainsi que sur la bibliothèque et le membre dont elles font partie
  - DSPFD - Support amélioré des paramètres `FILE`
  - DSPFD - Support amélioré de la `TYPE MBR` sortie des paramètres pour inclure des valeurs supplémentaires : mbfile, mblib, mbfcdt, mfccn
- Screen
  - Priorité de position du curseur améliorée pour `DSPATR(PC)`
  - Amélioration de la validation des champs d'enregistrement des sous-fichiers en ignorant la validation frontale des champs « protégés »
  - Support amélioré pour l'initialisation des enregistrements dans un poste de travail avec plusieurs champs de tableau partageant les noms des composants
  - Support amélioré pour les indicateurs de réponse dans les `DSPF` mots clés (`SFLMSG`,`SFLMSGID`, `CHANGE` et les touches de commande)
- RPG
  - Support du cycle de programme amélioré pour une meilleure gestion des champs lus à partir des fichiers primaires/secondaires
  - Ajout de la prise en charge du champ de contrôle partagé pour la lecture des fichiers principaux/secondaires
  - Méthode `%SUBST` intégrée améliorée pour gérer les champs à deux octets dans les instructions de comparaison
  - Support amélioré de l'indicateur ZERO pour le fonctionnement du MVR
- DDS
  - Ajout de la prise en charge des fichiers logiques multiformats avec un format d'enregistrement faisant référence au même enregistrement physique
- DataQueue
  - Gestion améliorée des interruptions de travail pour les tâches en attente de messages de file d'attente de données en nettoyant le consommateur pendant les interruptions
  - Migration de RabbitMQ vers Spring-AMQP pour une meilleure gestion des canaux et une meilleure mise à l'échelle des threads
- Misc

- SQLExecutorGénérateur amélioré pour prendre en charge les requêtes comportant plusieurs espaces blancs et accolades ouvertes sans espaces de début
- Support DAO amélioré pour gérer correctement le positionnement du curseur lors du changement de direction de lecture
- Initialisation affinée des clés après les opérations de récupération et de suppression afin de garantir la suppression correcte des enregistrements associés avant d'insérer des enregistrements mis à jour
- Code généré par le mappeur DAO optimisé pour améliorer les performances d'exécution

AWS Moteur de transformation Blu Age 4.6.0

ZoS

Améliorations

- COBOL
  - Analyse améliorée de la `RESERVE` clause avec un littéral facultatif `AREA/AREAS`
  - Support COBOL amélioré avec `DATA DIVISION` déclaration optionnelle, prenant en charge des cas de test rationalisés
  - Paragraphe sur les noms spéciaux amélioré en ajoutant la prise en charge de `ALPHABETSYMBOLIC`,, et de `CLASS` clauses, de commutateurs et de `FORMFEED` variables
  - Ajout du support en `SYSIN` tant que nom mnémotechnique dans les déclarations `ACCEPT`
  - Prise en charge améliorée des `PICTURE` clauses pour les symboles « $ », « 0 », « CR » et « DB » dans les calculs de taille `PIC` logique
  - Transformation `USE` des instructions améliorée pour plusieurs scénarios de fichiers
  - Transformation améliorée des `ALTER` relevés pour de multiples modifications
  - Ajout du support pour les constantes figuratives `ZERO` `HIGH-VALUE` `LOW-VALUES` dans la clause `delimited by`
- SQL
  - Transformation améliorée de la valeur par défaut pour la cible PostgreSQL afin de gérer les guillemets autour de la `CURRENT_TIMESTAMP` valeur par défaut
  - `WITH CHECK OPTION`Clause Handle des vues SQL

AS400

Améliorations

- DDS
  - Prise en charge améliorée des fichiers logiques multiformats qui font référence plusieurs fois au même enregistrement physique
- RPG
  - `MOVEMOVEL`Fonctionnement amélioré pour mieux gérer les zéros de rembourrage
  - Gestion améliorée des appels de fonctions imbriqués dans les évaluations et les conditions
- COBOL400
  - Ajout de la prise en charge de la transformation du `IN` mot clé dans `SELECT` les déclarations
  - Amélioration de la prise en charge des points manquants dans les entrées de description des données, conformément à la dernière version de COBOL où les points sont supposés être absents
  - Positionnement amélioré du curseur lors `REWRITE` des opérations
  - Support amélioré pour l'`START`instruction permettant de verrouiller l'enregistrement à la position actuelle du fichier
  - Support amélioré de la directive du compilateur `COPY DDS` pour générer toutes les structures de données d'entrée/sortie
- Misc
  - StateMachines - Transformation améliorée pour améliorer la déclaration des états composites conformément au paradigme stateless4j
  - Nettoyage amélioré des fichiers LF contenant des caractères spéciaux
  - Support amélioré du figuratif `*ALL` avec des valeurs hexadécimales
  - Support des `MOVE` opérations amélioré pour la conversion implicite des types numériques en types de caractères
  - Génération de beans de rapports optimisée pour trier par nom d'imprimante associé, évitant ainsi les doublons ou les conflits de noms
  - Support amélioré des mots clés `EXTFILE` combiné `USROPN` à la gestion de la valeur littérale et du format `libname/filename`

Notes de mise à jour 4.5.0

Date de sortie : 20 décembre 2024

Cette version de AWS Blu Age Runtime et AWS Blu Age Transformation Engines inclut les fonctionnalités clés suivantes.

- Support JCL — Il est désormais possible de générer et d'exécuter des scripts JCL à la volée dans le contexte d'exécution. Cette fonctionnalité accroît la flexibilité et l'automatisation du traitement des tâches par lots. Nous avons mis à jour le support des utilitaires JCL lors de l'exécution, avec un ensemble d'améliorations apportées à SORT, ICETOOL, INFUTILB et IDCAMS (voir les détails dans les sections suivantes). Ces améliorations offrent des capacités de traitement des données plus robustes et plus efficaces.
- Support des annuaires de liaison et des groupes d'activation pour les applications modernisées AS/400 — Les annuaires de liaison améliorent l'organisation du système en gérant les références de procédures exportées, tandis que les groupes d'activation rationalisent la gestion du contexte d'exécution. Ces fonctionnalités améliorent la précision et la fiabilité, une gestion robuste des ressources et des interactions système optimisées. Il en résulte un système plus résilient, organisé et efficace pour les applications AS4 00 modernisées.
- Mises à jour des dépendances : — Mise à jour de tous les frameworks frontaux (BAC/JAC et applications modernisées) vers les versions de support à long terme (LTS). La mise à jour d'Angular de la version 17 à la version 18 introduit un nouveau modèle de réactivité et rationalise la gestion des états, réduisant ainsi la complexité et améliorant la maintenance des applications pour les développeurs. Node.JS a également été mis à jour de la version 20 à la version 22.

Nous avons testé cette version du AWS Blu Age Runtime avec la pile suivante. D'autres versions peuvent également être compatibles.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 22.11.0
	Npm 10.9.0
	Angulaire 18

couche de service	Spring Boot 3.3.5
	Spring Core 6.1.14
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Runtime version 4.5.0

ZoS

Nouvelles fonctionnalités

- JCL — Ajout de la possibilité d'invoquer un traitement par lots à partir de programmes en ligne. Nous avons ajouté un service permettant de gérer les scripts JCL stockés dans un fichier dédié TDQueue lorsqu'un programme modernisé les génère à la volée. Ce service permet de reconstruire le message JCL, de le refactoriser en un script groovy et d'exécuter ce script groovy.
- ADABAS — Ajout du support pour le programme ADABAS. Grâce à cette prise en charge, le moteur d'exécution émule les commandes ADABAS pour l'accès à la base de données (disponible pour Oracle uniquement).

Améliorations

- COBOL
  - Support amélioré de l'instruction DISPLAY grâce à l'option NO ADVANCING
  - Précision accrue dans la gestion des signes de change, permettant à l'utilisateur de bénéficier d'une structure COBOL transformée plus précise
  - Support amélioré pour l'attribution de valeurs lors du déplacement d'un champ non signé vers un champ signé et vice versa
  - Support amélioré de la taille des blocs pour les fichiers GDG et les fichiers concaténés
- CICS
  - Ajout du support pour OpenStatus et EnableStatus des ensembles de données Blusam

  - Ajout du support pour la `SET DATASET` commande
- JCL — TRIER
  - Gestion améliorée de la taille d'enregistrement des ensembles de données
  - Amélioration de la prise en charge de l'`OUTFIL`instruction permettant de produire des fichiers de sortie contenant uniquement les enregistrements des fichiers d'entrée conformément aux valeurs spécifiées dans STARTREC et dans les options `ENDREC`
  - Support amélioré des `OVERLAY` déclarations
  - Amélioration de la prise en charge de l'`OUTREC`instruction permettant de gérer une variante de l'`EDIT`option. Nous soutenons désormais `EDIT(...)` en plus de `EDIT=(...)`
  - Ajout du support pour le modèle (p, m, f, OPERATOR, p2, m2, f2) dans les opérations arithmétiques
  - Vous pouvez utiliser la clause `DUMMY` file du `SORT` programme à partir d'une JCL pour gérer des fichiers d'entrée vides et bénéficier de la génération de fichiers vides.
- JCL — ICETOOL
  - Support amélioré de la `SORT FIELDS=COPY` déclaration par le biais du `SORT` programme
- JCL — INFUTILB
  - Support amélioré pour le calcul de la taille des enregistrements s'il n'est pas spécifié dans la JCL et si la propriété DFSIGDCB est désactivée
  - Clause UNLOAD with INTO améliorée pour DECIMAL en actualisant la précision et l'échelle en fonction des champs de la clause into
  - Méthode de formatage améliorée dans VarcharFormatter
  - Support amélioré avec une nouvelle option configurable qui permet aux utilisateurs de contrôler la manière dont les champs VARCHAR sont traités pendant le déchargement des données en ce qui concerne le comportement de remplissage, garantissant ainsi flexibilité et précision dans les processus d'extraction des données.
- JCL — IDCAMS
  - Suppression améliorée pour les fichiers dont le suffixe générique et le nom sont définis soit directement entre parenthèses, soit par de simples guillemets
  - Précision améliorée pour tirer parti du code de retour MAXCC
- JCL — IKJEFT01 - Ajout d'un indicateur de fonctionnalité `systin.encoding` (par défaut `=ASCII`) pour prendre en charge le codage spécifique du jeu de données de fichiers SYSTSIN

- JCL — Prise en charge améliorée de la propriété BDW pour un fichier de sortie généré dans une étape JCL et les étapes suivantes utilisent le même système de fichiers en entrée et `DISP=PASS`
- MF
  - Support amélioré pour l'en-tête à 2 octets pour le fichier Record Sequential
  - Gestion améliorée des codes de retour pour la commande DELETE
  - Ligne avancée d'écriture améliorée pour le fichier séquentiel d'enregistrement
- Redis
  - Initialisation améliorée du modèle Redis pour les points de contrôle JCL et les Jics TSQueues
  - Accessibilité et lisibilité améliorées des informations de verrouillage des enregistrements du jeu de données Redis
- SQL
  - Analyse améliorée de FOREIGN KEY avec la clause REFERENCES
  - Fourni une fonction de mise en cache extensible pour stocker les types graphiques existants d'origine dans la base de données, améliorant ainsi la traçabilité des données et facilitant le calcul graphique
  - Support d'analyse amélioré du modèle de requêtes SQL CASE WHEN dans les utilitaires d'exécution
  - Fonction intégrée améliorée de SQL Postgres Blu Age gwdecimal sur laquelle le moteur d'exécution s'appuie pour s'adapter à la fonction intégrée DB2 DECIMAL.
- Misc
  - Support amélioré pour l' NumericEditedType utilisation de l'opérande SIGN
  - Génération améliorée de la configuration de la source de données principale SpringBootLauncher dans l'application modernisée
  - Flexibilité améliorée pour séparer les journaux des applications du chemin lié à la tâche appelée.
  - Amélioration de la prise en charge de la valeur vide lors de la comparaison de champs provenant de NumberUtils
- FICHIER — Support amélioré des ensembles de données de blocs variables dans les fichiers sous-jacents
- MQ — Gestion améliorée des connexions MQ pour un environnement de haute disponibilité prêt à être utilisé
- Compatibilité améliorée avec les files d'attente MQ en ajoutant la prise en charge des clients autres que JMS afin d'améliorer le codage et la gestion des jeux de caractères

- Support amélioré pour les caractères de contrôle ANSI pour le fichier Ebcdic

AS400

Nouvelles fonctionnalités

- Ajout du support pour les données exportées dans les programmes liés
- Ajout d'un support spécifique à l'ILE pour la division par zéro

Améliorations

- COBOL400
  - Support amélioré de l'EOF dans l'état des fichiers
  - Augmenter le support de précision de l'instruction Cobol START pour prendre en charge le mot clé EQUAL dans la clause KEY IS
- CL
  - Ajout du support pour la commande UPDENVPARM
  - CRTPF - Ajout du support pour les tables accessibles avec une partition
  - RCVF - Support amélioré des fichiers logiques avec override
  - FTP - Prise en charge améliorée des fichiers d'E/S logiques avec OVRDBF, journal de sortie amélioré et prise en charge ajoutée des fichiers d'E/S dans le répertoire de travail
  - CPYFRMIMPF - Ajout du support pour les paramètres, `ERRRCDFILE` `TIMFMT` `ERRRCDOPT`
  - CPYF - Création de partitions QTEMP améliorée
  - CPYF - Ajout d'un message de surveillance lorsque le fichier *FROM est vide
  - OVRPRTF - Ajout du support pour les nouveaux paramètres :`PAGESIZE`,`OUTQ`,`DEV`,,`LIP`,`CPI`, `OVRFLOW` `LVLCHK` `FORMTYPE` `HOLD`
  - Précision accrue lors de l'utilisation du `FMTOPT` paramètre avec `MAP` et des `DROP` options dans la `CPYF` commande pour permettre de copier des données d'un fichier source contenant des colonnes supplémentaires vers un fichier cible
  - Précision accrue dans la gestion du mappage des modèles génériques de chemin du système de fichiers dans la commande `RMVLNK`
  - La commande `RMVM` (Supprimer la machine virtuelle) a été améliorée pour gérer les tables de `DROP` partition afin de garantir un nettoyage complet des ressources associées.
  - OPNQRYF - Support amélioré du paramètre *FILE pour la commande

  - Gestion CPF0000 implémentée pour englober tous les messages CPFx
  - CHGDTAARA - Ajout du support pour le mot clé *ALL pour modifier l'ensemble de la zone de données
- Screen
  - tables/subfile displaying by increasing accuracy for scrolling and position/priorityCurseur amélioré
  - `CHECK(RB)`Fonctionnalité améliorée `CHECK(RZ)` pour les champs non numériques et non signés
  - Support amélioré de la fonctionnalité d'écran d'aide pour les mots clés `HLPARA`
- RPG
  - Support amélioré des fonctionnalités intégrées `%SubDt`
  - Support amélioré pour les procédures utilisant une structure de données locale décrite de manière externe
  - Ajout de la prise en charge du paramètre de code d'erreur facultatif `QMHSNDPMQMHRMVPM` , et `QMHRCVPM`
  - Support amélioré de la méthode `%SUBST` intégrée pour mieux gérer les champs à double octet.
  - Ajout du support pour %TLOOKUP intégré et ses variantes (%TLOOKUPGE, %TLOOKUPGT, %TLOOKUPLE, %TLOOKUPLT)
- Zone de données
  - Support amélioré pour le fonctionnement OUT lorsque le facteur 1 est vide
  - Lectures simultanées améliorées sur la même zone de données
  - Ajout d'une variable `blu4iv.dtaara.library.disable` de configuration pour désactiver les bibliothèques pour la zone de données
  - Support étendu pour tirer parti des bibliothèques nommées grâce à des opérations de zone de données permettant à l'utilisateur de structurer l'emplacement des zones de données comme il le souhaite.
- DataQueue
  - Utilisation améliorée du canal RabbitMQ
  - RabbitMQ Consumer amélioré pour ne tenter d'annuler le client qu'une seule fois
  - Amélioration de la récupération de la file d'attente de données depuis RabbitMQ en n'essayant BasicGet que lorsque le temps d'attente est égal à 0
- Misc

- Espace utilisateur - Comportement amélioré lorsque plusieurs tâches tentent de récupérer le même espace utilisateur simultanément
- Prise en charge améliorée de la suppression des enregistrements non validés dans le cadre du contrôle des engagements
- Entité : prise en charge améliorée des omissions consécutives, car OMIT a une signification implicite `AND`
- Ajout de la prise en charge des cas de chameau dans les entités, les mappeurs et les setters pour gérer les coutumes nommées définies grâce à une refactorisation supplémentaire
- Amélioration de la propagation des informations utilisateur à partir des transactions de l'environnement AS4 00 via l'ensemble de l'application.
- Précision améliorée lors de la fin d'une tâche planifiée par Quartz en cas d'interruption.
- Amélioration du soutien au contrôle des engagements pour l'adapter à la portée du programme

AWS Moteur de transformation Blu Age 4.5.0

ZoS

Améliorations

- JCL - Génération de groovy améliorée pour le jeu de données KSDS basé sur l'analyse LISTCAT
- COBOL
  - Analyse améliorée de l'`COPY-REPLACING`instruction pour gérer le remplacement du sous-champ qualifié lorsque le nom de ce sous-champ est ambigu
  - Support amélioré pour la `SYSOUT` définition dans la `SPECIAL-NAMES` déclaration
  - Support amélioré des ZEROES figuratifs dans la déclaration `ADD n TO ZERO`
  - Amélioration de la prise en charge `REPLACE` des instructions pour traiter les problèmes multilignes en aplatissant les touches multilignes et les blocs de texte
  - Support amélioré pour les opérations arithmétiques ADD/SUBTRACT/MULTIPLY/DIVIDE avec clause `GIVING`
  - Support d'analyse initialisé de REPORT SECTION et de ses actions associées (INITIATE, TERMINATE, GENERATE report)
- Divers - Améliorez la génération et la robustesse des rapports météorologiques

AS400

Améliorations

- DDS
  - Support amélioré de la longueur implicite de type DATE
  - Support amélioré du stop-zero-suppression caractère sur le mot clé EDITWORD
  - Support amélioré du nom de colonne DESC car il s'agit d'un mot réservé dans la base de données
- RPG
  - Support amélioré du %TIME intégré
  - Génération améliorée d'instructions EVALR pour gérer l'affectation d'une valeur de chaîne à une variable de longueur plus courte avec un meilleur ajustement à droite
  - Analyse SQL améliorée autour du paramétrage des options
  - Support amélioré pour l'initialisation de PSDS dans les programmes NOMAIN RPGLE
  - Support amélioré du mot clé LIKE pour définir un champ numérique DDS comme étant compressé, quelle que soit sa description externe
  - Nettoyage amélioré des noms de fichiers en remplaçant « $ » par « DL »
  - Support amélioré du %SUBST intégré pour gérer les valeurs à double octet
- COBOL400
  - Écran - Support amélioré de l'enregistrement DSPF lors des opérations d'E/S
- CL
  - Renommage amélioré des noms de variables réservées
  - Amélioration de la prise en charge des conditions de sélection et d'omission pour gérer des fichiers de formats multiples
- Misc
  - Réduction du nombre d'entités dupliquées liées aux opérations sur les fichiers (EOF, FOUND, EQUAL)
  - Génération améliorée de fichiers JRXML pour QPRINT, une imprimante standard sur AS/400. Lorsqu'il est utilisé, le fichier JSON créé ne contiendra aucune référence au programme ou au fichier. Un seul fichier JRXML est généré (QPrint-QPrint.jrxml)
  - Amélioration de l'affichage d'informations supplémentaires sur les messages pour les composants affichant des messages provenant de la file d'attente du programme

Notes de mise à jour 4.4.0

Date de sortie : 13 novembre 2024

Cette version de AWS Blu Age Runtime and Transformation Engines met l'accent sur la mise à niveau des dépendances critiques et des technologies prises en charge tout en améliorant les performances de multiples fonctionnalités. Parmi les principales fonctionnalités et modifications apportées à cette version, citons :

- Mises à jour des dépendances : les applications de console (BAC et JAC) et les applications modernisées s'exécutent désormais sur Bootstrap 5. Le AWS Blu Age Runtime est désormais alimenté par le framework Spring Boot 3.3.5.
- Performances : amélioration des performances d'exécution des machines d'état (jusqu'à 10 fois plus rapide), grâce à une nouvelle implémentation qui surmonte la dégradation des performances après la mise à niveau de la bibliothèque Spring State Machine de la version 2.5.1 à la version 4.0.0. Cette mise à niveau n'était pas facultative car la version 2.5.1 n'était plus maintenue et contient les versions Critical et High CVEs. Il inclut une implémentation de machine d'état d'exécution sur la plate-forme vers une nouvelle bibliothèque, avec une implémentation de machine d'état légère et efficace, exempte de CVE, et offrant de meilleures performances globales.
- Simplification de l'accès à la base de données : refonte complète des composants utilisés pour accéder à la base de données DAOs, notamment les entités JPA, les entités DDS DataSimplifier et les mappeurs. Cette refonte a été motivée par la nécessité de mieux prendre en charge la fonctionnalité OVRDBF (Override Database File) courante dans les projets 00. AS4 Il permet de traiter un plus grand nombre de cas avec une architecture simplifiée pour le code généré.

Nous avons testé cette version du AWS Blu Age Runtime avec le stack suivant. D'autres versions de composants peuvent également être compatibles.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 18.18
	Npm 9,8
	Angulaire 17

couche de service	Spring Boot 3.3.5
	Spring Core 6.1.14
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 4.4.0

ZoS

Nouvelles fonctionnalités

- COBOL - Ajout du support pour l'instruction JSON GENERATE
- COBOL - Ajout du support pour les blocs de contrôle
- MF - Ajout du support pour la directive du compilateur FCDREG
- Blusam - Ajout d'une fonctionnalité relative aux ensembles de fichiers VSAM avec une implémentation basée sur le schéma de base de données - Seul PostgressQL est pris en charge
- Blusam - Ajout du support pour la gestion du TTL (Time to live) pour les éléments de données mis en cache par Blusam (moteur de cache Redis)
- JCL - IDCAMS - Ajout d'une nouvelle propriété `idcams.encoding.forced` pour forcer le jeu de caractères utilisé pour décoder la carte SYSIN
- JICS - La `jics.db.dataScriptLocation` propriété a été étendue de manière `application-main.yml` à accepter une liste de chemins de fichiers et de dossiers. L'ordre de la liste est important. Le premier fichier SQL est exécuté en premier et ainsi de suite. Lorsqu'un dossier est exécuté, les scripts SQL qu'il contient ne sont exécutés dans aucun ordre défini.
- Ajout du support de l'utilitaire CEE3 ABD

Améliorations

- Blusam - Amélioration du temps de chargement et de l'encombrement mémoire des grands ensembles de données existants vers Blusam pour les clients utilisant le moteur PostgreSQL (nous avons observé une multiplication par 8 de la vitesse de chargement pour les grands ensembles de données)
- Blusam - API exportDataSet toS3 améliorée avec support des informations d'identification
- Blusam - Amélioration du téléchargement de fichiers LISTCAT pour la création d'ensembles de données
- Blusam - Support amélioré pour Dynamic READ à l'aide d'une clé explicite
- Blusam - Amélioration de la logique du mécanisme d'écriture différée
- JCL - Support JES amélioré pour améliorer le verrouillage des fichiers lors d'une exécution parallèle
- JCL - Ajout du support pour les déclarations `INCLUDE MEMBER`
- JCL - DNSUTILB - Support amélioré de la clé dupliquée pour gérer les cas particuliers où la clé primaire contient des espaces
- JCL - DSNUTILB - Amélioré LoadTask pour optimiser les performances lors du chargement de données GRAPHIQUES
- JCL - INFUTILB - Ajout de la prise en charge de la fonction « `fetchsize` quand `chunksize` » n'est pas défini
- JCL - INFUTILB - Support amélioré pour les requêtes renvoyant un jeu de résultats vide
- JCL - INFUTILB - Robustesse améliorée lors du traitement des données dans CHUNK
- JCL - INFUTILB - Support amélioré pour le déchargement avec champ nullable
- JCL - INFUTILB - Support amélioré pour le type numérique
- JCL - INFUTILB - Déchargement amélioré pour les champs nullables
- JCL - SORT - Support amélioré pour la syntaxe OUTREC
- JCL - SORT - Analyse améliorée de l'instruction DATE1
- JCL - SORT - Support amélioré de la clause INREC PARSE avec RDW
- JCL - SORT - Mise en forme améliorée des champs à l'aide de masques d'édition
- JCL - SORT - Support amélioré de « SubString » dans OUTREC
- JCL - SORT - Support amélioré pour les cartes compatibles MF
- JCL - UNLOAD - Support amélioré de la taille des champs avec Postgresql
- JCL - IDCAMS - Amélioration des performances pour le chargement de fichiers (ensemble de données VSAM) grâce à l'introduction du mode en bloc

- PL/1 - Améliore la prise en charge du NumericEditedType formatage pour éviter les écarts d'échelle
- IMS - Support amélioré pour la colonne _right de la base de données IMS dans NodeSorter
- CICS - Commande améliorée `RECEIVE MAP` avec `SET` et sans `INTO`
- BMS - Support amélioré de la valeur initiale du champ
- SQL - Analyse DateTimeFormat syntaxique améliorée pour les modèles `ddMMMyy`
- COBOL - Amélioration de la prise en charge de NumericEditedType la valeur lorsque la virgule décimale n'est pas prise en compte lors de l'obtention de la valeur
- Support amélioré pour la lecture de champs de longueur variable dans un fichier séquentiel de lignes
- Prise en charge améliorée de l'héritage de la taille des enregistrements à partir du catalogue de jeux de données pour les fichiers GDG
- Support amélioré pour l'impression de rapports en permettant des lignes d'avancement personnalisables
- Initialisation améliorée des données d'enregistrement pour les fichiers à blocs variables (VB)

GS21

Nouvelles fonctionnalités

- Écran - Ajout du support pour les fichiers PSAM
- Écran - Ajout du support pour ATTR2
- Ajout du support pour l'écosystème AIM (Advanced Information Manager).
- Ajout du support PED dans AIM

Améliorations

- BitUtils Signatures améliorées à gérer RangeReference
- Support amélioré DummyFileConfiguration pour ajouter les attributs RecordSize/rdw/bdw/blksize/blkszlim
- Support amélioré de l'instruction VPOINT pour gérer le cas d'un enregistrement introuvable
- Robustesse accrue lors de l'accès au tableau d'octets d'enregistrements
- Mappage des caractères du jeu de caractères JEF amélioré

- Support amélioré pour la gestion des tableaux et des conditions dans le mappage JDBC
- Prise en charge améliorée des requêtes SQL dans les différentes instructions NDB, meilleure gestion des variations de syntaxes SQL à l'aide de constantes pour chaque partie d'une requête SQL.
- Support amélioré pour que le GS21 PackedType dernier grignotage soit C, D ou F pour la validation numérique
- Écran - Support amélioré pour ACSAPI et DefaultPsamController pour SPA et ENTER
- Écran - Support amélioré des verbes ACSAPI et NDB

AS400

Nouvelles fonctionnalités

- Ajout du support pour les fichiers de base de données au format multi-enregistrement
- Refonte du cadre d'accès à la base de données AS4 00
  - Fonctionnalités améliorées en matière de remplacement de fichiers
  - Suppression des composants obsolètes et réduction de la complexité
  - Rationalisation du code généré par les anciens programmes
  - Composant DAOCycle Manager intégré au plugin Blu4IV, nous permettant de tirer parti des fonctionnalités AS4 spécifiques à 00 de notre environnement d'exécution personnalisé.
- JOB - Support amélioré pour la gestion des tâches (Quartz) afin d'ajouter la possibilité d'interrompre un travail/un groupe de tâches. Ajout d'un point de terminaison d'API REST pour interrompre une tâche avec l'identifiant d'exécution spécifié (unique pour chaque tâche puisqu'il s'agit d'une clé primaire). En cas d'interruption réussie, le moteur d'exécution met à jour le statut de la tâche sur « INTERROMPU ».
- Ajout du support pour le programme utilitaire CEERAN0
- Ajout du support pour le mode passif. Ajout du YAML `configuration gapwalk-application.cl:ftpservice:passive` pour activer le mode passif
- Ajout d'une fonctionnalité permettant de créer des sessions QTEMP et de retarder le nettoyage QTEMP
- Ajout du support pour la fonctionnalité de compilation BNDDIR pour définir des dépendances explicites entre les programmes
- Ajout de la prise en charge du mécanisme des groupes d'activation

Améliorations

- CL - Commande RMVMSG améliorée sur la file de messages du programme pour gérer le mot clé *PREV
- CL - Support amélioré pour les remplacements dans OPNQRYF
- CL - Ajout du support pour les paramètres MSGLEN et SECLVLLEN pour la commande RTVMSG
- CL - Amélioration de la prise en charge de CRTDUPOBJ pour gérer les cas où NEWOBJ n'est pas transmis et prise en charge ajoutée des noms de table génériques
- CL - Support amélioré du FTP pour gérer les paramètres GET, RMTSYS et BINARY
- CL - Amélioration des performances des requêtes CLRPFM et ajout d'une option permettant d'utiliser TRUNCATE au lieu de DELETE
- CL - SBMJOB amélioré pour gérer correctement le paramètre USER afin de l'utiliser en tant qu'UTILISATEUR lorsqu'une tâche est soumise
- CL - Support des commandes DLTOVR amélioré pour gérer le cas de *ALL
- Zone de données - Support amélioré pour Blu4 DataArea en ajoutant la journalisation pour la gestion des exceptions
- Zone de données - Support amélioré pour Blu4 DataArea afin de récupérer une nouvelle DataAreaDao instance pour chaque thread
- Zone de données - Verrouillage amélioré des zones de données, évitant les verrouillages au niveau des enregistrements et utilisant plutôt le mécanisme de verrouillage nouvellement mis en œuvre
- Zone de données - L'opération d'écriture de la zone de données se poursuit désormais lorsqu'aucun verrou n'est acquis et qu'un indicateur d'erreur est fourni
- Rapport - Amélioration de la prise en charge du chemin de sortie des rapports et de la convention de dénomination pour les rapports imprimés. A permis aux clients de personnaliser le chemin de sortie du rapport ainsi que le nom. Le client peut définir son propre chemin et sa propre convention de dénomination sans affecter aucun autre projet.
- JOB - Support amélioré pour la gestion des tâches (Quartz) afin de mettre à jour le statut des tâches en cas de fin de tâche anormale. Par exemple : « arrêt » ou « arrêt anormal » de Tomcat
- Écran - Gestion améliorée de la valeur numérique sur le champ avec modification du mot marqué d'un signe négatif
- Écran - Fenêtre contextuelle de rendu améliorée avec uniquement titleColorTop
- Écran - Support amélioré pour la récupération des informations d'aide afin de traiter les cas où aucun élément d'aide générale n'est trouvé

- Écran - Amélioration de l'affichage de l'écran « informations supplémentaires » lorsque vous appuyez sur F1 sur la ligne de message du sous-fichier
- Écran - Affichage amélioré des pieds de page de message pour SFLMSG
- Écran : interface améliorée permettant de supprimer un enregistrement dans son intégralité lorsqu'un nouvel enregistrement le chevauche
- Mise en file d'attente : récupération améliorée des messages RabbitMQ pour consommer moins de ressources
- Mise en file d'attente - Implémentation améliorée de la file de données RabbitMQ pour ne récupérer qu'un seul message à la fois.
- SQL - Amélioration de la gestion de SQLCODE par le SQLExecutor Builder pour les requêtes de table dynamiques CREATE et DROP
- SQL - Support amélioré d'OVRDBF sur les requêtes
- SQL - SQLExecutor Générateur amélioré afin que les remplacements OVRDBF soient appliqués aux instructions préparées
- RPG - Support amélioré pour les spécifications d'entrée et de sortie des fichiers disque décrits par le programme
- RPG - Support amélioré pour la lecture des fichiers principaux et secondaires avec l'indicateur MR (Matching Records). L'ordre de récupération d'un cycle DAO avec champs de correspondance a été amélioré.
- RPG - Support amélioré pour les fichiers principaux et secondaires. Amélioration de la mise à jour des fichiers principaux et des fichiers de sortie Les fichiers secondaires mettent à jour/écrivent du code.
- RPG - Ajout du support pour l'instruction RETURN au format libre
- RPG - Amélioration de la transformation et de la gestion à l'exécution des assignations décimales numériques,
- RPG - Génération améliorée de variables binaires
- RPG - Support amélioré pour EDITC
- RPG - Gestion améliorée de la zone de données locale
- Support amélioré des champs DDS partagés par plusieurs types de périphériques (DISK, WORKSATION, PRINTER)
- Gestion améliorée des dérogations afin que les dérogations activées n'aient plus d' PFs incidence LFs

- Blu4 amélioré pour ivWebController ne pas réinitialiser le nom d'utilisateur et l'identifiant utilisateur aux valeurs par défaut
- Amélioration de l'ajustement de l'index lors de la lecture des enregistrements lorsque le sens de lecture change
- Amélioration du placement du curseur lors des lectures d'enregistrements après les opérations de mise à jour/suppression
- Support amélioré de la lecture sur un DAO multi-entités lorsque le sens de lecture change
- Support amélioré pour les espaces utilisateur afin d'éviter que les instances ne soient réutilisées par tous les threads au lieu que chaque thread ait sa propre instance
- Prise en charge améliorée de l'accès simultané à plusieurs threads lors de la lecture des enregistrements
- Amélioration du stockage du nom d'utilisateur/identifiant d'utilisateur via la configuration YML SharedContext
- Version améliorée des enregistrements verrouillés avec des valeurs mises à jour
- Ajout de la prise en charge du comportement spécifique du compilateur OPM pour l'instruction NEXT SENTENCE

Capacités transversales

Nouvelles fonctionnalités

- La nouvelle propriété metadata.ini ajoutée `legacy.compilerto` spécifie l'ancien compilateur des artefacts à transformer. La prise en charge de certaines instructions COBOL, telles que NEXT SENTENCE, varie en fonction de la valeur que vous définissez.
  - « ZOS » pour un ancien système z/OS.
  - « ILE » ou « OPM » pour le système AS4 00. Par défaut = « ILE » quand `legacy.system` = « as400"

Améliorations

- Front-end - Les composants des champs d'écran ont été repensés pour élargir la gamme des types de champs pris en charge. Cette amélioration permet au moteur d'exécution de répondre à une plus grande variété d'exigences en matière de saisie et de données par les utilisateurs dans AS4 00.
- Méthode améliorée `isValid()` pour séparer le signe-octet activé ZonedType

- Support amélioré `StringConcatenationBuilder::withPointer` pour la concaténation impliquant le CRLF
- Amélioration de la prise en charge du codage spécifique à double octet afin de les rendre sûrs dans les threads
- Amélioration des performances des machines à états grâce à l'intégration d'un nouveau framework
- Algorithme amélioré pour l'optimisation des affectations afin d'éviter les réécritures inattendues

AWS Moteur de transformation Blu Age 4.4.0

ZoS

Améliorations

- LISTCAT - Analyseur amélioré pour éviter les doublons
- LISTCAT - Support amélioré de l'ESDS au système de fichiers dans JCL/Groovy
- CICS - Support amélioré de LENGTH OF pour les instructions CICS

AS400

Améliorations

- Amélioration de la génération d'enregistrements DDS
  - Amélioration de la prise en charge de l'enregistrement DDS pour générer des entités correspondant à la structure d'enregistrement DDS
  - A fourni un support pour les champs partagés et les fonctions de mappage qui correspondent mieux à l'héritage
  - Amélioration de la gestion des fichiers décrits en externe et des fichiers décrits par le programme
- RPG - Détection RPG améliorée pour les modules avec uniquement une forme libre
- RPG - Amélioration de la prise en charge de l'instruction COPY permettant d'ignorer le mot clé `*LIBL/` en tant que préfixe pour localiser un cahier d'application
- RPG - PF - Support amélioré pour la spécification d'entrée avec des enregistrements physiques à partir de pfile
- RPG - Ajout du support de la déclaration On-Exit
- RPG - Support amélioré des mots clés LikeRec
- RPG - Amélioration du mappage des champs DSPF renommés

- CL - Résolution améliorée des noms de champs
- COBOL - Support amélioré de la conversion de l'hexadécimal en caractère
- Support amélioré pour la génération de types décimaux
- Support amélioré du message FIXME pour le code existant non pris en charge (affichage de la ligne complète de l'ancien code)
- Performances améliorées sur AWS Transformation Engine (étape d'analyse AS4 00)
- Support amélioré du mot clé LikeRec pour l'aligner sur les spécifications du fichier
- Support amélioré de la fonction intégrée %Diff
- Ajout de la prise en charge du symbole monétaire à caractères spéciaux sur l'étiquette DSPF

Notes de mise à jour 4.3.0

Date de sortie : 16 septembre 2024

Cette version de AWS Blu Age Runtime and Modernization Tools vise à étendre les capacités et la couverture afin de moderniser les fonctionnalités du mainframe. Parmi les principales fonctionnalités et modifications apportées à cette version, citons :

- CICS : support supplémentaire pour échanger des données depuis les terminaux et exécuter des transactions avec les données entrantes en prenant en charge la commande SEND MAP avec Map Reference.
- JCL : nouvelle fonctionnalité qui permet de redémarrer l'exécution la plus récente d'un traitement par lots à partir d'une étape JCL/PROC précédemment échouée, ou de déclencher un redémarrage différé en contournant les étapes précédemment exécutées. Cela permet de mieux contrôler le traitement par lots grâce à des points de contrôle persistants au niveau des étapes.
- AS400 : Support de bibliothèques supplémentaire, performances améliorées et robustesse des commandes couramment utilisées telles que CPYF, OVRDBF, SBMJOB, OPNQRYF et bien d'autres encore.

Nous avons testé cette version du AWS Blu Age Runtime avec le stack suivant. D'autres versions de composants peuvent également être compatibles.

Composant	Version testée

Java	Java 17
Couche de présentation	Nœud JS 18.18
	Npm 9,8
	Angulaire 17
couche de service	Spring Boot 3.2.5
	Spring Core 6.1.5
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 4.3.0

ZoS

Nouvelles fonctionnalités

- CICS - Support ajouté pour la référence cartographique dans la commande SEND MAP
- CICS - Ajout de la prise en charge de la commande RECEIVE et de la prise en charge de l'exécution de transactions avec les données de l'écran `JicsTransactionRunner`
- Ajout de la prise en charge de l'en-tête IIH pour les messages JMS
- COBOL - Ajout de la prise en charge de plusieurs espaces incorporés dans le pseudo-texte pour l'instruction REPLACING
- COBOL - Ajout du support pour l'instruction JSON PARSE
- Blusam - Ajout du support pour KMS afin de proposer la fonctionnalité « Exporter un ensemble de données »

- BAC - Ajout de la configuration de `application-main.yaml` pour définir la taille de l'enregistrement afin de filtrer les masques chargés correspondant à cette taille d'enregistrement
- JCL - INFUTILB - Ajout du support pour le mot clé INTO dans le cadre de la déclaration de contrôle BMC
- GS21 - Ajout de la gestion SOSI pour le codage JEF
- GS21 - JCL - Ajouté KDJBR14 en tant qu'alias de IEFBR14
- GS21 - JCL - Ajout de KQCAMS en tant qu'alias d'IDCAMS
- MF - Ajout du support pour les fichiers compatibles COBOL MF en fonction de l'assistance sur le terrain
- MF - Ajout du support pour le mécanisme de tri pour les fichiers compatibles COBOL MF
- MF - Ajout du support pour les fichiers manquants non facultatifs ouverts compatibles COBOL MF

Améliorations

- JCL - DSNUTILB - Opération LOAD améliorée avec le type ZONED DECIMAL
- JCL - DSNUTILB - Ajout du support de la clé dupliquée
- JCL - DSNUTILB - Ajout du support pour le mécanisme de restauration sur la commande LOAD
- JCL - INFUTILB - UNLOAD amélioré avec les nouvelles propriétés FETCHSIZE et CHUNKSIZE
- JCL - IKJEFT1 A - Amélioration de la lecture des fichiers SYSTSIN en ajoutant le jeu de caractères actuel
- JCL - DFSORT - Ajout du support pour les options et DATE4 DATE5
- JCL - DFSORT - Ajout de la prise en charge du type de bloc variable en entrée et du type de bloc fixe en sortie
- JCL - DFSORT - Ajout du support pour ALTSEQ
- JCL - Métadonnées de point de contrôle améliorées avec identifiant Web de tâche
- JCL - Amélioration de la purge du point de contrôle du redémarrage par lots pour REDIS
- IMS - Fonction EXPRESS implémentée pour la commande PURGE
- IMS - Ajout du support pour les options PCBNAME et LIST pour la déclaration PCB
- COBOL - Ajout du support pour l'instruction GO TO sans cible
- CICS - Support amélioré de l'instruction INTO dans RecordAdaptable READQ TS
- CICS - Support amélioré pour la commande INQUIRE TRANSACTION
- CICS - Support amélioré pour SetBytes dans la commande READNEXT

- CICS - Support amélioré pour la commande START sans option CHANNEL
- CICS - Support ajouté pour le type de référence pour Inquire TSQueue
- CICS - Support amélioré pour la commande RECEIVE MAP lorsque map et mapset sont des références
- CICS - Support amélioré pour les options FROM et LENGTH pour la commande RECEIVE MAP
- CICS - Ajout du support de l'attribut RecordAdaptable
- CICS - Support amélioré de la commande RECEIVE pour gérer le débordement
- CICS - Ajout de la prise en charge de la règle de tranche dans les instructions CICS
- CICS - Support amélioré pour les structures de liaison DFHCOMMAREA et DFHEIBLK. Le moteur de transformation prend en charge des définitions plus implicites
- CICS - Ajout du support pour les options START, NEXT et END pour la commande INQUIRE CONNECTION
- CICS - Ajout du support pour les types « int » et « reference » pour l'option LENGTH de la commande RECEIVE
- CICS - Support amélioré pour l'analyse de la commande INQUIRE NETNAME
- CICS - Ajout du support pour le nom de groupe pour JicsQueueBuilder
- Blusam - Ajout du support pour les fichiers indexés commençant par une clé générique
- Blusam - Chargeurs Blusam améliorés
- BAC - Support amélioré pour la synchronisation des données dans un environnement multi-instance lorsque Redis est utilisé pour centraliser les valeurs mises en cache, y compris les données réelles et les verrous
- BAC - Interface utilisateur améliorée (style, logo, case à cocher)
- BAC et JAC - Ajout de la configuration de `application-main.yaml` pour récupérer le nom d'utilisateur et le mot de passe de l'utilisateur super administrateur par défaut dans le secret d'AWS Secrets Manager en spécifiant l'ARN
- BAC et JAC - Mise à niveau de la dépendance vers Bootstrap 5
- Amélioration des points de contrôle JCL et de la configuration du modèle JICS Redis TSQueues
- Support amélioré pour la taille du pointeur en fonction de AMode
- Ajout de la prise en charge de l'absence de comparaison sur NumericEditedType
- Propriétés MDC SLF4j appliquées avant la journalisation
- Support amélioré de la lecture de fichiers pour gérer plusieurs lignes vides

- MF - Support amélioré pour l'initialisation des variables de pointeur pour la directive du compilateur COBOL MF InitPtr
- Redis - Fonctionnalité améliorée GwFileLock sur l'aspect concurrence grâce à une implémentation basée sur Redisson

AS400

Nouvelles fonctionnalités

- CL - Ajout du support pour la commande CHGPF
- RPG - Ajout du support pour les fonctions %HOURS, %MINUTES et %SECONDS
- COBOL - Ajout du support du fichier SORT avec l'architecture Blu4iv DAO

Améliorations

- CL - Amélioré PgmClose pour être enregistré en tant que programme et accepter une variété d'objets pour le paramètre OPNID
- CL - RTVMBRD refactorisé pour gérer plusieurs bibliothèques et membres
- CL - Ajout du support pour le paramètre TOLIB sur la commande MOVOBJ
- CL - Support amélioré de la partition sur la commande CPYFRMSTMF
- CL - Ajout du support pour le paramètre SNDMSG TOUSR
- CL - Support amélioré de la commande OVRDBF
- CL - Performances améliorées pour la commande OVRDBF - Mettre à jour les valeurs par défaut pour srcfile et member
- CL - Copie de fichiers améliorée avec la commande CPYF
- CL - Commande CPYF repensée pour être plus robuste et mieux gérer QTEMP, CRTFILE, FROMRCD & TORCD, MBROPT et FMTOPT (MAP & DROP)
- CL - Support amélioré pour la commande CPYF dans les cas où FROMFILE et TOFILE ont des colonnes incompatibles
- CL - Amélioration de la gestion par CPYF NOCHK des colonnes portant des noms différents lorsque REPLACE est spécifié
- CL - Ajout d'une implémentation vide pour la commande CRTDUPOBJ sur les fichiers logiques
- CL - Problème d'indexation des sous-chaînes géré avec la commande CHGDTAARA

- CL - Support amélioré de la commande SBMJOB
- CL - Made OverrideManager et OpnqryfHelper mappages insensibles aux majuscules et minuscules
- Écran - Amélioration de la mise au point initiale du premier champ modifiable lorsqu'aucun curseur n'est spécifié
- Écran - Position de mise au point améliorée après la fermeture et lors de l'utilisation du menu d'aide
- Écran - Amélioration de la mise au point du curseur après avoir appuyé sur la page haut/bas dans le composant du tableau
- Écran : prise en charge améliorée des messages d'erreur et du focus sur plusieurs champs
- Écran - Amélioration du calcul du numéro de ligne pour les champs de sous-fichiers
- Écran - Support amélioré des sous-fichiers initialisés à l'aide de SFLINZ
- Écran - Support amélioré pour la saisie numérique uniquement
- Écran - Gestion améliorée du mot clé WINDOW dans le DSPF avec 3 paramètres
- Écran - Position améliorée du pied de page pour les tableaux contenant des enregistrements contenant plus d'une ligne
- Écran - Navigation de page améliorée pour que le message de rotation reste collé sur la page haut/bas
- Fonctionnalité EDITC améliorée pour le code d'édition 3
- Mécanisme de verrouillage de la zone de données Blu4IV amélioré pour ne rien faire lorsqu'il n'y a pas de verrou à déverrouiller au lieu de lancer une exception
- Ajout du support pour renvoyer le nombre de lignes affectées dans StraightQueryBuilder
- Mécanisme de journalisation QTEMP amélioré
- Amélioré DAOManager reads/writes/deletes pour les cas d'utilisation sur un fichier remplacé par une bibliothèque de fichiers et de bibliothèques différente

Capacités transversales

Nouvelles fonctionnalités

- Ajout d'un moyen centralisé de gérer les propriétés du système liées au SSL/TLS par configuration, permettant l'utilisation de AWS Secrets Manager
- Configuration améliorée des ressources IBMMQ avec AWS Secrets Manager

- JCL - Ajout de la configuration de l'emplacement temporaire pour les fichiers groovy résolus par l'exécution via la propriété YML tempFilesDirectory et ajout de la possibilité de spécifier s'il faut purger le contenu du dossier des fichiers temporaires au démarrage de l'application via la propriété YML cleanTempFiles DirectoryAtStartup
- Ajoutez des secrets AWS pour toutes les informations d'identification Redis

Améliorations

- Conversion améliorée du type alphanumérique au type numérique édité
- Vérification améliorée de DataUtils ::IsNumeric pour PackedType
- Horodatage amélioré des fichiers journaux
- Connexion gérée séparément ZonedType. decodeAsString
- COBOL - Support amélioré de l'instruction INITIALIZE
- Support amélioré de DataUtils. compareAlphInt pour gérer les espaces de début et de fin pour AS4 00 et ZOS
- SQL - Validation implicite améliorée de l'exécution du curseur en lecture seule
- SQL - Mécanisme de mise en cache des métadonnées amélioré
- Supprimer la connexion à la base de données Jics/Blusam de l'application Gapwalk `application-main.yml`

Outils de modernisation, version 4.3.0

ZoS

Nouvelles fonctionnalités

- GS21 - Ajout du support pour COBOL GS21 CONSTANT SECTION
- GS21 - Ajout d'un encodage JEF aux jeux de caractères disponibles

Améliorations

- CICS - Ajout du support pour l'analyse de la commande DOCUMENT CREATE
- CICS - Ajout du support pour analyser la commande CICS WEB EXTRACT
- CICS - Ajout du support pour l'analyse de la commande WEB WRITE

- CICS - Ajout du support de transformation pour DB2 CONN SIGNIN et PLAN
- CICS - Support amélioré pour l'analyse de la commande SEND MAP en ignorant l'option TERMINAL
- CICS - Support amélioré pour l'analyse de la commande RETURN en ignorant l'option ENDACTIVITY
- MFS - Support amélioré pour générer des fichiers MFS avec une extension spécifique
- COBOL - Support amélioré pour l'instruction REPLACE
- COBOL - Chemin dynamique géré et directive du compilateur MF
- COBOL - Améliore la prise en charge de la valeur OMISTED dans la déclaration CALL
- COBOL - Accès aux champs multidimensionnels amélioré pour prendre en charge la valeur signée
- COBOL - Ajout du support pour la clause OF pour l'instruction FILE STATUS
- COBOL - Amélioration de l'analyse syntaxique des instructions RESULT-SET-LOCATOR
- JCL - IDCAMS - Ajout du support pour l'abréviation RECORDS

AS400

Nouvelles fonctionnalités

- CL - Ajout du support pour les variables définies et basées sur un pointeur dans la transformation CL
- CL - Ajout du support pour les caractères spéciaux dans le DCLF
- Ajout du support pour l'API de récupération d'appels (QWVRCSTK)

Améliorations

- RPG - Transformation améliorée des paramètres de procédure à l'aide `likeds` de mots clés
- RPG - Vérifiez le support du mot clé EXTNAME
- RPG - Valeur littérale de support améliorée *ALL
- RPG - Support amélioré pour les spécifications de sortie et les fichiers décrits par le programme
- DDS - Résolution améliorée des champs DDS dans un LF qui fait référence à un PF faisant référence à un dictionnaire PF
- Écran - Indicateurs effacés lorsque l'instruction CLEAR est utilisée pour effacer un enregistrement du DSPF

- CL - Transformation/génération améliorée des paramètres CL avec des listes d'éléments

Capacités transversales

Améliorations

- SQL - Amélioration de la génération de requêtes SQL contenant N avec un caractère tilde
- COBOL - Support amélioré de l'instruction LENGTH OF pour les champs de groupe
- COBOL - Support amélioré des champs REDÉFINIS à l'aide de cahiers

Notes de mise à jour 4.2.0

Date de sortie : 10 juillet 2024

Cette version de AWS Blu Age Runtime and Modernization Tools est axée sur les performances et la sécurité. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Nous avons amélioré les performances de transformation, en particulier pour les grands projets comportant plus de 30 millions de lignes de code. Nous avons mis en œuvre un ensemble d'améliorations et les résultats que nous avons obtenus ont montré une réduction du temps de plus de 150 % et des cycles terminés en quelques minutes au lieu de plusieurs heures. La principale amélioration que nous avons mise en œuvre est la configuration d'un mécanisme de temporisation pour limiter le temps maximum alloué à l'analyse afin d'ignorer les fichiers présentant des problèmes détectés. Nous marquons les fichiers ignorés afin que vous puissiez les examiner ultérieurement si nécessaire.
- Nous avons ajouté la prise en charge d'un système de gestion des serrures distribué pour les projets AS4 00. Dans un environnement de haute disponibilité (multi-nœuds) où plusieurs instances de l'application ciblent la même base de données, le maintien de la cohérence des données tout au long du cycle de vie de ces instances constitue un défi de taille. Pour relever efficacement ce défi, nous avons ajouté Redis en tant que serveur de mise en cache partagé et externe afin de coordonner toutes les instances lors de l'exécution en mode batch.
- Nous avons ajouté une nouvelle fonctionnalité de pagination dynamique pour le composant de tableau. L'objectif de cette fonctionnalité est d'améliorer le temps de réponse et de réduire l'utilisation de la mémoire pour les tables comportant un grand nombre de lignes. Cette fonctionnalité permet au composant de table de ne charger qu'une partie des données et de récupérer davantage d'enregistrements à la demande lorsque vous naviguez dans les pages.

Pour améliorer encore l'expérience, la plateforme prend également en charge la préextraction des données. Cette nouvelle fonctionnalité de pagination dynamique fournit une expérience utilisateur plus efficace et réactive pour les applications comportant de grands ensembles de données.

- Pour relever un défi majeur qui revient fréquemment, nous avons ajouté la prise en charge des programmes COBOL imbriqués. Auparavant, la solution de contournement pour moderniser les programmes COBOL imbriqués impliquait de séparer manuellement les programmes en différents fichiers, de les lier via la section de liaison et de les obliger à s'appeler avec les arguments nécessaires. Ce processus était non seulement chronophage, mais également sujet aux erreurs. Vous pouvez désormais moderniser les programmes COBOL imbriqués sans avoir à les séparer manuellement.

Nous avons testé cette version du AWS Blu Age Runtime avec le stack suivant. D'autres versions de composants peuvent également être compatibles.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 18.18
	Npm 9,8
	Angulaire 17
couche de service	Spring Boot 3.2.4
	Spring Core 6.1.5
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 4.2.0

ZoS

Nouvelles fonctionnalités

- DB2 - Ajout de la prise en charge de l'invocation de procédures stockées sans qualificatif de schéma dans la requête SQL
- COBOL - Ajout du support pour la fonction HEX-OF
- COBOL - Ajout du support pour les programmes imbriqués
- COBOL - Ajout du support pour TEST-DATE-YYYYMMDD FUNCTION et TEST-DAY-YYYYDDD
- CICS - Ajout du support pour l'option UCTRANST dans la commande SET TERMINAL
- CICS - Ajout du support pour la commande INQUIRE CONN DB2
- BluSam - Ajout de la prise en charge de la suppression de clés sur VSAM à accès dynamique
- IMS - Ajout du support pour la commande TERM
- BAC - Ajout de contrôles d'autorisation sur tous les points de terminaison BAC REST
- BAC - Configuration ajoutée `application-main.yaml` pour définir une taille d'enregistrement afin de filtrer les masques chargés correspondant à cette taille d'enregistrement
- BAC et JAC : ajout d'une configuration permettant de `application-main.yaml` récupérer le nom d'utilisateur et le mot de passe de l'utilisateur super administrateur par défaut dans le formulaire secret en spécifiant l'ARN `command`

Améliorations

- JCL - SORT - Support amélioré de la clause OMIT pour gérer les conditions avec Shiftin et les caractères ShiftOut
- JCL - SORT - Support amélioré pour le champ BDW
- JCL - SORT - Support amélioré pour plusieurs concaténations GDG avec le champ BDW
- JCL - DFSORT - Ajout du support pour les clauses INREC PARSE STARTAFT/STARTAT
- JCL - IEBGENER - Gestion améliorée de la taille des enregistrements pour les fichiers de sortie
- JCL - INFUTILB - INDICATEUR NULL désactivé basé sur YML- FIX GRAPHIC CASE
- JCL - Support amélioré pour la gestion FormatterParser des constantes dans le champ OUTREC
- JCL - Données de chargement améliorées pour le type graphique dans l'utilitaire de programme DSNUTILB

- JCL - SORT - Support amélioré pour le format décimal zoné
- JCL - SORT - Support amélioré de la clause OMIT pour gérer les conditions avec Shiftin et les caractères ShiftOut
- MQ - Amélioration de la gestion de la connexion MQ pour l'adapter à plusieurs flux de travail professionnels
- CICS - Support amélioré de la référence du pointeur pour les instructions EXEC CICS READ SET (ptr-ref)
- COBOL - Support amélioré pour l'enregistrement de la section ADDRESS OF linkage
- COBOL - Ajout du support pour les fonctions EXP et 0 EXP1
- COBOL - Support amélioré pour l'instruction REPLACE à l'aide d'un cahier
- COBOL - Accès aux champs multidimensionnels amélioré pour prendre en charge les valeurs signées
- MF COBOL - Ajout du support pour les fichiers séquentiels à format variable
- IMS - Lecture améliorée de la configuration des fichiers IMS YML pour permettre l'utilisation de variables d'environnement
- IMS - Gestion de méthodes supplémentaires pour spécifier le numéro de segment
- IMS - Robustesse accrue lorsqu'un programme IMS est appelé à partir d'une transaction démarrée par programme
- IMS - Amélioration des critères de recherche (version SSA) pour prendre en compte la longueur actuelle de la clause WHERE si la longueur de segment implicite n'est pas fournie
- IMS - Lecture améliorée de la configuration des fichiers IMS YML pour permettre l'utilisation de variables d'environnement
- Support amélioré pour la clause VALUE dans NumericEditedType
- Support amélioré pour la concaténation de chaînes afin de gérer le cas où la première chaîne à concaténer est vide, vide ou vide

AS400

Nouvelles fonctionnalités

- Ajout de la prise en charge de la pagination dans le composant Table ; les projets peuvent utiliser cette fonctionnalité pour réduire le temps de réponse et la taille lorsqu'un composant Table comportant un grand nombre de lignes est chargé

- Ajout du support des bibliothèques pour les requêtes SQL dans l'application AS4 00 ; les bibliothèques étant converties en partitions dans les applications modernes, nous avons adapté le runtime pour réécrire les requêtes en conséquence
- RPG - Ajout du support pour la bibliothèque QTEMP pour les requêtes SQL
- RPG - Ajout d'un encodage dans la fonction CONVERT pour gérer les valeurs d'entrée vides
- RPG - Ajout du support pour les fonctions %HOURS, %MINUTES et %SECONDS
- CL - Ajout de la commande CHGPFM
- CL - Ajout du support pour le mot clé *FROMLIB dans la commande CRTDUPOBJ
- CL - Ajout du support pour la création de tables et de partitions pour les noms de table supérieurs à 9 caractères
- CL - Ajout du support pour la suppression de fichiers plats dans les sous-dossiers pour la commande DLTF

Améliorations

- Écran - ErrorMessage Amélioré pour lier un champ spécifique et ajouter à ArrayMessageLine
- Écran - Curseur errormsg amélioré
- Écran - Amélioré ArrayMessageLine pour ne pas être inclus dans l'ordre des onglets
- Écran - Affichage amélioré des tableaux de messages d'erreur pour l'écran AS4 00
- SQL - Amélioration de la prise en charge du curseur pour valider la transaction à la fermeture afin d'éviter les blocages lors de la création de partitions
- CL - Ajout du support pour la PgmCall commande et amélioration du modèle non pris en charge par QCMDEXC
- CL - Support amélioré de la commande CHKOBJ pour gérer OBJTYPE PGM
- CL - Support multi-bibliothèques amélioré pour CPYF et les autres commandes CL qui traitent des bibliothèques et des partitions
- CL - Ajout du support pour transmettre une variable de nom de programme dans la commande CALL PGM
- CL - A traité le cas du type d'objet par défaut
- CL - Ajout du support multi-bibliothèques pour la commande CRTDUPOBJ
- CL - Gestion améliorée des connexions à la base de données sur plusieurs commandes
- CL - Support amélioré de RMVLNK pour gérer le cas où un fichier ou un répertoire n'est pas trouvé et le message du moniteur CPF0000

- CL - CLRPFM amélioré pour prendre en compte la bibliothèque lors de la suppression d'enregistrements
- CL - CPYF - Commande améliorée pour prendre en charge la bibliothèque QTEMP, le paramètre FmtOpt (*NoChk) et le caractère de contrôle
- CL - Correction de la gestion des guillemets et des paramètres manquants dans les commandes RMVLNK et CPY
- RPG : étendue des variables améliorée ; elle DataArea est désormais intégrée à la zone de travail au lieu de la portée de liaison
- RPG - Amélioration des requêtes de lecture DAO pour qu'elles s'exécutent sans transaction pour éviter les blocages
- Recherche de messagerie MQ améliorée en ajoutant un découpage à MSGQ lors de la recherche dans la base de données
- Suppression des déclarations de transactions inutiles sur le support de connexion à la base de données
- Amélioration de la mise à jour du statut des tâches de Quartz en cas d'exception
- Ajout d'un support pour gérer le cas où un tableau d'indicateurs n'est pas initialisé

Capacités transversales

Nouvelles fonctionnalités

- Redis - Ajout d'une configuration Redis globale pour tous les caches Redis
- Ajout d'une fonctionnalité de suivi de session pour permettre de stocker des informations de suivi de session (ID de session, nom d'utilisateur associé, horodatage de création et ID de nœud) en conservant les données dans Redis
- Ajout d'une configuration d'emplacement temporaire pour les fichiers groovy résolus lors de l'exécution via la propriété YML `tempFilesDirectory` ; ajout de la possibilité de spécifier s'il faut purger le contenu du dossier des fichiers temporaires au démarrage de l'application via la propriété YML `cleanTempFilesDirectoryAtStartup`

Améliorations

- Prise en charge améliorée de l'implémentation du pool de connexions, des propriétés de configuration pour les sources de données utilitaires

- Support amélioré pour le mode imprimante et le contrôle des chariots ANSI grâce à l'utilisation des clauses ADVANCED et WRITE BEFORE
- Version angulaire mise à jour sur l'application frontale pour les projets modernisés
- Construction améliorée de la syntaxe de l'URL du gestionnaire de secrets pour DB2
- Amélioré le DataUtils. compareAlphInt méthode pour ajouter un support pour les espaces de fin
- Support SQL amélioré pour les sorties de type blob
- Robustesse accrue pour les déclencheurs de tâches via le point de terminaison post/script

Outils de modernisation, version 4.2.0

ZoS

Nouvelles fonctionnalités

- CICS - Ajout du support pour l'analyse des commandes WEB CICS
- CICS - Ajout du support pour la transformation de la commande MONITOR
- CICS - Ajout du support pour l'analyse de la commande CICS SEND MRO
- COBOL - Ajout du support pour l'analyse de l'instruction NO REWIND
- COBOL - Ajout du support pour le type de numéro de l'option UCTRANST dans la commande CICS SET TERMINAL
- COBOL - Ajout de la prise en charge de la clause MULTIPLE FILE dans I-O-SECTION
- CSD - Ajout du support pour la transformation de plusieurs fichiers CSD
- CSD - Ajout du support pour la génération de jicsFileAix .json à partir de plusieurs fichiers CSD
- IDCAMS - Ajout du support pour la création d'un ensemble de données d'enregistrement relatif (RRDS)

Améliorations

- Performances améliorées lors du calcul des masques SQL
- COBOL - Analyse améliorée de la clause RESERVE inutile dans FILE-CONTROL
- COBOL - Analyse améliorée de SECTION et CLASS
- COBOL - Gestion améliorée du DFHRESP
- COBOL - Support amélioré pour EXIT PARAGRAPHE grâce à Perform

- IMS - Amélioration de la prise en charge des noms de segments spécifiés à l'aide de doubles parenthèses
- IMS - Amélioration de la génération de codes d'état lorsque SCHD et TERM sont invoqués
- COBOL - Amélioration de la génération de champs DEPENDING ON
- COBOL - Transformation améliorée de la fonction intégrée DB2 TO_TIMESTAMP

AS400

Nouvelles fonctionnalités

- Ajout du support pour la conversion de champs alphanumériques en CHAR dans les scripts SQL
- COBOL400 - Ajout du support pour les fichiers de base de données décrits par le programme

Améliorations

- DDS - Support amélioré pour le nom ALIAS
- Support amélioré pour le type float sans valeur initiale
- COBOL 400 - Calcul de taille amélioré pour le type zoné signé

Capacités transversales

Améliorations

- Amélioration des rapports d'identification des erreurs liés à l'analyse DDS et SQL
- Génération de code améliorée sur les branches de condition
- Performances améliorées lors de la génération de rapports météorologiques

Notes de mise à jour 4.1.0

Date de sortie : 31 mai 2024

Cette version de AWS Blu Age Runtime and Modernization Tools est axée sur les performances et la sécurité. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Transformation et performance : pour permettre aux projets dotés d'une base de code importante (+ 50 millions de lignes de code) de réussir leur transformation, nous avons optimisé les performances et l'empreinte mémoire de l'ensemble du mécanisme de transformation.

- BAC/JAC : La sécurité AWS est la priorité absolue. Les applications modernisées avec AWS Blu Age doivent être conformes aux normes de sécurité. Nous avons apporté quelques mises à niveau majeures à la console d' BluSam administration (BAC) et à la console d'administration JICS (JAC) afin de les rendre plus sûres :
  - Mise à jour de l'application vers Angular v17.
  - Outre le support natif d'AWS Cognito, nous avons ajouté un support générique OAuth qui permettra aux clients d'utiliser le fournisseur d'identité de leur choix avec une plus grande flexibilité.
  - Configuré et étendu les fonctionnalités de sécurité à l'aide des en-têtes appropriés.
- AS400 - Support multi-nœuds pour le mécanisme de verrouillage de la base de données. Possibilité de brancher un serveur de mise en cache partagé et externe (Redis) pour exécuter une application par lots sur plusieurs instances, comme la modernisation d'un AWS mainframe géré.

Cette version du runtime Blu Age a été testée avec la pile suivante. D'autres versions peuvent également être compatibles.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 18.18
	Npm 9,8
	Angulaire 16.1
Couche de service	Spring Boot 3.2.5
	Spring Core 6.1.5
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21c
Serveur d'application	Apache Tomcat 10.1.17

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 4.1.0

ZoS

Nouvelles fonctionnalités

- Configuration ajoutée pour la gestion dynamique des OAuth2 fournisseurs. Introduction de SECRET_ OAUTH2 _PROVIDER_NAME_KEY pour spécifier le fournisseur. Méthode de récupération des secrets mise à jour pour gérer plusieurs fournisseurs. Les secrets garantis sont récupérés en toute sécurité à partir de AWS Secrets Manager.
- Ajout de la prise en charge des AWS Secrets Manager propriétés DB2 SSL pour vous permettre de définir un certificat SSL (sslTrustStoreemplacement) et un mot de passe (mot de sslTrustStore passe) pour déverrouiller le fichier keystore.
- Ajout de la prise en charge des sources de données commerciales externes.
- JCL - Ajout de la prise en charge du mécanisme de point de contrôle pour le redémarrage par lots.
- JCL - Ajout du support pour les paramètres DCB, la taille d'enregistrement et le RDW.
- JCL - Ajout d'une configuration dynamique des noms de dossiers pour les fichiers temporaires générés.
- REDIS - Ajout de la configuration du pool dans la configuration Redis pour JICS.
- REDIS - Ajout d'un index de base de données dans la configuration Redis pour Catalog et JICS.
- BatchScript - Ajout de la propagation du nom de l'étape pour les exécutions de programmes.
- CICS - Ajout du support pour la commande ADDRESS SET.
- CICS - Ajout du support pour PURGE MESSAGE et JUSTIFY.

Améliorations

- JCL - INFUTILB - Support amélioré pour la désactivation de l'indicateur nul en fonction de la propriété YML.
- JCL - INFUTILB - Support amélioré pour le type de données CHAR/BPCHAR.
- JCL - ICEGENER - Ajout du support pour la copie de flux d'entrée multilignes dans des fichiers.
- JCL - IEBGENER - Support amélioré pour la gestion de la conversion de fichiers à blocs variables en fichiers à blocs fixes.

- JCL - DFSORT - Amélioration de la prise en charge des paramètres à plusieurs chiffres à la date d'opération.
- JCL - DFSORT - Ajout du support pour la clause INCLUDE=ALL.
- JCL - Amélioration de la prise en charge de l'utilitaire SORT pour gérer le champ BDW en sortie.
- JCL - Support amélioré pour la concaténation DD.
- JCL - Support amélioré pour Input Stream.
- JCL - DSNUTILB - Support amélioré pour l'instruction NULLIF ().
- JCL - INFUTILB - Ajout du support pour le déchargement des données avec l'option NOPAD.
- JCL - INFUTILB - Support amélioré pour la date actuelle dans INFUTILB.
- JCL - Ajout de vérifications de l'existence et de la taille des fichiers avant d'utiliser un fichier.
- JCL - GDG - Amélioration de la gestion des sous-répertoires pour GDG.
- MQ - Ouverture de connexion améliorée dans l'implémentation JMS.
- MQ - Amélioration du réglage de la longueur des données du message GET pour la source de données XA.
- MQ - Cahier standard CMQV décomposé pour éviter les erreurs de compilation et les utilisations de refactorisation.
- BluSam - Amélioration de la prise en charge des demandes de suppression d'ensembles de données inexistants.
- Support amélioré pour l'instruction ALLOCATE.
- Robustesse améliorée du nommage TS-QUEUE.
- BatchScript - Préservation améliorée du code de retour de l'étape précédente lors de la réexécution de la tâche.
- Ensemble de données - Amélioration de la vérification de l'existence des fichiers lorsqu'un fichier existe et est temporaire.
- Ensemble de données - Amélioration de la simultanéité lors de la localisation des fichiers GDG à supprimer.
- Ensemble de données - Ajout du support pour obtenir la taille d'enregistrement du jeu de données GDG.
- CICS - Amélioration de la prise en charge de l'option SUSPENDU dans la commande INQUIRE TASK LIST.
- CICS - Support amélioré pour LOAD SET à l'aide de l'instruction ADDRESS OF.

- CICS - Amélioration des arguments CICS non gérés REMOTESYSTEM lorsque CICS INQUIRE.
- CICS - Support amélioré de la commande GETMAIN pour gérer l'option SET avec un pointeur défini avec le mot clé OF.
- JICS - Robustesse améliorée de la méthode jics XAPrepare () en ajoutant la vérification de l'état des transactions.
- JICS XA - Ajout d'une vérification de l'état des transactions et amélioration de la terminaison du thread de transaction.
- BAC - Authentification améliorée basée sur les rôles côté client et refactorisation/centralisation de tous les appels d'API.
- BAC - Implémentation d'une fonctionnalité pour bloquer l'accès public au BAC et au JAC en fonction de la configuration
- BAC - Mise à niveau des dépendances : Angular 17.
- BAC - Intégration de sécurité améliorée avec OAuth2.
- BAC - DDL amélioré généré par Hibernate.
- BAC - Mécanisme d'ensemble de données d'exportation amélioré.
- JAC - Mise à jour vers Angular 17 et signalant tous les travaux spécifiques effectués par BAC (ROLE, sadmin conf, XSRF, logout).
- COBOL - Ajout du support pour les fonctions CHAR et ORD-MIN.
- Amélioré FileFactory pour conserver la taille des enregistrements du catalogue dans la disposition du MOD.
- Enregistrement activé à l'aide de MDC pour les transactions JICS.
- SQLCA > SQLSTATE amélioré produit pour les procédures stockées générant des ensembles de résultats ad hoc.
- Support amélioré pour la planification des tâches liées à la dernière mise à niveau du printemps.

AS400

Nouvelles fonctionnalités

- Ajout du support multi-nœuds pour le verrouillage des enregistrements de base de données à l'aide de Redis.
- Ajout du support pour BINARY CHARACTER pour le type DDS.
- CL - Ajout du support pour la génération de fichiers de rapports personnalisés.

- RPG - Ajout du support pour le mot clé RENAME sur les fichiers primaires/secondaires.

Améliorations

- Support de base de données amélioré pour la gestion de la colonne CTID avec une clause JOIN.
- Position du curseur améliorée pour plusieurs DSPATR (PC).
- Amélioration de la journalisation en cas d'exception de lecture.
- Enregistrement des tâches Quartz amélioré pour inclure les propriétés des tâches dans le MDC.
- Support amélioré pour l'écran d'aide AS4 00.
- CL - Amélioration de la prise en charge de la commande RMVJOBSCDE pour accepter les numéros d'entrée suivis d'espaces.
- CL - Amélioration de la prise en charge de la commande RMVJOBSCDE pour supprimer un calendrier de travail utilisant un nom de tâche générique.
- CL - Amélioration de la prise en charge de la commande SAVOBJ pour trier les enregistrements par clé de table.
- CL - Amélioration de la prise en charge de la commande CPYF pour établir une nouvelle connexion pour les requêtes de base de données.
- CL - Amélioration de l'insertion des messages de demande dans les messages de file d'attente avec SNDPGMMSG.
- CL - Configuration de la file d'attente des tâches améliorée pour spécifier la file d'attente des tâches par défaut.
- CL - Amélioration de la commande CRTPF pour prendre en charge la bibliothèque QTEMP et le paramètre RCDLEN.
- CL - Support amélioré pour la commande CHKOBJ - Vérifiez la partition avec la bibliothèque.
- CL - RTVMGS amélioré pour envoyer CPF24 07 et CPF2419 lorsque le fichier/identifiant est introuvable.
- CL - Amélioration de l'interprétation CPYTOIMPF et CPYFRMIMPF des anciens paramètres de formatage.
- CL - Ajout du support pour le paramètre OVRPRTF USRDTA.
- CL - Amélioration de la commande CL CPYTOIPF pour établir une nouvelle connexion afin d'éviter de fermer les ensembles de résultats existants.
- CL - Amélioration de CHGDTAARA afin qu'il ne modifie plus la longueur de la zone de données lors de la mise à jour du contenu.

- CL - Gestion améliorée des connexions à la CICommand base de données.
- Interaction optimisée entre le front-end et le back-end.
- COBOL - Transformation mise à jour pour gérer le FILLER dans les cahiers.
- Affichage amélioré des informations supplémentaires sur les messages pour les messages personnalisés envoyés au front-end.
- Mise à jour de la valeur par défaut du sélecteur dans app.component.ts.
- Découpage du texte amélioré à l' split-dynamic-fieldécran.
- Amélioration de l'affichage du message d'erreur avec plusieurs écritures suivies d'une lecture.

Capacités transversales

Nouvelles fonctionnalités

Ajout du support pour la configuration dynamique du secret du OAuth2 fournisseur.

Améliorations

- Impression - Amélioration de la prise en charge des paramètres QCMDEXC pour la gestion des guillemets et amélioration de la formation des noms de rapport
- Prise en charge améliorée de la syntaxe délimitée activée RecordAdaptable.
- Enregistrement des InspectBuilder erreurs amélioré pour ajouter du contexte à la chaîne source.
- DataSimplifier - robustesse accrue pour l' ByteArray affectation.
- Journalisation MDC améliorée avec de nouveaux attributs d'exécution.

Outils de modernisation, version 4.1.0

ZoS

Nouvelles fonctionnalités

- Ajout du support pour plusieurs transformations de fichiers CSD
- COBOL - Ajout du support pour l'instruction CICS ALLOCATE.
- COBOL - Ajout du support pour ON SIZE ERROR dans l'instruction ADD CORRESPONDING.
- COBOL - Ajout du support pour EXIT PARAGRAPHE.

Améliorations

- COBOL - Support amélioré pour le copybook -INC.
- COBOL - Support amélioré pour l'initialisation de FILLER.
- COBOL - Support amélioré pour la comparaison de valeurs figuratives.
- COBOL - Support amélioré pour WHEN ANY dans les clauses WHEN consécutives dépourvues de blocs de code intermédiaires.
- COBOL - Support amélioré pour les constantes figuratives.
- COBOL - Support amélioré pour le calcul de la taille des caractères compressés.
- COBOL - Amélioration de l'argument CICS non géré KEEP pour SPOOLCLOSE.
- COBOL - Génération améliorée pour la fonction TEST-NUMVAL.
- COBOL - Amélioration des arguments de génération Java sur le support du framework INSPECT.
- CICS - Support amélioré pour la définition de DFHCOMMAREA.

AS400

Nouvelles fonctionnalités

- RPG - Ajout d'un mécanisme de détection des erreurs pour générer le DDS (incomplet) afin de ne pas bloquer la génération du programme.
- Ajout de la prise en charge du mot-clé de spécification de description de fichier INCLUDE.

Améliorations

- RPG - Analyse entièrement gratuite améliorée.
- RPG - Robustesse accrue grâce à la détection des erreurs.
- RPG - Initialisation améliorée du champ/DS avec le mot clé d'exportation.
- RPG - Fonctionnement DAO amélioré pour gérer les indicateurs.
- RPG - Gère la valeur par défaut de PERRCD avec CTDATA.
- RPG - Mise à niveau de l'analyseur Free-RPG pour enregistrer une erreur unique par règle d'analyse.
- PRTF - Gestion de la collision de noms entre PRTF et JRXML.
- COBOL - Support amélioré du mot clé LIKE.

Capacités transversales

Améliorations

- Robustesse accrue pour l'API ErrorID
- Optimisation des performances pour la transformation de projets de grande envergure. Par exemple : délai d'expiration pour ignorer les fichiers bloqués, réutilisation de la classification de Blu Insights et meilleure allocation de mémoire.
- Optimisation de l'empreinte mémoire lors de la transformation COBOL/PL1 .
- CVE fixe sur des sites tiers (jQuery et bootstrap).
- Options TimeOutParser gérées dans TC.
- Amélioration de la réécriture de plusieurs espaces sur les requêtes SQL.
- Curseur en lecture seule amélioré avec attribut de sensibilité.

Notes de mise à jour 4.0.0

Date de sortie : 8 avril 2024

Pour obtenir des instructions sur la migration de AWS Blu Age Runtime 3.10.0 vers la version 4.0.0, consultez. the section called “Migration de la version 3.10.0 vers la version 4.0.0”

Cette version de AWS Blu Age Runtime and Modernization Tools est axée sur la mise à niveau des dépendances critiques et des technologies prises en charge tout en améliorant les performances de multiples fonctionnalités. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- • Passez de Spring Boot 2.7 à 3.2.4, Spring Core 5.3 à 6.1.5 et Tomcat 9.0 à 10.1.17 pour améliorer la sécurité, les performances et la maintenabilité en utilisant des versions activement corrigées et maintenues.
  - Chargement différé sur l'application frontale pour créer plus rapidement de grands projets avec plus de 2 000 écrans et réduire l'initialisation de l'affichage de 10 s à 300 ms.
  - Support de l'affichage DBCS sur les applications frontales pour améliorer la prise en charge des caractères codés sur deux octets afin de fournir une nouvelle police qui gère les caractères codés sur deux octets et sur un octet, empêche la saisie d'un octet dans un champ codé sur deux octets et gère les champs contenant des caractères mixtes codés sur deux octets et sur un octet.

- Fonction de surveillance des threads pour l'application AS4 00 Online permettant d'exécuter l'application AS4 00 avec parallélisation.
- Amélioration des performances relatives au contexte et à l' RunUnitinitialisation grâce à l'ajout d'un mécanisme configurable permettant de pré-initialiser le contexte du programme, réduisant ainsi l'impact du chargement de structures complexes inhérentes à la complexité existante.

Cette version du AWS Blu Age Runtime a été testée avec la pile suivante. D'autres versions peuvent également être compatibles.

Composant	Version testée
Java	Java 17
Couche de présentation	Nœud JS 18.18
	Npm 9,8
	Angulaire 16.1
Couche de service	Spring Boot 3.2.4
	Spring Core 6.1.5
	Spring State Machine 4.0.0
Couche de persistance	Moteur PostgreSQL 14
	Oracle 21
Serveur d'application	Apache Tomcat 10.1.17

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 4.0.0

ZoS

Nouvelles fonctionnalités

- Ajout du support pour l'instruction include '-INC CPYNAME'.
- CICS - Ajout du support pour l'instruction PUSH/POP HANDLE.
- COBOL - Ajout du support pour « ASSIGN TO DYNAMIC ».
- Ajout du support pour DB2 UNLOAD à l'aide d'INFUTILB.
- Ajout de la prise en charge du mot clé SEQNUM dans une instruction OVERLAY of INREC.

Améliorations

- SORT - Ajout de la prise en charge des caractères spéciaux (parenthèses et astérisques) dans les chaînes de tri littérales C'... '.
- SORT - Support amélioré pour l'argument OUTFIL NOMATCH- (..).
- SORT - Ajout du support pour la définition des données SYMNAMES.
- SORT - Gestion améliorée des arguments TO= et LENGTH=.
- SORT - Gestion améliorée de la disposition des MOD.
- SORT - Ajout du support pour l'argument HIT=NEXT.
- ICEGENER amélioré pour ajouter la prise en charge de l'encodage de fichiers de sortie spécifiques.
- INFUTILB - Support amélioré pour la clause WITH UR.
- INFUTILB - Support amélioré pour le déchargement lorsque writeNullIndicator la valeur est fausse.
- DSNUTILB - Robustesse améliorée pour l'étape de chargement lorsque le mot clé NULLIF se trouve après un mot clé SQL facultatif.
- DSNUTILB - Support amélioré pour le nom de colonne d'isolation.
- DSNUTILB - Ajout du support pour charger un fichier vide dans une table.
- DNSUTILB - Ajout de la prise en charge de la disposition MOD pour le fichier SYSDISC DNSUTILB.
- IDCAMS - Support amélioré pour les commentaires.
- JCL - Ajout du support pour les colonnes avec guillemets doubles. LoadTask
- JCL - Gestion améliorée des requêtes SQL UNLOAD concernant la suppression des espaces blancs.
- JCL - Réponse améliorée du script Groovy lorsqu'une exception se produit pendant le traitement afin de garantir un format JSON.
- JCL - Disposition améliorée des fichiers de vérification dans le cas de DISP=NEW et DISP=OLD.

- JCL - Support amélioré pour gérer plusieurs références de génération GDG avec un caractère spécial dans le nom de base GDG.
- JCL - Support amélioré pour charger un fichier factice.
- JCL - Support amélioré pour le paramètre tempFilesDirectory YML.
- JCL - Amélioration du retour JSON lorsqu'il est nécessaire d'échapper aux guillemets doubles dans un élément de chaîne.
- JCL - Amélioré FileUtils pour prendre en charge le nom de base GDG.
- JCL - Programme DSNTEP amélioré pour DB2 l'exécution de plusieurs requêtes.
- Ajout du support pour les haricots de printemps.
- Amélioré SQLConverter pour éviter de rectifier les mauvaises dates.
- JicsTimeBuilder Gestion améliorée de YYYYDDD.
- Possibilité d'accéder aux pots personnalisés depuis Groovy.
- IMS - Navigation améliorée entre les enregistrements dans la mise en œuvre de la base de données IMS.
- IMS - CBLTDLI amélioré pour pouvoir lancer le programme utilise la purge.
- IMS - DFSRRC00 capable de transmettre les paramètres de groovy au programme principal.
- Ajout de la prise en charge de la commande JICS qui n'a pas été invoquée par le biais d'un TransactionRunner.
- JICS - Performances améliorées grâce à l'utilisation d'un cache configurable.
- BluSam - Ajoutez la possibilité de désactiver le préchauffage BluSam lors de l'ouverture afin d'améliorer les performances des grands ensembles de données.
- BluSam- Amélioration du comportement de suppression/renommage sur les ensembles de données ordinaires BluSam .
- BluSam - Performances améliorées lors des opérations d'enregistrement.
- Simplificateur de données amélioré pour les méthodes déterminant si une chaîne est de faible valeur.
- Support amélioré pour les problèmes liés à la décimale groupée et à l'ordre de tri.
- Configuration améliorée DB2 en tant que source de données principale avec AWS Secrets.
- FileSystem API améliorée pour exposer l'état du fichier.
- Entrée de flux de DynamicFileBuilder lecture améliorée avec LineSeparator.
- Simplificateur de données amélioré pour les méthodes déterminant si une chaîne est de faible valeur lorsqu'elle traite un jeu de caractères 0. CUSTOM93

- SQL - Traitement de sortie des procédures stockées SQL amélioré.
- SQL - Mappage lambda amélioré pour plusieurs tables avec des alias.
- COBOL - Support amélioré pour l'instruction LENGTH OF.
- COBOL - Ajout du support pour l'instruction TRANSFORM.
- COBOL - Ajout du support pour 9 nouvelles fonctions mathématiques.
- COBOL - Support amélioré pour INTEGER-OF-DAY FUNCTION.
- COBOL - Support amélioré pour le niveau 88 impliquant une valeur figurative.
- COBOL - Transformation améliorée pour l'instruction SET ADDRESS.

AS400

Nouvelles fonctionnalités

- Suppression des entités indicatrices dupliquées.
- Ajout du support pour les personnages DBCS.
- Ajout de la gestion du mot clé HELP pour le contrôle des enregistrements de sous-fichiers.
- Ajout d'un paramètre de configuration pour activer/désactiver la mise en majuscules du nom de colonne et diviser le contenu de la colonne de commentaires sur pipe char.
- Ajout de la prise en charge de l'utilisation de 0x0c comme dernier élément pour les champs de type Packed.
- RPG - Prototypes gérés déclarés avec ExtProc (« système »).
- CL - Le paramètre 'CLEAR' géré par cl-command RMVMSG + introduit des files d'attente de messages hors programme en mémoire.
- CL - Gestion des instructions génériques transmises aux appels SBMJOB CMD ().
- CL - Ajout des commandes STRCMTCTL et ENDCMTCTL. Mécanisme de verrouillage modifié et nettoyage des transactions et des serrures.
- CL - Ajout du support du paramètre RCDDLM pour la commande CPYTOIMPF.
- CL - Ajout de la gestion des zéros de remplissage dans la commande SAVOBJ.
- CL - Ajout de la gestion des bibliothèques incluses dans le nom qualifié du paramètre OBJ pour RTVOBJD.
- CL - Ajout du support pour les paramètres de commande CPYTOIPF STRDLM, STRESCCHR et RMVBLANK.

- CL - RTVMGS amélioré pour envoyer CPF24 07 et CPF2419 lorsque le fichier/l'identifiant est introuvable.
- CL - Commande RCVF améliorée pour recevoir des enregistrements de n'importe quelle bibliothèque fournie dans le paramètre DEV.

Améliorations

- Les valeurs par défaut de l'exécuteur de tâches Blu4IV ont été modifiées afin de permettre une meilleure mise à l'échelle par défaut.
- Parameterhelper modifié pour convertir la liste de chaînes en String. ElementaryRangeReference
- CTID amélioré pour gérer les colonnes inexistantes dans POSTGRE.
- Robustesse accrue pour prendre en charge l'API d'espace utilisateur « QUSPTRUS ».
- Ajout du support pour les espaces utilisateur APIs QUSRUSAT et QUSCUSAT.
- Support amélioré pour l'API User Space (QUSPTRUS) sans code d'erreur.
- Ajout du support pour CRON Job Scheduling à l'aide de Quartz.
- Support amélioré du cycle de programmation des RPG.
- Gestion des transactions Blu4iv améliorée.
- Le verrouillage des enregistrements des fichiers sous contrôle des engagements au cours d'une même transaction a été amélioré.
- Gestion améliorée de l'initialisation des sous-fichiers.
- Affichage amélioré des indicateurs de défilement pour les lignes de message.
- Empêche les zéros de suivre les numéros envoyés via la file d'attente de données.
- Écran d'informations supplémentaires sur les messages amélioré.
- Amélioration des opérations d'écriture JPA pour prendre en compte la bibliothèque actuelle.
- Comportement amélioré ProgramJobExecutor lors de l'exécution de programmes sans paramètres.
- Ajout d'une fonctionnalité permettant de transmettre directement les arguments des liens frontaux aux scripts principaux.
- Gestion des transactions améliorée pour les métadonnées des tâches.
- CL - Ajout du support pour le paramètre SECLVL dans RTVMSG.
- CL - Ajout d'une implémentation vide pour CLRLIB.
- CL - Support amélioré de CPYFRMIMPF pour la copie à la fois depuis une base de données et un fichier CSV.

- CL - Implémentation améliorée de CPYFRMIMPF pour ignorer les colonnes supplémentaires.
- CL - Amélioration de l'interprétation CPYTOIMPF et CPYFRMIMPF des anciens paramètres de formatage.
- CL - Ajout d'un paramètre removeDecimalPoint pour formater les valeurs numériques dans SAVOBJ.
- CL - Commande RCVF améliorée pour gérer correctement la condition EOF.
- CL - RTVSYSVAL - Implémentation SYSVAL = QDATETIME.
- CL - Commande OVRDBF modifiée pour obtenir le champ comme nom de table par défaut.
- CL - RTVJOBA Valeur non disponible pour le paramètre : USRLIBL.
- CL - Gestion des barres obliques principales dans le paramètre SNDPGMMSG MSGF.
- CL - Amélioration de la prise en charge des caractères génériques dans le fichier source dans la commande DSPFFD.
- CL - Gestion améliorée du paramètre PGMQ dans RCVMSG et SNDPGMMSG.
- CL - A rendu le paramètre RTVMSG MSG facultatif pour s'aligner sur les anciens documents.

Capacités transversales

Nouvelles fonctionnalités

- Fonctionnalité améliorée lors du passage du paramètre à la clause USING du curseur OPEN.
- Performances : amélioration de la pré-initialisation du contexte et du réglage RunUnit des performances.

Améliorations

- Amélioration du mécanisme de vidage des faibles valeurs à partir de la commande UNLOAD du programme utilitaire INFUTILB.
- Ajout de l'option de schéma actuel de support sur le gestionnaire secret des sources de données.
- Temps d'exécution amélioré pour ne pas prendre en compte les paramètres transmis lorsque le curseur est ouvert lorsqu'ils ne sont pas nécessaires.
- Validation du format numérique améliorée pour les champs numériques.
- Diagnostic SQL amélioré dans un environnement d'exécution hautement parallèle.
- Introduction de l'Unicode pour les séquences d'octets de pages de codes (FE FD).

- DataSimplifier optimisation des performances : instructions d'attribution améliorées.
- DataSimplifier optimisation des performances - Améliorez la valeur par défaut pour l'initialisation du type numérique afin d'éviter toute utilisation inutile. BigDecimal

Outils de modernisation, version 4.0.0

ZoS

Nouvelles fonctionnalités

- Ajout du support pour la gestion du programme Abend.
- Support amélioré pour générer un ensemble de données AIX.
- COBOL - Ajout du support pour la clause JUSTIFIED sur ALPHANUMERIC/ALPHABETIC/ GRAPHIC les champs.

Améliorations

- Amélioration de la gestion des attributs PURGETHRESH pour les définitions de ressources TRANSCLASS.
- Support amélioré pour la définition des données et l'instruction MOVE.
- CICS - Support amélioré pour la commande DELAY sur l'option MILLISECS.
- Mappage SQL Lambda amélioré pour plusieurs tables avec des alias.
- Support amélioré pour la recherche de champs par les parents.
- Paramètre SQLCA sqlstate amélioré pour les opérations COMMIT et ROLLBACK.
- COBOL - Améliorez l'analyse en commentant les paragraphes obsolètes
- COBOL - Support amélioré pour la clause REPLACING.
- COBOL - Ajout du support pour les fonctions mathématiques ASIN ACOS LOG TAN.
- COBOL - Ajout de la prise en charge de plusieurs instructions AFTER dans PERFORM VARIING.
- COBOL - Support amélioré pour les champs RENAMES (niveau 66).
- COBOL - Méthode LENGTH OF améliorée pour obtenir la longueur à un index spécifique dans un champ de tableau.
- COBOL - Ajout de la prise en charge de plusieurs clauses AFTER dans les instructions PERFORM VARIING.
- COBOL - Support amélioré pour la clause RENAMES.

- COBOL - Support amélioré du mot clé PICTURE.
- COBOL - Support amélioré pour l'analyse des champs de niveau 88.
- COBOL - Amélioration de la condition goto dépendante avec les éléments de données de table.

AS400

Nouvelles fonctionnalités

- Ajout d'une fonctionnalité permettant de transmettre des arguments aux appels Java directs du front-end.
- CL - Génération %SST améliorée, y compris le support de *LDA avec CL→Java.
- RPG - Ajout d'un enregistrement décrit par le programme de support pour les fichiers DISK.

Améliorations

- Fichier d'affichage amélioré, résolution des champs référencés avec le mot clé « REFFLD ».
- Support amélioré du mot-clé de fichier d'affichage SETOF-CSRLOC.
- Fichiers supprimés du contrôle des engagements après la fermeture.
- Comportement cohérent garanti pour les opérations de lecture et d'écriture simultanées sur une table lorsqu'elles sont effectuées par le même programme.
- Affectation gérée à la sous-chaîne de. SizePrefixedAlphanumericType
- Géré en passant la structure de données à la procédure avec un paramètre de chaîne de longueur variable.
- Amélioration de la rétention des valeurs numériques non valides lors de l'événement OnBlur et de la création d'écouteurs d'événements pour les champs valides uniquement.
- Messages d'erreur améliorés sur les écrans et mise en évidence des champs dont la saisie n'est pas valide.
- Gestion améliorée des champs d'écran conditionnés par des indicateurs.
- Défilement activé avec la molette de la souris.
- Ajout de la prise en charge des touches de fonction pour l'écran d'aide.
- Support amélioré pour le texte long dans le split-dynamic-field composant.
- Gestion améliorée des fichiers LF à enregistrements multiples lors du changement de nom des enregistrements.

- CL - Commande RTVJOBD améliorée pour gérer les fichiers LF (vues).
- CL - Commande OVRDBF améliorée lorsqu'elle est utilisée sur un LF à enregistrements multiples.
- RPG - Scénario géré dans lequel la procédure définit une variable portant le même nom que le paramètre renommé.
- RPG - Amélioration de la gestion de *ZEROS lors de l'initialisation de Signed BinaryInteger.
- RPG - Gestion améliorée des pointeurs vers des variables non locales (de référence).
- RPG - Amélioration de la gestion des instructions ELSEIF après les IFxx instructions.
- RPG - Ajout du support pour les champs définis avec LIKE sur le prototype.
- RPG - Amélioration de la prise en charge du mot clé LIKE dans un champ créé par LIKEREC.
- RPG - Amélioration de la génération d'opérateurs avec des figuratifs.
- RPG - Amélioration de l'analyse de l'expression matricielle xxx (\ *) et prise en charge de cette expression dans %lookup.
- RPG - Code de LookUp fonctionnement amélioré avec des indicateurs élevés et égaux (ou faibles et égaux).
- RPG - Analyse des formulaires libres améliorée.
- RPG - Analyse améliorée des constantes nommées i-Card qui suivent les formats d'enregistrement i-Card.
- RPG - Support amélioré pour les types INTEGER et UNSIGNED.
- COBOL - Ajout d'une clause de support INDIC au format DSPF dans l'instruction COPY DDS.
- COBOL - Grammaire améliorée pour les instructions DISPLAY et ACCEPT afin de débloquer la transformation et la génération.
- COBOL - Ajout du support pour les fichiers DISK.
- COBOL - Programmes de support améliorés pour les fichiers d'affichage DDS.
- COBOL - Ajout du support pour la clause LIKE.
- COBOL - Ajout du support pour le fichier DISK décrit par le programme.
- COBOL - Ajout du support pour le nom de fichier avec suffixe.

Capacités transversales

Nouvelles fonctionnalités

- Géré le chargement différé des composants cartographiques des projets Web.

Améliorations

- Génération Java améliorée des paramètres des indicateurs SQL.
- Capacité améliorée à gérer les variables impliquées dans l' DB2 instruction SET.
- Amélioration de l'augmentation du taux d'erreur à la fin du curseur extrait lorsque la sortie est un tableau à une seule entité.
- Chemin géré sous Linux.
- Data Migrator gère les vulnérabilités et supprime les dépendances inutilisées.

Notes de mise à jour 3.10.0

Cette version des outils d'exécution et de modernisation de AWS Blu Age est axée sur les principales mises à niveau et améliorations de base du produit, dans le but d'améliorer les performances et la robustesse à toutes les étapes de transformation et d'exécution. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Mise à niveau de version de Java 8 vers Java 17, ce qui augmente la sécurité et les performances, et permet aux clients de déployer et d'exécuter des applications implémentées dans un langage plus moderne et d'utiliser des versions récentes de framework tiers.
- Support supplémentaire pour la gestion de grands espaces de mémoire partagée entre les utilisateurs ou les tâches, ainsi que pour le stockage de données réutilisables après le redémarrage de l'application ou de l'instance.
- Accès plus rapide à de grands ensembles de données dans Blusam grâce à un mécanisme de pagination qui permet de récupérer un sous-ensemble d'enregistrements de manière incrémentielle.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.10.0

Ce runtime est basé sur Java17, Spring2.7 et Angular16.

ZoS

Nouvelles fonctionnalités

- Blusam - Ajout du support pour les grands ensembles de données grâce à un mécanisme paginé dans lequel les index sont stockés et chargés à l'aide de pages

Améliorations

- DataUtils.compare amélioré pour gérer la conversion de priorité inférieure d'une chaîne en nombre
- Ajout du support pour vérifier que non ByteRange est créé avec des valeurs incorrectes via la propriété YML DataSimplifier. byteRangeBoundsVérifiez
- RemoveSosi () amélioré pour prendre en charge l'initialisation d'un caractère vide GraphicAlphanumericType
- Robustesse accrue pour le fonctionnement des tâches et lecture sécurisée de l'état GDG
- Blusam - Ajout du support pour effacer le cache des ensembles de données Blusam via une nouvelle méthode nommée .removeCache () CoreBluesamManager
- Blusam - Amélioration du comportement de suppression/renommage pour les ensembles de données Blusam classiques
- Redis - Support amélioré pour le déverrouillage des ensembles de données et la suppression du verrouillage des enregistrements
- JICS - Amélioration du message d'erreur en cas d'échec des demandes
- JCL - Ajout de la prise en charge de la concaténation de variables ControlM basée sur le caractère point
- JCL - Ajout du support pour Write ADVANCING (ADV) pour les fichiers GDG
- JCL - Support amélioré pour le numéro de génération actuel après suppression de tous les fichiers GDG
- JCL - Support amélioré pour la lecture de RDW/RecordSize à partir du catalogue lors de la création du jeu de données
- JCL - Ajout du support pour mettre à jour l'objet de ressource (depuis AbstractSequentialFile) lors de l'ouverture du fichier avec la taille de l'enregistrement de sortie de données
- JCL - Performances IDCAMS améliorées
- JCL - Support amélioré pour PRINT STATEMENT en ajoutant « CHAR » comme alias de « CHARACTER »
- SORT - Support amélioré pour les opérations de copie d'un ensemble de données de longueur fixe Blusam vers un ensemble de données de longueur variable
- SORT - Grammaire de tri améliorée pour gérer certaines instructions spécifiques

AS400

Nouvelles fonctionnalités

- Ajout de la prise en charge des espaces utilisateurs et de ses applications associées APIs
- Ajout de la prise en charge du paramètre TOMSGQ de SNDPGMMSG et mise en œuvre des files d'attente de messages
- CL - Ajout du support pour les paramètres FILE et SPLFNAME pour la commande OVRPRTF
- CL - Ajout du support pour la gestion des bibliothèques pour la table de partition correspondante avec la commande CPYF
- CL - Ajout de la prise en charge de la gestion de la commande CHGCURLIB et de la prise en compte de la bibliothèque actuelle lors de la création de requêtes
- CL - Ajout du support pour la gestion de la commande cl dans le cadre de l'appel stacktrace

Améliorations

- Amélioré MessageHandlingBuilder pour une meilleure gestion de l'entrée de trace de la pile d'appels
- Exécution parallèle améliorée de la fonctionnalité ContextPreconstruct
- Attributs d'affichage améliorés lorsqu'un enregistrement est créé par SFLINZ
- SAVOBJ amélioré pour permettre la gestion de plusieurs fichiers de sortie
- Amélioration de la gestion des programmes groovy en les ajoutant programCallStack lorsqu'ils sont appelés depuis un programme Java
- Détection améliorée du positionnement supérieur du modal d'aide
- Fonctionnalité TopGMQ améliorée lorsque le paramètre TomSGQ est fourni pour SNDPGMMSG
- Récupération améliorée des messages prédéfinis et fonctionnalité du chargeur de messages
- Amélioration de la gestion par CPYTOIMPF des caractères séparateurs dans le contenu
- Verrouillage de déverrouillage amélioré sur l'enregistrement READ

Capacités transversales

Nouvelles fonctionnalités

- Ajout d'une traduction pour les messages système sur le Front-End

- Ajout d'une nouvelle méthode ExecutionContext pour renvoyer la pile d'appels du programme
- Définissez un séparateur de lignes (pour simplifier les données) quel que soit l'environnement réel
- Ajout de la possibilité de configurer le chemin JSON du modèle SQL

Améliorations

- Amélioration de la méthode de comparaison DataUtils. compareAlphInt() lorsqu'il s'agit de rembourrage
- Création d'un drapeau pour autoriser un comportement personnalisé en cas d'exception dans les requêtes de curseur
- Conversion graphique améliorée de la base de données LOWVALUES

Troisième partie

- Mise à niveau pour atténuer les risques CVE-2024-21634, CVE-2023-34055, CVE-2023-34462, -JAVA-ORG-SPRINGFRAMEWORKSECURITY-5905484, CVE-2023-46120, CVE-2023-6481, CVE-2023-6378, CVE-2023-5072) IN1

Outils de modernisation, version 3.10.0

ZoS

Améliorations

- COBOL - Ajout du support pour la fonction ABS
- JCL - Étendue variable améliorée : attachée à STEP au lieu de JOB
- Injection améliorée des paramètres du curseur pour les valeurs faible/élevée
- Analyse CSD améliorée, notamment pour les TRANSACTIONS à distance

AS400

Améliorations

- Vérification en blanc supprimée pour l'indicateur de niveau de contrôle
- Ajout de la prise en charge du nom externe pour les mots clés IMPORT/EXPORT
- Ajout du support pour %LEN sur les champs

- CL - Ajout du support pour les nouveaux opérateurs pour le langage CLLE
- CL - Ajout du support pour les IF imbriqués
- COBOL - Gestion améliorée de la commande START lorsqu'elle est utilisée avec plusieurs touches
- DSPF - Gestion améliorée de la position du curseur avec un numéro d'enregistrement
- DSPF - Amélioration du formatage pour les champs numériques signés, les champs numériques uniquement et les champs à grande échelle
- DSPF - Amélioration de la détermination du titre pour Screen General Help
- DSPF - Support amélioré des spécifications d'entrée/sortie
- DSPF - Gestion améliorée des séparateurs de regroupement lors de la validation d'un champ numérique
- Sortie de mappage/enregistrements DDS améliorés
- Amélioration de la capacité du mot clé REFFLT du fichier d'imprimante à résoudre les champs référencés
- RPG - Support amélioré pour les déclarations « TOUTES gratuites »
- RPG - Analyse des conditions améliorée et prise en charge ajoutée de la gestion de CABXX sans résultat TAG
- RPG - Gestion améliorée des spécifications d'entrée des champs numériques
- RPG - Gestion améliorée des appels de procédure dans les conditions IF/ELSEIF/WHEN
- RPG - Gestion améliorée de la commande READ lorsqu'elle est appelée sur un fichier dspf
- RPG - Améliorer la prise en charge des fichiers faisant référence à un DDS inexistant
- Améliorez la gestion du REFFLD lorsqu'un nom de format d'enregistrement physique est transmis
- Ajout du support pour utiliser « return » comme nom de colonne de base de données

Capacités transversales

Nouvelles fonctionnalités

- Oracle - Permet de définir des utilisateurs plutôt que SYS pour stocker les fonctions intégrées

Améliorations

- Version Java améliorée de v8 à v17
- Condition SQL améliorée avec le nom de colonne du cluster

- Ajout de la prise en charge des clauses ORDER BY depuis la vue

Notes de mise à jour 3.9.0

Cette version de AWS Blu Age Runtime and Modernization Tools met l'accent sur de multiples améliorations transversales apportées au produit dans le but d'améliorer les performances des architectures à haute disponibilité, ainsi que sur de nouvelles fonctionnalités permettant d'améliorer l'exécution des tâches à un niveau supérieur. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Mise à niveau de la version d'Angular 13 vers Angular 16, renforçant la sécurité et donnant accès à de nouvelles fonctionnalités qui améliorent les performances des applications en ligne des clients.
- Ajoutez la prise en charge des fonctionnalités intertâches dans AS4 00, avec notamment le fait que les offres d'emploi peuvent envoyer des messages de demande de manière synchrone entre elles, ce qui permet le découplage dans les tâches modernisées.
- Améliorations des performances liées à l'utilisation de Redis, notamment l'optimisation du pool de connexions, une sécurité de connexion élevée et un mécanisme de verrouillage des ensembles de données amélioré.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.9.0

ZoS

Nouvelles fonctionnalités

- Programme de tri : entrées VSAM mises à jour avec une longueur fixe
- JHDB DB : ajout d'un délai d'attente configurable

Améliorations

- Support amélioré du séparateur de lignes à diffuser en continu s'il est utilisé dans la concaténation de fichiers
- Support amélioré pour ouvrir des fichiers séquentiels concaténés. Initialiser DataSetIndex après ouverture du fichier

- Support amélioré pour le séparateur décimal virtuel lorsque a NumericEditedType est affecté à une valeur numérique
- Support amélioré pour NumericEditedType les valeurs non négatives
- IDCAMS : les cartes SYSIN sont désormais lues à l'aide de la propriété « encoding » définie dans le fichier .yml application-utility-pgm
- IDCAMS : grammaire mise à jour pour prendre en charge l'argument FILE (..) dans l'instruction DEFINE CLUSTER
- INFUTILB : Ajout de la prise en charge de l'argument DFSIGDCB pour remplacer les paramètres DCB de DD SYSREC
- INFUTIL : prise en charge améliorée du paramètre « DFSIGDCB YES »
- SPLICE amélioré pour gérer un énorme fichier d'entrée
- DFSORT : Gestion améliorée des champs de commentaires
- DFSORT : Ajout du support pour le format numérique libre (signé/non signé) (SFF/UFF)
- SORT : Ajout du support d'analyse pour les instructions OPTION PRINT et OPTION ROUTE
- SORT/ICEMAN : Ajout du support pour les opérations de division fermées (champ avec opérateur DIV)
- Support amélioré pour CICS READ à l'aide d'une clé générique
- Correction de la fonction StringUtils .chargraphic pour supprimer le SOSI d'un type graphique
- Améliorez les performances sur DataUtils. isDoubleByteEncodage
- JCL : prise en charge améliorée du mode de disposition KEEP pour un ensemble de données temporaire. Le système change la disposition en PASS
- JCL : gère les paramètres DCB de manière dynamique
- JCL : sorties SUM FIELDS améliorées pour les valeurs incorrectes
- JCL : Common DDUtils : :getContent recherche désormais le RecordSize dans le catalogue
- JCL : lecture des attributs RDW/RecordSize du catalogue lors de la création du jeu de données
- JCL : Ajout du support pour DCB=.MYDD pour copier les paramètres DCB d'un DD dans un autre au cours de la même étape
- JCL : système d'héritage amélioré de la taille des enregistrements
- JCL : ajout d'un verrou de jeu de données exclusif (Redis)
- Redis : ajout du support SSL pour le mode autonome
- Redis : ajout du nombre de verrous Redis synchronisé avec le verrou

- Redis : paramètres de pool pris en charge pour le verrouillage Redis
- Redis : actualisation optimisée des métadonnées avec Redis
- Redis : support amélioré du cluster Redis
- Amélioration des verrous ouverts avec le mode IO
- Les ensembles de données améliorés verrouillent les performances et suppriment les verrous inutilisés
- Chemin amélioré de l'ensemble de données lors du désenregistrement du fichier
- Invalidation améliorée du cache de fenêtre avant la lecture
- Ajout de la prise en charge de l'utilisation des fournisseurs de sources de données utilitaires Thread Safe
- Contrôle de nullité amélioré de DatasetState
- Support amélioré pour ne pas rouvrir les ensembles de données déjà ouverts
- Robustesse accrue pour le fonctionnement final de la tâche
- Support amélioré pour l'ordre des index, des clés, permettant les doublons
- Support amélioré pour l'ordre de sérialisation des listes à ignorer
- Ajout de la prise en charge de la fonction de vidage de bogues pour aider à diagnostiquer les problèmes d'ordre des index
- Support amélioré pour l'actualisation des métadonnées
- Support amélioré pour la lecture en bloc sur Blusam

AS400

Nouvelles fonctionnalités

- Crée un registre du contexte de l'application
- Support du mot clé DSPF CLRL (NO) Support de la surveillance des blocages d'enregistrements
- Support pour keyed DataQueue
- Support pour les messages de demande pour les tâches par lots
- Ajout du support pour le fichier d'imprimante décrit par le programme pour 00 COBOL AS4
- Gère la commande RMVJOBSCDE cl
- Amélioration pour RUNSQL/DLYJOB

- CHKOBJ : augmentation du code d'erreur existant pour le paramètre LIB
- SNDPGMMSG : prend en charge les paramètres de chaîne
- RTVDTAARA : sous-chaîne améliorée dans LDA
- DSPFD : paramètre FILE pris en charge ajouté pour un nom de fichier spécifique
- RUNQRY : Support du fichier SQL dans QRY PARAM
- CRTDUPOB : Support pour copier les données entre les zones de données
- SBMJOB : convertit les instructions à utiliser JobQueueManager
- OPNQRYF : Ajout du support pour la bibliothèque Qtemp
- CRTDUPOBJ : Logique améliorée pour copier le contenu d'une partition
- CRTDUPOBJ : Ajout du support pour Qtemp pour les vues
- RTVSYSVAL : Support de la valeur SYSVAL, QDATFMT dans la commande CL
- CHKOBJ : Ajout du support pour OUTQ
- RTVJOBA : Supporte le paramètre SWS
- SNDPGMMSG et RCVMSG : paramètres supplémentaires pris en charge MSGF, MSGFLIB, MSGDTA, MSGTYPE, KEYVAR, MSGKEY, MSGID

Améliorations

- Supports améliorés des cartes d'E/S WORKSTATION
- Gestion améliorée du message défini superposé au message précédent
- Supporte des informations de message supplémentaires sur array-messageline
- Accès amélioré aux wrappers de tableaux autonomes dans EVAL, SortA et figuratives
- Améliorez DAOs le nettoyage à la fin de l'application en ligne
- Ajout de la prise en charge de formats de date supplémentaires et amélioration de la gestion des entrées de chaîne
- Amélioration de la gestion CVTDAT de SYSVAL en ajoutant la classe d'assistance aux valeurs du système Decode et des paramètres de construction à partir de la commande CL SbmJob
- Le package com.netfective.bluage.gapwalk.rt.blu4iv a été supprimé de l'analyse des composants gapwalk-cl-command
- Amélioration de la prise en charge des messages prédéfinis pour l'API de file d'attente de messages

- Amélioration du support retrieveSubfileRecord pour les enregistrements écrits dans un autre programme
- Amélioration de la prise en charge des messages immédiats pour l'API de file d'attente de messages
- Gestion améliorée de la zone de données locale lors de la soumission d'une offre
- Démarre JobQueues automatiquement au démarrage du serveur
- Utilise la configuration ApplicationContext pour décoder les paramètres de SBMJOB
- Amélioration des messages d'erreur fournis par le système
- Permet à RTVMSG de rechercher des fichiers .properties dans des sous-répertoires imbriqués
- Gère la réinitialisation des entités liées à des pointeurs incorrects/invalides
- Amélioré MessageHandlingBuilder pour afficher MsgID et le MsgFile nom sous forme de chaînes pour RCVMSG
- Méthode de withMsgFile nom améliorée de l'API de mise en file d'attente des messages
- Mécanisme de verrouillage de la zone de données amélioré
- RTVMBRD : Support des minuscules et des majuscules pour le paramètre FILE
- CRTDUPOBJ : Gestion améliorée des vues
- CPYTOSTMF : Gestion améliorée de la connexion
- CPYF : amélioration de la gestion du nom du répertoire lors de la copie à partir d'un fichier plat
- RCVF : gère correctement les paramètres DEV/RCDFMT et la transformation de RCDFMT pour Groovy et Java
- RCVF : gère les appels suivants et évite de réinitialiser le curseur
- CPYF : Ajout du support pour l'écriture à partir de fichiers plats
- CRTDUPOBJ : Ajout de la gestion du nouvel obj avec la bibliothèque Qtemp
- CHGDTAARA : augmentation de la longueur maximale de la zone de données de 256 à 2000
- SAVOBJ : Assurez-vous que les enregistrements enregistrés sont dans l'ordre d'insertion
- RTVDTAARA : Valeurs récupérées (à ne pas découper)
- CHKOBJ : renvoie les messages de surveillance corrects lorsque le membre n'existe pas
- RTVDTAARA : Ajout du support de la sous-chaîne LDA
- RTVDTAARA : renvoie des espaces blancs jusqu'à la longueur de la variable spécifiée dans le paramètre RTNVAR

- RTVDTAARA : prend en charge les paramètres entiers pour le début et la longueur et prend en charge le dernier format de transformation
- CHGDTAARA : Ajout de la prise en charge des paramètres qui incluent les limites inférieures et supérieures
- CHKOBJ : gère la valeur VIEW pour le type d'objet paramètre
- CHKOBJ : résultat défini sur vrai quel que soit le membre si la vue existe

Capacités transversales

Nouvelles fonctionnalités

- Gère la génération de rapports dans des fichiers .txt
- Ajout de la propriété de source de données CurrentSchema XA au gestionnaire de secrets
- Ajoutez la propriété YAML database.cursor.raise.already.opened.error pour permettre au framework de générer l'erreur SQLCODE 502 lorsque le curseur est déjà ouvert

Améliorations

- Ajout de pompons Gapwalk à l'emballage AWS Blue Age sur Amazon EC2
- Utilise le nouveau paradigme du gestionnaire de signaux par défaut
- Ajout d'un support pour le verrouillage lorsque la disposition est MOD ou ANCIENNE
- Ajout d'un cache pour stocker les modèles de date/heure de la base de données
- Fonction de contrôle améliorée de PackedType
- Améliorez les fonctions DataUtils .setTo pour les enregistrements avec VariableSizeArray
- Gère l'option MQ SYNCPOINT en ce qui concerne l'unité d'exécution
- Framework activé pour définir le SQLCODE lors d'une transaction d'annulation
- Ajout d'un nom de classe de pilote automatique en fonction du secret de la clé du moteur
- Délai d'expiration du programme/de la transaction
- Restaurer la position du curseur après le rollback lors de l'accès au curseur

Troisième partie

- Mettez à niveau les SDK SnakeyAML, Redisson et Amazon, supprimez YamlBeans (atténuez les problèmes CVE-25857, CVE-2023-24621, CVE-2023-42809, CVE-2023-44487)

Outils de modernisation, version 3.9.0

ZoS

Améliorations

- Support amélioré pour XML-TEXT en tant que source pour une cible de type String
- Flux de travail STM vers UML amélioré pour prendre en charge le modèle de division X/ (Y/Z)
- JHDB DB : accepte l'appel ROLLBACK avant toute mise à jour de base de données
- JHDB DB : accepte ROLLBACK même si la transaction est terminée (NOP)
- JCL : fonction de validation des étapes améliorée
- SORT : gère la fonction SUM avec des valeurs négatives décimales de zone
- COBOL : ajoute la prise en charge de l'échappement entre guillemets simples ou doubles dans les chaînes littérales

AS400

Améliorations

- Fonction intégrée %editc améliorée : gestion du code d'édition X en ajoutant des zéros en tête
- Gestion améliorée de la valeur initiale des champs en entrée uniquement
- Ajout de touches d'action pour les boîtes de dialogue d'aide
- Enregistrement du pied de page du tableau dynamique apparaissant en bas
- Commande START gérée sans KEY PHASE pour les fichiers qui spécifient une clé d'enregistrement réelle
- Valeur par défaut ajoutée pour les types float et NumberUtils ::pow
- Ajout du support pour définir une variable à l'aide de LIKE (IN)
- Gestion de la boucle FOR mise à jour pour permettre l'omission d'éléments facultatifs
- Analyse RPG mise à jour pour associer les enregistrements au nom du tableau CTDATA
- Gestion améliorée des indicateurs pour les CABxx relevés
- Supporte le paramètre facultatif sur le mot clé COMMIT
- Amélioration de la prise en charge des mots clés FORMAT dans LF
- Code d'opération LOOKUP géré avec indicateurs élevés et égaux (ou faibles et égaux)

- Nom de clé PF géré déclaré entre guillemets
- Amélioration de la gestion de l'EDTCDE X pour ne pas supprimer les zéros en tête
- Amélioration de la prise en charge de MSGCONE dans le fichier d'imprimante, sans générer d'étiquettes anonymes
- Le CONTENU des champs est partagé par plusieurs structures de données
- Paramètre ERRSFL géré en combinaison avec SFLMSG/SFLMSGID
- Code principal amélioré avant la portée de la déclaration du processus d'un RPG entièrement gratuit
- Spécification de contrôle conditionné par analyse syntaxique ajoutée
- Support amélioré pour la méthode setErrSfl () dans le dataholdermapper
- Résolution de type améliorée pour les variables créées en interne
- Support amélioré pour l'opcode Z-ADD
- Amélioration de la gestion du champ constant avec une valeur DFT
- Améliorez la prise en charge du champ entier dans le statut du programme ds
- Affectation d'indicateurs gérée dans les paramètres ENTRY
- Amélioration du filtre des mots clés propagés via le mot clé ref/reffield
- Structure de DataArea données sans nom prise en charge
- Gestion améliorée du type de données du pointeur
- Eléments manipulés du tableau utilisés pour définir des variables avec le mot clé LIKE : accès au tableau dans le champ de sortie
- Support amélioré pour le numérique signé, affichage uniquement des chiffres
- Relation logique prise en charge sur la carte O
- Cas de test pour %CHAR en caractères alphanumériques
- Spécification de contrôle prise en charge (mot-clé principal)
- EDTCDE avec deux paramètres dans le fichier d'imprimante
- Analyse FullFree RPG améliorée
- Amélioration du tableau dynamique pour garantir le positionnement correct du pied de page
- Ajout du support pour l'initialisation des types numériques avec TOUTES les constantes figuratives
- Gestion améliorée de plusieurs fichiers logiques RPG faisant référence au même fichier physique

- Améliorez la détection des champs modifiés sur un écran moderne
- Synchronisation modale avec des champs dynamiques
- Amélioration de la gestion du champ numérique signé en sortie uniquement
- Améliorez la prise en charge des cartes d'E/S WORKSTATION

Capacités transversales

Nouvelles fonctionnalités

- Outil de migration de données : ajout de la propriété ebcdicFilesWith VarcharIn VB pour permettre de prendre en compte la longueur VARCHAR de 2 octets lors de la lecture d'octets
- Implémentation d'une API commune pour enregistrer les erreurs
- Implémentation BluAgeErrorDictionaryUtils et utilisation d'une API commune pour enregistrer les erreurs et/ou les informations dans COBOL2 Model, RPGCycle Builder, Definitions2Model et FieldsProcessor
- Grammaire SQL améliorée pour prendre en charge différentes définitions de clauses d'isolation

Améliorations

- Version Angular améliorée vers la version 16
- Angular : version ajv améliorée de 6 à 8.9

Troisième partie

- Groovy mis à jour vers la version 2.4.15

Notes de mise à jour 3.8.0

Cette version de AWS Blu Age Runtime and Modernization Tools est axée sur de multiples améliorations transversales du produit afin d'améliorer sa qualité et sa sécurité, ainsi que sur l'amélioration des performances de mise en cache et sur l'unification des supports de commandes dans une distribution unique. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Mise à niveau de version de Spring 2.5 vers Spring 2.7, augmentant le support de maintenance, les performances et la sécurité de la plateforme.

- Unification de la prise en charge de plus de 82 commandes CL dans le cadre de la over-the-counter distribution afin de faciliter l'utilisation et le déploiement d'applications modernisées qui utilisaient auparavant des scripts CL.
- Nouveautés APIs disponibles pour mieux fonctionner et interagir avec les ensembles de données BluSam, telles que l'importation intégrée vers le service géré et la capacité de répertorier les informations de métadonnées des ensembles de données.
- Améliorations des performances et extension de l'utilisation de Redis, y compris la disponibilité en mode cluster, la récupération de données à haute disponibilité, la standardisation de l'utilisation des secrets.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.8.0

ZoS

Nouvelles fonctionnalités

- Gestion de la définition de la clé sous forme de chaîne pour DynamicFileBuilder
- DFSORT : Ajout de la prise en charge des éléments multiples dans OUTFIL TRAILER1 + Initialisation de la grammaire DFSORT
- DDUtils Outil courant : gestion de la taille des enregistrements dans les données in-stream
- Fichier indexé : manipulation de l'option GENKEY

Améliorations

- Services de chargement BluSam externalisés dans un fichier jar séparé
- Ajout de la prise en charge de la configuration de l'emplacement pour le stockage des fichiers temporaires
- Mécanismes de cache partagé améliorés pour les cas impliquant plusieurs nœuds
- Utilisation du cache partagé : IDCAMS vérifie l'optimisation
- Améliorez l'injection ROWID pour la sélection intégrée
- JCL : chaque procédure de travail in-stream est désormais générée dans un fichier Groovy distinct
- Assurez une couverture de card-demo-v 2 % sur les cartes IDCAMS JCL

- BluSam : évitez les doublons lors de l'utilisation de plusieurs instances
- Réduction de l'encombrement mémoire lors de l'hydratation du cache
- Support de configuration du pool Jedis
- Séparateur de ligne ajouté au flux s'il est utilisé dans la concaténation de fichiers
- Support des cartes EBCDIC + commentaires de bloc (/*.../) dans l'utilitaire IDCAMS
- Requête de support de base de données : prise en charge des chaînes à double octet lors de la conversion du niveau 49 en SQL
- Grammaire DFSORT : implémente 17 instructions de contrôle + intégration de 2 d'entre elles (OMIT/INCLUDE)
- Améliorez les colonnes GRAPHIC, récupérez INFUTILB
- Support pour la lecture de fichiers avec tableau de taille variable
- Support pour la signature ZonedType par nibble où le premier bit du dernier octet est « E »
- DFSORT/ICETOOL ajoute la prise en charge de l'argument NOMATCH =(..) si un enregistrement ne correspond à aucune des constantes de recherche CHANGE
- Compatibilité avec Redis Cluster
- Gestion du statut du job (échec) en fonction d'un code de sortie groovy
- Support amélioré de CICS SYNCPOINT ROLLBACK.
- Fenêtre de pré-extraction pour optimiser l'utilisation du cache Redis
- JCL/GROOVY : hérite de la propriété IsRDW de l'ensemble de données de l'étape précédente lorsque DISP= (, PASS)
- Gestion de copies partielles de données avec un tableau de taille variable

AS400

Nouvelles fonctionnalités

- Support pour les cartes d'E/S pour l'affichage de fichiers
- Support pour les informations de message supplémentaires pour les mots clés DSPF ERRMSGID et CHKMSGID
- Support pour plusieurs messages d'erreur sur l'écran frontal
- Ajout ou amélioration de la prise en charge de 82 commandes CL dans l' gapwalk-cl-commandapplication

Améliorations

- Support amélioré pour DELETE et READ sous contrôle des engagements
- ConvertDate à l'intérieur du %dec intégré
- En-têtes de sécurité XSS renforcés
- Robustesse et cohérence améliorées de la génération STM (meilleure gestion de : ligne de suite dans un rpg en format libre, virgules pour la partie décimale, blocs de forme libre dans la définition/déclaration)
- DataHolderMapper Génération améliorée
- Robustesse accrue et possibilité de modification dans DataAreaFactory
- Amélioration du changement de focus sur la touche de tabulation
- Performances améliorées lors de la génération de rapports Jasper
- Affichage décimal amélioré avec rembourrage 0
- Support amélioré pour le champ ROW/COL dans INFDS
- Améliorez la prise en charge des champs modifiés depuis l'écran
- Ajout de getters pour le nom et le chemin du rapport généré
- Amélioration de la longueur de la file d'attente de données
- Configuration automatique améliorée des files d'attente de tâches pour répondre aux nouvelles normes de Spring Boot 2.7
- Mises à jour améliorées du poste de travail pour plusieurs sessions simultanées

Capacités transversales

Nouvelles fonctionnalités

- Support de l'absence de tolérance de données non valides pour Packed
- Pagination/filtrage ajoutés pour répertorier les points de terminaison des jeux de données

Améliorations

- Stratégie de transformation des requêtes ORACLE améliorée lors de la comparaison de colonnes par rapport à une chaîne vide
- Gestion du BLOB à l' DB2 aide des programmes utilitaires DSNTEP et INFUTILB. Les BLOB DB2 sont désormais modernisés vers des postgres de type BYTEA.

- Amélioration de la suppression du dernier élément du curseur
- Support amélioré pour la suppression du fichier RRDS
- Performances améliorées de AWS Blusam Secret
- Gestion améliorée des connexions aux bases de données dans le framework SQL
- Clés de gestion de secrets normalisées pour AWS plusieurs sources de données
- Correctifs de régression des performances
- Fonction de vérification améliorée pour PackedType
- Gestion améliorée de LOW-VALUE pour PackedType
- Emballage de sécurité à ressort amélioré pour la connexion Cognito
- Ne pas appliquer le codage et le décodage par point de transfert de code sur les bases de données ciblées DB2

Troisième partie

- Mise à niveau de Spring Boot de la version 2.5 à la version 2.7

Outils de modernisation, version 3.8.0

ZoS

Nouvelles fonctionnalités

- JCL : Gestion du flux avec le retour du chariot « \ r »

Améliorations

- Journalisation améliorée pour empêcher la division par zéro lors de la modernisation d'une clause DIVIDE with ON SIZE ERROR
- JCL : support amélioré pour appeler une procédure dans une procédure
- Support du mot clé OF dans la commande FORMATTIME CICS en cas de champs ambigus
- JCL : prise en charge du caractère Â¥ dans les variables
- JCL : calcul de la RC en fonction des étapes précédentes
- Comparaison d'octets au lieu de chaînes lorsque PL1 SUBSTR est utilisé

- Amélioration de l'initialisation de tableaux multidimensionnels à partir d'une source unique
- Analyse syntaxique améliorée du COBOL lorsqu'il implique une seule requête SQL dans un bloc IF

AS400

Nouvelles fonctionnalités

- Support de l'instruction IF imbriquée dans CL
- Support amélioré pour l'instruction ENDDO dans RPG Freeform

Améliorations

- Support amélioré pour le niveau de contrôle du conditionnement
- Retour du prototype amélioré avec LIKE
- Support amélioré pour les fonctions de gestion %months, %year, %days
- Support de la fonction d'aide pour l'ensemble de l'écran
- Gestion des BLANKS figuratifs passés en paramètre
- Amélioration de l'expression EVAL avec l'opérateur « »
- Gestion de la commande START sans KEY PHASE
- Amélioration de la gestion du mot clé LIKEREC
- Amélioration des sous-champs anonymes
- Amélioration de la procédure renvoyant un type non signé
- Support amélioré pour le fonctionnement RESET (RPG gratuit), les fonctionnalités intégrées %CHAR et %DEC
- Amélioration de la fonction intégrée %LOOKUPXX
- Support amélioré pour le mot clé LIKEDS sur la procédure sans prototype
- Gestion du type de tableau de mots clés Dim (VAR, AUTO)
- Support amélioré pour XFOOT
- COBOL : prise en charge améliorée des champs RENAMES
- CL : support en condition (vraie)
- Amélioration de la gestion des tableaux autonomes avec le mot clé LIKE
- Amélioration de la fonction intégrée %INT

- Analyse complète et gratuite du RPG amélioré
- Support amélioré pour le tableau dans la liaison
- CL2GROOVY : Déclaration Support Select
- Amélioration du mot clé DSPF « ERRMSGID »
- Amélioration de la gestion de l'initialisation des octets avec des zéros en tête
- Amélioration des valeurs autorisées pour les champs numériques
- Manipulation de l'extendeur H pour une instruction EVAL au format libre
- CL to Groovy : Support de la sous-chaîne LDA
- Support amélioré pour la réinitialisation sur un enregistrement
- Amélioration de la gestion de l'EDTCDE et de l'EDTWRD avec les références
- Mappage des champs de saisie amélioré avec les champs DDS
- Support amélioré pour déplacer un personnage dans le tableau IN
- Amélioration du prototype avec le mot clé LIKEDS
- Support amélioré pour le mot clé DSPF DSPATR
- Analyse améliorée de la carte D avec +/-
- Robustesse accrue dans les appels de programmes
- Robustesse accrue dans le processus de résolution sur le terrain

Capacités transversales

Améliorations

- FrontEnd: simuler un événement de collage pour une entrée IME

Troisième partie

- Mise à niveau de Spring Boot de la version 2.5 à la version 2.7

Notes de mise à jour 3.7.0

Cette version des outils d'exécution et de modernisation de AWS Blu Age inclut principalement des améliorations visant à mieux prendre en charge les commandes et les utilitaires, des fonctionnalités

d'intégration à AWS Secrets Manager et de nouvelles fonctionnalités de surveillance. Certains des principaux changements apportés à cette version sont les suivants :

- Plusieurs composants d'exécution peuvent désormais utiliser AWS Secrets Manager pour améliorer la configuration de sécurité des applications modernisées, principalement liées aux sources de données des services publics, aux files d'attente Redis pour TS, au BluSam cache et aux verrous.
- Point de terminaison de surveillance qui permet de récupérer les métriques des transactions, des lots et de la JVM pour l'optimisation de l'utilisation des ressources et la gestion opérationnelle, telles que le statut, la durée, le volume, etc.
- Nouvelles fonctionnalités pour prendre en charge les appels IBM MQ dans les RPG et augmentation de la couverture de transformation de JCL SORT et IDCAMS.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.7.0

Rubriques

- ZoS
- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- Améliorez l'analyse des requêtes impliquées dans les applications utilitaires du programme en utilisant du langage SQL similaire à la grammaire. (V7-9401)
- Gérer un tableau de taille variable indexé lorsqu'il est décalé (V7-9904)
- Support de la colonne INSERT SQL TIME au DB2 format 24:00:00 (V7-10023)
- Support de la requête SQL INSERT à partir de tableaux avec les options FOR ROWS et ATOMIC (V7-10105)
- JCL SORT - amélioration TranscodeTool pour prendre en charge OUTREC avec IFTHEN (V7-10124)

- JCL SORT - ajout de la prise en charge du mot clé DATE dans la commande OUTREC (V7-10125)
- JCL - ajout de la prise en charge des procédures In-Stream (V7-10223)

Améliorations

- Un ensemble de données marqué de la disposition « PASS » doit être disponible pour toutes les étapes du travail (V7-9504)
- Support de l'attribut JCL SCHENV (V7-9570)
- Support SEND avec option CTLCHAR (V7-9714)
- COBOL - Gérer différents jeux de caractères séparateurs de lignes dans les instructions ACCEPT (V7-9875)
- Évitez les annulations multiples (V7-9958)
- Autoriser l'utilisation de la disposition MOD à ajouter à la fin des fichiers GDG (V7-10031)
- Optimisation : refactorisation de PutAll (V7-10063)
- PutAll refactorisation : ajout de pagination (V7-10063)
- Rendre le délai de lecture du client Jedis configurable (V7-10063)
- UseSsl support pour le mode autonome (V7-10114)
- Support EIBDS après ouverture réussie du fichier (V7-10147)
- Support EIBDS après une demande de contrôle de fichier (V7-10147)
- Améliorer le support de CICS SYNCPOINT (V7-10187)
- BluesamRedisSerializer: problème avec MetadatapPersistence (V7-10202)
- Support de Redis AWS Secrets Manager pour les files d'attente TS (V7-10204)
- Support JCLBCICS sur la personnalisation de la taille du nom du DD (V7-10224)
- Ajoute la prise en charge du chemin absolu dans l'instruction IDCAMS DELETE (V7-10308)

AS400

Nouvelles fonctionnalités

- Implémentation de la fonction d'aide pour les écrans AS4 00 (V7-9673)

Améliorations

- Nombre d'enregistrements dans l'INFDS (V7-9377)

Capacités transversales

Nouvelles fonctionnalités

- Support pour Runtime on pour EC2 envoyer des journaux à Amazon CloudWatch (D87990246)
- Ajout d'un nouveau point de terminaison pour récupérer les métriques relatives aux lots, aux transactions et à la JVM (D88393832)

Améliorations

- Support des sources de données AWS Secrets Manager pour l'utilitaire pgm (V7-9570)
- Ajout du support DB2 pour DSNUTILB DISCARD (V7-9798)
- Support pour l'écriture dans le logger au lieu du flux de sortie système par défaut dans les fichiers SYSPRINT et SYSPUNCH par défaut (V7-10098)
- Support du cache BluSam Redis et des propriétés de connexion verrouillées dans AWS Secrets Manager (V7-10238)
- Support de la connexion SSL sur Db2 XA AWS secret (V7-10258)
- Metadonnées mises à jour pour IDCAMS REPRO et VERIFY (V7-10281)
- Gestion améliorée du code de retour IDCAMS Abend (V7-10307)

Outils de modernisation, version 3.7.0

Rubriques

- ZoS
- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- PLI - Affectation améliorée pour la section transversale des matrices et les matrices bidimensionnelles (V7-9830)

AS400

Nouvelles fonctionnalités

- Manipulation des indicateurs de niveau de commande (V7-9227)
- Support du paramètre EXTNAME *INPUT (V7-9897)
- Réécriture Goto améliorée : Support pour les balises situées dans les instructions SELECT OTHER (V7-9973)
- Support du mot clé REFSHIT DSPF (V7-10049)

Améliorations

- Amélioration de la gestion du mot clé de description de fichier EXTIND (*INUx) (V7-7404)
- Transformation de fichiers SQLDDS améliorée (V7-7687)
- Les objets de fichier ne sont plus générés pour les fichiers AS4 00 (V7-9062)
- Gestion améliorée du mot-clé de description de fichier EXTDESC (V7-9268)
- Gestion améliorée de la version intégrée de %CHAR (V7-9311)
- Support amélioré pour le défilement de page sur le dernier enregistrement sans SFLEND (V7-9322)
- Support amélioré pour les structures de données préfixées (V7-9436)
- Support pour les dimensions définies avec %SIZE (V7-9472)
- Support pour la gestion du nom de champ PF déclaré entre guillemets (V7-9557)
- Fonctionnement des fichiers amélioré, insensible aux majuscules et minuscules (V7-9785)
- Support pour le champ initialisé à *USER (V7-9806)
- Support pour le type COMP en AS4 00 (V7-9840)
- Analyse COBOL4 00 améliorée sur (Not) InvalidKey (V7-9922)
- Gestion améliorée de l'opération SCAN (V7-9971)
- Support amélioré de l'opcode GOTO (V7-9973)
- Gestion améliorée du fonctionnement EXCEPT (V7-9977)
- Prise en charge améliorée des préfixes (V7-10000)
- Support pour les appels MQ en RPG (V7-10007)

- Intégration %LOOKUP améliorée (structure de données de tableau à clés) (V7-10022)
- Support pour les opérations Close *All (V7-10036)
- Support de l'instruction UPDATE AS ROW CHANGE SQLDDS (V7-10051)
- Amélioration de la gestion de la valeur littérale de type Long (V7-10073)
- Grammaire RPG améliorée (utilisation du mot-clé INZ comme nom du sous-programme) (V7-10074)
- Grammaire RPG améliorée pour prendre en charge les valeurs numériques avec une partie fractionnaire vide (V7-10077)
- Support amélioré pour les champs partagés entre CL et un fichier externe (V7-10081)
- Support amélioré pour les indicateurs conditionnels DDS (V7-10084)
- Support du type binaire DDS avec les programmes COBOL (V7-10100)
- Amélioration de la collision de noms avec le lien (V7-10109)
- Support pour les procédures de mixage principales et d'exportation (V7-10112)
- Support amélioré DataStructure dans une sous-procédure (V7-10113)
- Support amélioré de CLEAR (V7-10126)
- Support amélioré de la boucle DO (V7-10134)
- Support de SQLTYPE dans un RPG entièrement gratuit (V7-10151)
- Analyse améliorée des conditions sur le mot clé DDS (V7-10155)
- Génération DSL améliorée (V7-10163)
- Amélioration de ProcessIndicators lorsque la condition est une expression binaire. (V7-10164)
- Amélioré GOTOs avec la condition Else (V7-10168)
- Support pour le type Heure et horodatage dans le DSPF (V7-10173)
- Analyse améliorée de la ligne de continuation pour DDS (V7-10183)
- Support COBOL pour RENAMES FLD OF RECORD (V7-10195)
- Analyse des indicateurs conditionnels améliorée sur les champs DSPF (V7-10221)
- Support de l'analyse syntaxique du mot-clé DDS NOALTSEQ (V7-10288)
- Support : menu d'aide et champs masqués (V7-10314)
- Vérification améliorée de l'intégrité des mots clés d'aide DSPF (V7-10328)
- Ne plus propager tous les mots clés dans le champ Ref (V7-10347)

Capacités transversales

Nouvelles fonctionnalités

- Data Migrator - Gestion des données CLOB (V7-9665)

Améliorations

- Propagation de la propriété JCL SCHENV depuis la définition JOB vers PROC GROOVY via (V7-10225) JobContext
- FrontEnd - Réglage de la taille de la fenêtre en cas d'absence de bordure (V7-10358)

Notes de mise à jour 3.6.0

Cette version de AWS Blu Age Runtime and Modernization Tools fournit de nouvelles fonctionnalités pour les migrations existantes vers zOS et AS4 00, principalement destinées à étendre les mécanismes de support CICS, à compléter les fonctionnalités JCL, à optimiser les performances des fonctionnalités simultanées et à volume élevé, et à ajouter des fonctionnalités. multi-data-source Certains des principaux changements apportés à cette version sont les suivants :

- Amélioration de la gestion dynamique des fichiers JCL, extension des instructions actuelles et gestion des ensembles de données concaténés, exécution de plusieurs instructions dans un seul bloc et transfert de données des lots vers les programmes.
- Prise en charge améliorée de plusieurs commandes CICS, y compris la recherche de plusieurs types de ressources CICS.
- La possibilité de disposer de différentes bases de données lors de l'utilisation de Blu Age Runtime Utilities, idéale pour les scénarios dans lesquels les données commerciales sont distribuées entre plusieurs sources.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.6.0

Rubriques

- ZoS
- AS400

- Capacités transversales

ZoS

Nouvelles fonctionnalités

- JCL - DynamicFileBuilder - Gestion améliorée des descripteurs de fichiers (V7-9408)
- Conversion de format améliorée sur certaines DB2 fonctions SQL intégrées lors de l'appel de l'utilitaire INFUTILB UNLOAD (V7-9554)
- Affectations de réseaux multidimensionnels PLI améliorées (V7-9592)
- Gestion de la redirection sysout vers un fichier (V7-9992)

Améliorations

- Ajout du déclenchement de procédures stockées pour les DB2 SGBDR (V7-9155)
- SORT gère la conversion au format PDF (V7-9286)
- JCL/GROOVY - Amélioration de l'instruction REPRO pour prendre en charge les ensembles de données DUMMY (V7-9424)
- Améliorer le support CICS UNLOCK (V7-9606)
- Gérer la taille de valeur par défaut pour Union (V7-9648)
- JCL/GROOVY handle different termination/dispositiondans des ensembles de données concaténés (V7-9653)
- Rendre PageSize configurable pour les ensembles de données Blusam (V7-9680)
- DSNUTIL - autorise le chargement de 24:00:00 comme heure valide dans LUW (V7-9697) DB2
- Support de comparaison de valeurs élevées (0xff) dans NumberUtils .ne ()/ NumberUtils.eq () (V7-9731)
- JCL/GROOVY - supporte DO... Mots clés THEN dans les IF-THEN-ELSE clauses IDCAMS pour exécuter plusieurs instructions dans un seul bloc (V7-9750)
- JHDB non valide a appelé un programme en dehors de JHDBBatch Runner (V7-9782)
- Support des espaces blancs dans la carte de contrôle SORT OUTFIL (V7-9808)
- Améliorer le support CICS READ PREV (V7-9845)
- Améliorer l'accès simultané aux index des ensembles de données (V7-9864)
- Améliorer la prise en charge de CICS REWRITE (V7-9873)

- COBOL : prise en charge des instructions SYSIN multiligne dans les instructions ACCEPT pour transmettre des données d'un lot (JCL) à un programme (COBOL) (V7-9875)
- Groovy - Meilleure gestion de l'étape de ConcatenatedFileConfiguration création des fichiers (V7-9876)
- UTILITAIRE IDCAMS - Gestion de l'instruction DEFINE PATH (V7-9878)
- SORT BUILD - Ajustez l'option TRAN et gérez les blancs implicites (V7-9925)
- Améliorez CICS DELETE avec le support des options GENERIC (V7-9939)
- Améliorez le support CICS STARTBR et ENDBR (V7-9952)
- Améliorez les performances de clôture en cas d'accès simultané (V7-9953)
- Améliorer la gestion de l'état des fichiers au démarrage (V7-9991)
- Groovy - Autorise l'appel de GetDisposition ()/()/getNormalTermination() sur getAbnormalTermination ConcatenatedFileConfiguration (V7-10012)

AS400

Nouvelles fonctionnalités

- Support des indicateurs externes sur les mots clés COMMIT (V7-6035)
- Réinitialisez la boucle ReadC après l'écriture SFLCTL (V7-8061)
- Support de l'indicateur LR dans CALL (V7-9250)
- Ajout d'un nouveau type de champ dynamique (fractionné) pour gérer le champ de saisie sur plusieurs lignes (V7-9370)
- Support du fichier primaire/secondaire (V7-9390)
- Les zones de données locales sont désormais transmises à la tâche appelée lors de la soumission d'une tâche (V7-9775)
- Support de QTEMP pour la zone de données et prise en charge de la création de valeur de la zone de données. (V7-9916)
- Contrôle des engagements : support pour activer/désactiver le contrôle des engagements (V7-9956)
- Support des indicateurs externes sur les mots clés COMMIT

Améliorations

- Amélioration de l'affichage de la valeur 0 et de l'EDTWRD (V7-8933)

- Support du mot-clé DSPF « CHKMSGID » (V7-9125)
- Transaction de validation SQL à la fin du lot (V7-9232)
- Améliorer la prise en charge des mots clés EXPORT et IMPORT pour le champ et la structure des données (V7-9265)
- Support en minuscules DateHelper (V7-9461)
- Support de conversion *CYMD en *ISO (numérique) (V7-9488)
- Améliorer la gestion du %len intégré pour un champ variable (côté gauche et côté droit d'une expression) (V7-9733)
- Amélioration de la prise en charge des fonctions intégrées '%LOOKUPXX' XX (« LE », "LT », "GE », "GT ») (V7-10064)

Capacités transversales

Nouvelles fonctionnalités

- CICS - Améliore la transaction de demande de renseignements pour connaître le statut des options (V7-9712)
- JCL - Améliore le chargement pour Sysprint avec le fichier de sortie système (V7-9797)
- CICS - Améliore INQUIRE TSQUEUE (V7-9823)
- CICS - Amélioration du terminal de demande pour l'option userid (V7-9906)

Améliorations

- Améliorez le maniement de la comparaison avec le blanc (V7-8047)
- Améliorer la journalisation pour Jics et Blusam (V7-8847)
- Support des attributs étendus SOSI du BMS et du symbole programmé F8 pour les champs dynamiques (V7-8857)
- Gérer le dépassement de la mémoire tampon dans les paramètres du programme (V7-9138)
- Améliorer la simultanéité des threads et des écritures pour le registre des verrous Blusam (V7-9505)
- Support de configuration de sources de données multiples pour Utility-PGM (V7-9570)
- Mode de verrouillage du niveau d'enregistrement Blusam uniquement (V7-9626)
- Assurez-vous que la persistance des métadonnées résiste au redémarrage du serveur (V7-9748)

- Améliorer le nettoyage du DAO en cas d'exception (fermeture du navigateur) (V7-9790)
- Support DummyFile pour INFUTILB SYSPUNCH (V7-9799)
- Améliorer la prise en charge des valeurs négatives sur NumericEditedType (V7-9935)

Outils de modernisation, version 3.6.0

Rubriques

- ZoS
- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- JCL - Amélioration de la journalisation pour la fin de la procédure (V7-8509)
- PL1 - Amélioration de la génération de sacs pour le type de données PakedLong (V7-8917)
- JCL - Améliore la journalisation pour la fin de la procédure lorsque le fichier contient le marqueur « fin »//(V7-9509)
- PL1 - Améliorez la prise en charge de GET EDIT avec Fixed-point et SYSIN stream (V7-9593)
- DB2 - Amélioration de la prise en charge du DB2 type VARGRAPHIC (V7-9809)
- CICS - Améliore la sécurité des requêtes de commande pour l'option LOGMESSAGE (V7-9969)
- PL1 - Améliorez la génération de sacs pour l'intégration de CHARG/Chargraphic (V7-9989)

Améliorations

- PL1- Amélioration de la prise en charge du mot clé INCLUDEX (V7-9588)
- PL/I- Gère le mot clé CHARGRAPHIC comme paramètre valide de tout appel de méthode (V7-9589)
- Amélioration de la résolution des variables PL1 hôtes lorsqu'elles sont nommées avec des caractères spécifiques @ # $ §. (V7-9654)
- COBOL - Support des mots clés C01... C12 et S01... S05 en tant que paramètre de l'instruction WRITE ADVANCING à l'étape d'analyse (V7-9669)

AS400

Nouvelles fonctionnalités

- Support de la transformation SQL-DDS dans Analyzer (V7-7687)
- Automatiser la détection des fichiers SQL-DDS (V7-7687)
- Implémentation du prétraitement SQL-DDS (V7-7687)
- Support du mot clé ALIGN (V7-9254)
- Support du ExtName DSPF et de la matrice multi-DIM (V7-9663)
- InvalidKey Déclarations de support sur COBOL WRITE (V7-9793)

Améliorations

- Amélioration de l'opcode TESTB (V7-8865)
- Améliorer le support du DECFMT on focus (V7-8933)
- Gestion de l'indicateur résultant sur MOVE (V7-9224)
- Améliorer la prise en charge du modèle de mot clé pour le champ et la structure de données (V7-9278)
- Amélioration de LIKEDS (le DS défini à l'aide de LIKEDS est automatiquement qualifié) (V7-9302)
- COBOL - Améliorer la génération de la structure des indicateurs (V7-9423)
- Le paramètre Const du prototype n'est pas en lecture seule (V7-9437)
- Améliorez le mot clé EDTCDE avec le code d'édition « Y » (V7-9443)
- Support de génération du champ *ROUTINE dans PSDS et INFDS (V7-9487)
- Améliorer le champ de réécriture XXX pour le rendre autonome (la valeur par défaut est perdue lors de la réécriture) (V7-9522)
- Support amélioré des mots clés DSPF (V7-9658)
- Gestion de la valeur par défaut ZEROES sur le binaire (V7-9666)
- Support du pointeur implicite (V7-9719)
- Améliorez la gestion de l'appel intégré %size avec un seul paramètre (V7-9730)
- Améliorez le traitement des références de structure de données dans les appels intégrés (%ELEM) (V7-9736)
- Améliorer la gestion de la longueur signée pour le champ avec une référence LIKE dans la spécification de définition (V7-9738)

- Amélioration de REWRITE (V7-9791)
- Amélioration de la génération d'index à partir de fichiers DDS (V7-9803)
- Améliorez la robustesse des mappeurs avec une valeur numérique non valide (V7-9813)
- Amélioration SQLModel et génération de fichiers AllIndexes (V7-9818)
- Améliorez le support DS qualifié (V7-9863)
- Amélioration de la prise en charge de LOOKUP (avec un champ autonome COMME un DS en paramètre) (V7-9961)
- Améliorer le LIKE sur l'indicateur (V7-9985)
- Gestion de l'indicateur résultant sur le MVR (V7-9995)
- Support du caractère N avec tilde (V7-10021)
- Améliorez la génération de fichiers DDL modernes à partir des anciens fichiers SQLDDS (V7-10067)

Capacités transversales

Nouvelles fonctionnalités

- Personnaliser l'emplacement des ressources avec une propriété yml (D88816105)
- COBOL - Support de l'instruction EXIT PERFORM pour quitter un PERFORM en ligne sans utiliser de GO TO/PERFORM... VIA (V7-9582)
- Spécifier le codage existant par défaut à prendre en compte dans les métadonnées globales. (V7-9883)

Améliorations

- Améliorer la génération de masques (V7-9602)
- Améliorer l'échauffement du contexte (V7-9621)
- Sécurisez le thread Charset CUSTOM93 0. (V7-9674)
- Amélioration apportée à MOVEA (V7-9773)

Notes de mise à jour 3.5.0

Cette version de AWS Blu Age Runtime and Modernization Tools fournit de nouvelles fonctionnalités pour les anciennes migrations vers zOS et AS4 00, principalement orientées vers l'optimisation des

ensembles de données et de la messagerie, ainsi que des fonctionnalités Java étendues en tant qu'atout du processus de transformation. Certains des principaux changements apportés à cette version sont les suivants :

- Possibilité de migrer des programmes CL vers Java en plus de la fonctionnalité préexistante de scripts groovy, afin de faciliter son intégration avec d'autres programmes modernisés et de simplifier la courbe d'apprentissage des clients en unifiant le langage de programmation qui en résulte.
- Réduction du temps et optimisation des performances des chargements de jeux de données dans Redis grâce à la nouvelle fonctionnalité de masse de données.
- Capacité à exploiter et à transmettre des ensembles de données au cours des étapes de travail afin de moderniser les comportements traditionnels des ensembles de données.
- Extension de la migration SQL pour prendre en charge les fichiers d'entrée VB et migration simplifiée avec Java 11.
- Plusieurs nouveaux mécanismes pour une intégration plus rapide avec IBM MQ, notamment des en-têtes supplémentaires, un support GET/PUT étendu et la récupération automatique des métadonnées des files d'attente.
- Point de terminaison REST pour les métadonnées des ensembles de données et l'importation de jeux de données à partir de compartiments S3.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.5.0

Rubriques

- ZoS
- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- JCL SORT - Gérer la nouvelle superposition de mots clés (V7-9409)
- ZOS COBOL - amélioration de la prise en charge des caractères flottants (V7-9404)

- Port de RedisJics TSQueue destination RedisTemplate et ListOperations (V7-9212)
- ZOS JCL - améliore le chemin du répertoire temporaire avec le répertoire des fichiers s'il est défini via UserDefinedParameters (V7-9012)
- Manipulez la FONCTION ORD-MAX avec TOUS (tous les éléments de la gamme) (V7-9366)
- Des clés préfixées et lisibles par l'homme sont désormais utilisées lors du stockage des files d'attente TS dans Redis (V7-9212)
- Ajouter un point de terminaison get dataset pour l'API Blusam
- JCL - AJOUT du support pour les tâches par lots dont le nom comporte un caractère spécial tel que # (V7-9136)
- TSModel la récupération est désormais effectuée de manière robuste à la demande (V7-9212)

Améliorations

- Support INCLUDE non versionné dans les fichiers LNK (V7-6022)
- MQ - Support d'encodage amélioré (V7-9652)
- Amélioration de la prise en charge des octets doubles ou des jeux de caractères mixtes pour différents types de caractères (V7-9596)
- JCL - Support de la configuration du répertoire de fichiers dans IDCAMS (suppression des instructions NONVSAM) (V7-9609)
- Support du mode masse pour le chargement d'ensembles de données ESDS et RDS à partir de fichiers (V7-8639)
- Gérez l'ouverture d'un ESDS vide en mode entrée. (V7-9287)
- Améliorez l'instruction DEFINE CLUSTER avec la prise en charge des abréviations ORD/UNORD (V7-9451)
- Améliorations des performances du verrou Blusam Redis (V7-8639)
- Améliorez l'instruction DEFINE CLUSTER pour prendre en charge RECORDSIZE fourni dans la portée de l'argument DATA () (V7-9337)
- Ajoute le support des attributs BUFFERSPACE/UNIQUE sur les instructions DEFINE CLUSTER (V7-9419)
- Améliorez l'opération de lecture de Blusam pour les ensembles de données d'enregistrements de longueur variable. (V7-9391)
- L'ADRESSE CICS représente correctement le CWA manquant comme nul (V7-9491)
- Supprimer les écritures inutiles aux extrémités des verrous (V7-8639)

- Gérer l'injection de modèles de cache Redis dans le cache (V7-9510)
- Décode correctement le paramètre BPXWDYN (V7-9417)
- Amélioration de la consommation d'exportations LISTCAT (V7-9201)
- Prise en charge des caractères non imprimables dans le nom des files d'attente Blusam TS (V7-9212)
- Gérer la création de cartes de réception pour le champ avec mapset null (V7-9486)
- Améliorez BluesamRelativeFile les opérations de suppression et de réécriture pour le mode d'accès dynamique. (V7-8989)

AS400

Nouvelles fonctionnalités

- Ajout d'une fonctionnalité permettant de générer des fichiers CL sous forme de programmes Java via le pivot DS/STM standard (V7-9427)
- Support du fichier d'entrée avec le mode ADD (V7-9378)
- Amélioration de la gestion de l'ordre de tri et de la récupération pour prendre en charge la commande cl OPNQRYF (Open Query File) et ajout de la prise en charge du paramètre SHARE dans. OverrideItem (V7-9364)

Améliorations

- Support SFLNXTCHG sur (V7-8061) UpdateSubfile
- Modifier la portée du contexte CL lors de l'exécution de la commande CL (V7-9624)
- Gérer le code de retour pour le programme BPXWDYN (V7-9417)
- Effacez les moniteurs locaux. (V7-9624)
- Support du mot clé DSPF RTNCSRLOC (V7-9389)
- setOnGreaterOrEqual() ne définit pas la valeur égale à 1 (V7-9342)
- Mettre à jour le cache des champs sur UpdateSubfileRecord (V7-9376)
- Améliorer le support SFLNXTCHG (V7-8061)

Capacités transversales

Nouvelles fonctionnalités

- Ignorez le préfixe G sur la chaîne graphique littérale. (V7-9420)
- ZOS COBOL - Amélioration de la prise en charge de Fiedl.initialize () pour certaines structures spéciales (V7-9485)
- Autoriser l'initialisation du contexte de manière asynchrone pour améliorer les performances de démarrage du programme (V7-9446)
- SQL Release explicitement l'instruction prepare ouverte et ResulSet. (V7-9422)
- Amélioration de JMS MQ - prise en charge de MQ PUT/V7-7085 - MQRFH2 prise en charge du gestionnaire de files d'attente par défaut (V7-9400)
- SQL Management - Activer les conversions Lambda sur les paramètres des commandes SET (V7-9492)
- ZOS MQ JMS - Ajout du support à MQCOMIT et MQBACK (V7-9399)
- ZOS IBMMQ - Amélioration de la prise en charge de MQINQ (V7-9544)
- Gérez l'opération CONCAT avec un octet au lieu d'une chaîne lors de l'utilisation d'un codage à double octet. (V7-8932)
- ZOS IBMMQ - Amélioration de la prise en charge de la commande PUT avec les options SET_ALL_CONTEXT (V7-9544)

Améliorations

- Gérer les noms de fichiers gdg avec le caractère $ (V7-9066)
- SQL Diagnostic renvoie 1 sous forme de clause NUMBER lorsque l'instruction SQL précédente est réussie. (V7-9410)
- Schéma d'un champ dont la longueur n'est pas nulle (V7-7536)
- Support de la fonction PL1 GRAPHIQUE intégrée (V7-9245)
- MQ - Ajout du support de version pour le réglage des champs MQGMO (V7-9500)
- JMS MQ GET - Amélioration de la longueur des données renvoyées par le message (V7-9502)
- Définissez sqlerrd (3) avec le nombre d'éléments récupérés dans le contexte ROWSET. (V7-9371)

Outils de modernisation, version 3.5.0

Rubriques

- ZoS

- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- ZOS PLI - Support de l'index astérisque lors de l'assignation avec une expression binaire (V7-9178)
- JCL to BatchScript - Un «//» marque la fin de l'exécution de la tâche (V7-9304)
- ZOS PLI - amélioration de la prise en charge des caractères flottants et des signatures en type numérique édité (V7-8982)
- COBOL - Support de la fonction SUM intégrée (V7-9367)
- JCL- éventuellement, commentez le code mort après une instruction nulle (//) (V7-9202)
- JCL- Support de l'opérateur « | » dans la déclaration de condition (V7-9499)
- PL/I- Commentaire des directives de précompilation lors de l'étape de prétraitement pour empêcher les exceptions d'analyse syntaxique (V7-9507)

Améliorations

- Gérer la définition du flux avec un délimiteur (V7-9615)
- Amélioration de la gestion des exportations LISTCAT. (V7-9201)
- PL/I- Amélioration de la prise en charge des arguments « nuls » implicites (V7-9204)

AS400

Nouvelles fonctionnalités

- Support du mot clé DDS CONCAT (V7-9439)
- Refactorisez le code Java généré pour les mots clés DSPF. (V7-7700)
- Support : mots clés variables dans les champs d'une définition de structure de données (V7-9029)

Améliorations

- Améliorer l'analyse de la relation logique ET/OU (V7-9352)

- COBOL Améliore le mappage entre vo et DSentity (V7-9449)
- Afficher une valeur vide si la saisie numérique est focalisée (V7-9374)
- Variable locale dans le curseur de déclaration SQL (V7-9456)
- Problème de portée avec un DS vide (V7-9466)
- Tronquer les lignes après col 80 avant l'analyse (V7-9632)
- Améliorez la gestion des références de champ et des appels intégrés dans les mots clés (DIM, LIKE,...) dans la spécification de définition (V7-9358)
- Support des commentaires SQL (--) (V7-9632)
- FullFree analyse syntaxique, tapez Date/Time/Timestamp (V7-9542)
- Inclure le SQLCA lors de l' FullFree analyse syntaxique (V7-9333)
- Support amélioré du niveau de contrôle. (V7-9610)
- Comparaison de Handle DS avec *BLANKS (V7-9668)
- Améliorer la prise en charge de plusieurs indicateurs dans le DDS (V7-9318)
- Améliorer la prise en charge de plusieurs programmes DSPF (V7-9657)
- Améliorez la gestion des champs avec LIKE (cas d'une structure de données aimée et cas d'une structure de données aimée dans un tableau) (V7-9213)
- RPG gratuit, continuez à gérer au pied de la lettre (V7-9686)
- Support amélioré des dossiers de fin de programme (V7-9452)
- Support de la phrase LINKAGE dans l'instruction CALL. (V7-9685)
- Code d'opération CASXX (CASBB sans groupe CASXX) (V7-9357)
- Améliorer l'analyse FullFree des RPG (V7-9457)
- Le %LEN intégré ne prend pas en charge DS comme argument (V7-9267)
- Améliorations de MOVEA lorsque le facteur 2 est *ALL'X... ' (V7-9228)
- Support : attribution avec champ RENAME (V7-9385)

Capacités transversales

Nouvelles fonctionnalités

- Outil SQL Migrator : ajoutez une option OID pour une longueur d'enregistrement variable lors de l'étape de chargement d'ebcdic. (V7-9380)

- Outil SQL Migrator - Support de Java 11 sur l'option OID (V7-9599)

Améliorations

- Amélioration de la prise en charge des baies imbriquées (V7-9595)
- Remplacez Â¬ caractère par ! dans le cas de Â¬ est supporté par le codage original. (V7-9465)
- JCL - Support de la terminaison normale du PASS pour partager des ensembles de données entre les étapes de travail (V7-9504)
- Appliquez ON NULL à la définition de colonne sur ORACLE lorsqu'il s'agit de VARCHAR et de type de colonne de base de données nullable. (V7-9681)
- Améliorez la conformité des injections à ressort (V7-9635)

AWS Failles de sécurité dans Blu Age

Common Vulnerabilities and Exposures (CVE) est une liste de référence pour les vulnérabilités de cybersécurité connues du public. Chaque entrée contient un numéro d'identification, une description et au moins une référence publique.

Nous vous recommandons de toujours passer à la dernière version de AWS Blu Age afin de vous protéger contre les vulnérabilités connues. Les analyses de sécurité sont effectuées en permanence avec Amazon Inspector et les résultats sont classés en fonction de leur gravité dans le NIST.

La liste suivante détaille les CVEs correctifs apportés dans chaque version mineure disponible, qui résultent de l'utilisation de dépendances :

Version	CVE
4.8.0	CVE-2025-22228
4.7.0	CVE-2025-25193, CVE-2025-24970, CVE-2024-47535, CVE-2024-38820, CVE-2024-38809, CVE-2024-22262, CVE-2_45868
4.6.0	CVE-2024-12801, CVE-2024-12798, CVE-2024-50379, CVE-2024-56337

Version	CVE
4.5.0	CVE-2024-47535, CVE-2024-52316, CVE-2024-47535, CVE-2024-38827
4.4.0	CVE-2024-38820, CVE-2024-38821, CVE-2024-38809, CVE-2024-38816, CVE-2024-47554, CVE-2024-6484, CVE-2024-6485
4.3.0	CVE-2024-43788, CVE-2_25898, CVE-2021-30246, CVE-2024-21484, CVE-2024-34750
4.2.0	CVE-2020-11023, CVE-2023-26364, CVE-2019-11358, CVE-2020-11022, CVE-2021-23358, CVE-2017-18214, CVE-2_24785, CVE-2_31129, CVE-2023-48631
4.1.0	CVE-2024-29025, CVE-2024-23080, CVE-2024-22262, CVE-2024-30171, CVE-2024-29857, CVE-2024-30172
4.0.0	CVE-2016-1000027, CVE-2_1471, CVE-2024-1597, CVE-2024-22243, CVE-2024-22233, CVE-2024-22234, CVE-2024-22259, CVE-2024-22257, CVE-2024-29131, CVE-2024-29133

> **Note**
>
> Pour plus d'informations sur CVEs les correctifs apportés dans les versions précédentes, veuillez contacter votre responsable de livraison AWS Blu Age

Instructions de mise à niveau pour AWS Blu Age

Cette page contient des instructions pour la mise à niveau de la version AWS Blu Age.

Mises à niveau courantes

Dans la plupart des cas, lors de la mise à niveau de la version (non gérée) de AWS Blu Age RuntimeWARs, vous devez remplacer les artefacts (fichiers de configuration, scripts, etc.) de votre version précédente par ceux fournis dans la nouvelle et redémarrer votre application. Assurez-vous d'effectuer des tests de régression approfondis sur vos applications modernisées une fois la mise à niveau effectuée. Vous pouvez également contacter votre responsable de livraison AWS Blu Age pour obtenir des instructions spécifiques applicables à votre application.

Pour mettre à niveau la version (gérée) de AWS Blu Age Runtime, voirEnvironnements d'exécution gérés.

Certaines mises à niveau peuvent nécessiter une configuration supplémentaire pour garantir la compatibilité. Dans ce cas, suivez les instructions relatives à cette mise à niveau spécifique.

Migration de la version 3.10.0 vers la version 4.0.0

La principale modification apportée à la version 4.0.0 est la migration de Spring Boot 2.7 vers Spring Boot 3.2 et de Tomcat 9 vers Tomcat 10.

Changements de code

Cette section répertorie les modifications nécessaires pour rendre le code modernisé compatible avec AWS Blu Age Runtime 4.0.0. Vous pouvez ignorer cette section si vous décidez de lancer une nouvelle génération en utilisant la version 4.0.0 sur Blu Insights (Transformation Center).

Modifications du POM

Groupe	ArtifactId	Modification
org.slf4j	slf4j-api	Supprimer (est une dépendance transitive)
org.yaml	snakeyaml	Supprimer (est une dépendance transitive)
org.springframework.boot	spring-boot-starter-web	- Mettre à jour spring.bo ot.version vers la version 3.2.4 - Supprimer l'exclusion de log4 4j j-to-slf

Groupe	ArtifactId	Modification
org.springframework.boot	spring-boot-starter-jta-atomiques	Passez à com.atomikos : 3-starter:6.0.0 transactions-spring-boot
org.apache.commons	commons-dbcp2	Mise à niveau vers la version 2.10.0
org.postgresql	postgreql	Mise à niveau vers la version 42.7.2
com.microsoft.sqlserver	mssql-jdbc	Mise à niveau vers 12.4.2.jre11
com.oracle.database.jdbc	ojdbc8	Passage à la version 23.3.0.23.09 d'ojdbc11

Migrer de Javax vers Jakarta

La mise à niveau de Tomcat s'accompagne d'une migration du package Java Javax vers Jakarta. Assurez-vous de mettre à jour vos importations en conséquence de javax.* vers jakarta. *.

Presque toutes les anciennes classes référencées du package Javax se trouvent à Jakarta. Les exceptions connues à cette règle sont les `javax.xml` packages `javax.sql` et, qui sont toujours inchangés.

Atomikos change

En raison de la modification de dépendance mentionnée ci-dessus, les références à `org.springframework.boot.jta.atomikos.AtomikosDataSourceBean` doivent être remplacées par`com.atomikos.spring.AtomikosDataSourceBean`.

Suppression du dialecte PostgreSQL

La classe personnalisée `PostgreSQLDialect.java` est supprimée. Les références à celui-ci dans le lanceur principal doivent également être supprimées.

Déploiement (AWS Blu Age Runtime (non géré))

Tomcat

Cette version est compatible avec Tomcat`10.1.17`. La mise à niveau du serveur Tomcat vers cette version est nécessaire pour exécuter le Blu Age Runtime`4.0.0`. Assurez-vous de transférer les anciennes modifications de configuration (notamment les propriétés Catalina).

Dépendances partagées

Le dossier partagé d'exécution contient les up-to-date dépendances.

Dépendances supplémentaires

Si vous avez utilisé des dépendances supplémentaires (non incluses dans le runtime), vous devrez peut-être les mettre à jour. Le fichier readme situé dans le dossier supplémentaire répertorie les versions prises en charge.

AWS Cycle de vie de Blu Age

Cette section définit les dates de fin de vie (EOL) des versions majeures de AWS Blu Age Runtime. Cela vous permet de planifier les mises à niveau des versions afin de rester à jour avec les dernières fonctionnalités et maintenances. Pour mettre à jour votre version, consultezthe section called "Mise à niveau de AWS Blue Age".

Nous vous recommandons de vérifier les nouvelles versions tous les 3 mois et de passer fréquemment aux versions récentes. Pour chaque mise à niveau, vous devez effectuer des tests non de régression sur vos applications modernisées avant les déploiements critiques ou de production.

> **Note**
>
> Les dates de fin de vie peuvent être modifiées en raison de mesures de sécurité critiques. Pour en savoir plus, consultez Cycle de vie des composants.

AWS Fin de vie (EOL) de Blu Age Runtime

Le tableau suivant récapitule la date de fin de vie de chaque version majeure.

Version majeure	Date de fin de vie
Version 3	8 juillet 2024
Version 4	Pas encore publié

> **Note**
>
> La date d'expiration de la version majeure 4 sera alignée sur la disponibilité de la prochaine version majeure.

Pour comprendre le modèle de support des versions mineures, reportez-vous àCycle de vie des composants.

AWS Concepts de Blu Age Runtime

Comprendre les concepts de base du AWS Blu Age Runtime peut vous aider à comprendre comment vos applications sont modernisées grâce au refactoring automatique.

Rubriques

- AWS Architecture de haut niveau de Blu Age Runtime
- AWS Structure Blu Age d'une application modernisée
- Que sont les simplificateurs de données dans AWS Blu Age
- AWS Fard à joues Blue Age
- Programmes disponibles dans l'application Web utilitaire
- AWS Console d'administration Blu Age Blusam

AWS Architecture de haut niveau de Blu Age Runtime

Faisant partie de la solution AWS Blu Age pour la modernisation des programmes existants vers Java, le AWS Blu Age Runtime fournit un point d'entrée unifié basé sur REST pour les applications modernisées, ainsi qu'un cadre d'exécution pour ces applications, grâce à des bibliothèques fournissant des structures héritées et à une standardisation de l'organisation du code des programmes.

Ces applications modernisées sont le résultat du processus AWS Blu Age Automated Refactor visant à moderniser les programmes centraux et de milieu de gamme (appelés « anciens » dans le document suivant) vers une architecture basée sur le Web.

Les objectifs de AWS Blu Age Runtime sont la reproduction du comportement des programmes existants (isofonctionnalité), les performances (en termes de temps d'exécution des programmes et de consommation de ressources) et la facilité de maintenance des programmes modernisés par les

développeurs Java, grâce à l'utilisation d'environnements et d'expressions familiers tels que tomcat, Spring, getters/setters, fluent. APIs

Rubriques

- AWS Composants d'exécution Blu Age
- Environnements d'exécution
- Apatridie et gestion des sessions
- Haute disponibilité et apatridie

AWS Composants d'exécution Blu Age

L'environnement d'exécution AWS Blu Age est composé de deux types de composants :

- Ensemble de bibliothèques Java (fichiers JAR) souvent référencées sous le nom de « dossier partagé » et fournissant des constructions et des instructions héritées.
- Ensemble d'applications Web (fichiers de guerre) contenant des applications Web basées sur Spring fournissant un ensemble commun de cadres et de services aux programmes modernisés.

Les sections suivantes décrivent en détail le rôle de ces deux composants.

AWS Bibliothèques Blu Age

Les bibliothèques AWS Blu Age sont un ensemble de fichiers jar stockés dans un `shared/` sous-dossier ajouté au chemin de classe Tomcat standard, afin de les rendre disponibles pour tous les programmes Java modernisés. Leur objectif est de fournir des fonctionnalités qui ne sont ni nativement ni facilement disponibles dans l'environnement de programmation Java, mais typiques des environnements de développement existants. Ces fonctionnalités sont exposées d'une manière aussi familière que possible aux développeurs Java (getters/setters, basés sur les classes, fluents). APIs Un exemple important est la bibliothèque Data Simplifier, qui fournit aux programmes Java des structures de configuration et de manipulation de mémoire héritées (utilisées dans les langages COBOL PL1 ou RPG). Ces fichiers JAR sont une dépendance essentielle du code Java modernisé généré à partir de programmes existants. Pour plus d'informations sur le simplificateur de données, consultezQue sont les simplificateurs de données dans AWS Blu Age.

Application web

Les archives d'applications Web (WARs) constituent un moyen standard de déployer du code et des applications sur le serveur d'applications Tomcat. Ceux fournis dans le cadre du runtime AWS

Blu Age visent à fournir un ensemble de frameworks d'exécution reproduisant les environnements existants et les moniteurs de transactions (lots JCL, CICS, IMS...), ainsi que les services requis associés.

Le plus important est `gapwalk-application` (souvent abrégé en « Gapwalk »), qui fournit un ensemble unifié de points d'entrée basés sur REST pour déclencher et contrôler l'exécution des transactions, des programmes et des lots. Pour de plus amples informations, veuillez consulter AWS Blue Age Runtime APIs.

Cette application Web alloue des fils d'exécution Java et des ressources pour exécuter des programmes modernisés dans le contexte pour lequel ils ont été conçus. Des exemples de tels environnements reproduits sont détaillés dans la section suivante.

D'autres applications Web ajoutent à l'environnement d'exécution (plus précisément, au « Registre des programmes » décrit ci-dessous) des programmes émulant ceux disponibles et appelables depuis les anciens programmes. Deux catégories importantes de ce type sont les suivantes :

- Émulation de programmes fournis par le système d'exploitation : les lots pilotés par JCL s'attendent notamment à pouvoir appeler une variété de programmes de manipulation de fichiers et de bases de données dans le cadre de leur environnement standard. Les exemples incluent`SORT`/`DFSORT`ou`IDCAMS`. À cette fin, des programmes Java reproduisant un tel comportement sont fournis et peuvent être appelés en utilisant les mêmes conventions que les anciens programmes.
- Les « pilotes », qui sont des programmes spécialisés fournis par le framework d'exécution ou le middleware comme points d'entrée. Par exemple`CBLTDLI`, les programmes COBOL exécutés dans l'environnement IMS dépendent pour accéder aux services liés à IMS (base de données IMS, dialogue utilisateur via MFS, etc.).

Registre des programmes

Pour participer à ces constructions, frameworks et services et en tirer parti, les programmes Java modernisés à partir des anciens programmes adhèrent à une structure spécifique documentée dans. AWS Structure Blu Age d'une application modernisée Au démarrage, le AWS Blu Age Runtime collectera tous ces programmes dans un « registre des programmes » commun afin qu'ils puissent être invoqués (et s'appeler mutuellement) par la suite. Le registre des programmes fournit un couplage souple et des possibilités de décomposition (étant donné que les programmes qui s'appellent entre eux n'ont pas besoin d'être modernisés simultanément).

Environnements d'exécution

Les environnements et chorégraphies traditionnels fréquemment rencontrés sont disponibles :

- Les lots pilotés par JCL, une fois modernisés en programmes Java et en scripts Groovy, peuvent être démarrés de manière synchrone (blocage) ou asynchrone (détachée). Dans ce dernier cas, leur exécution peut être surveillée via des points de terminaison REST.
- Un sous-système AWS Blu Age fournit un environnement d'exécution similaire à CICS grâce à :
  - un point d'entrée utilisé pour démarrer une transaction CICS et exécuter les programmes associés tout en respectant la chorégraphie des « niveaux de course » du CICS,
  - un stockage externe pour les définitions de ressources,
  - un ensemble homogène d'`EXEC CICS`instructions se APIs reproduisant couramment en Java,
  - un ensemble de classes enfichables reproduisant les services CICS, tels que les files d'attente de stockage temporaires, les files de données temporaires ou l'accès aux fichiers (plusieurs implémentations sont généralement disponibles, telles qu'Amazon Managed Service pour Apache Flink, Amazon Simple Queue Service ou RabbitMQ pour les files d'attente TD),
  - pour les applications destinées aux utilisateurs, le format de description d'écran BMS est modernisé pour devenir une application Web angulaire, et le dialogue « pseudo-conversationnel » correspondant est pris en charge.
- De même, un autre sous-système fournit une chorégraphie basée sur des messages IMS et prend en charge la modernisation des écrans d'interface utilisateur au format MFS.
- En outre, un troisième sous-système permet l'exécution de programmes dans un environnement de type iSeries, y compris la modernisation des écrans spécifiés au format DSPF (Display File).

Tous ces environnements s'appuient sur des services communs au niveau du système d'exploitation tels que :

- l'émulation de l'allocation et de la disposition de la mémoire héritées (Data Simplifier),
- reproduction basée sur un thread Java du mécanisme d'exécution des « unités d'exécution » COBOL et de transmission de paramètres (`CALL`instruction),
- émulation d'organisations plates, concaténées, VSAM (via le jeu de bibliothèques Blusam) et GDG Data Set,
- accès aux magasins de données, tels que les RDBMS (`EXEC SQL`instructions).

Apatridie et gestion des sessions

L'une des fonctionnalités importantes du AWS Blu Age Runtime est de permettre des scénarios de haute disponibilité (HA) et d'évolutivité horizontale lors de l'exécution de programmes modernisés.

La pierre angulaire de cette situation est l'apatridie, dont un exemple important est la gestion des sessions HTTP.

Gestion des sessions

Tomcat étant basé sur le Web, un mécanisme important pour cela est la gestion des sessions HTTP (telle que fournie par Tomcat et Spring) et la conception apatride. En tant que telle, la conception de l'apatridie repose sur les éléments suivants :

- les utilisateurs se connectent via HTTPS,
- les serveurs d'applications sont déployés derrière un équilibreur de charge,
- lorsqu'un utilisateur se connecte pour la première fois à l'application, celle-ci est authentifiée et le serveur d'applications crée un identifiant (généralement dans un cookie)
- cet identifiant sera utilisé comme clé pour enregistrer et récupérer le contexte utilisateur vers/depuis un cache externe (magasin de données).

La gestion des cookies est effectuée automatiquement par le framework AWS Blu Age et le serveur Tomcat sous-jacent, ce qui est transparent pour l'utilisateur. Le navigateur Internet de l'utilisateur gérera cela automatiquement.

L'application Web Gapwalk peut stocker l'état de la session (le contexte) dans différents magasins de données :

- Amazon ElastiCache (Redis OSS)
- Cluster Redis
- dans la carte mémoire (uniquement pour les environnements de développement et autonomes, ne convient pas à HA).

Haute disponibilité et apatridie

Plus généralement, l'un des principes de conception du framework AWS Blu Age est l'apatridie : la plupart des états non transitoires nécessaires pour reproduire le comportement des programmes

existants ne sont pas stockés sur les serveurs d'applications, mais partagés via une « source unique de vérité » externe commune.

Les files d'attente de stockage temporaire ou les définitions de ressources de CICS sont des exemples de tels états, et les stockages externes typiques pour ces derniers sont les serveurs compatibles Redis ou les bases de données relationnelles.

Cette conception, combinée à l'équilibrage de charge et aux sessions partagées, permet de distribuer la plupart des dialogues destinés aux utilisateurs (OLTP, « Traitement transactionnel en ligne ») entre plusieurs « nœuds » (ici, les instances Tomcat).

En effet, un utilisateur peut exécuter une transaction sur n'importe quel serveur sans se soucier de savoir si le prochain appel de transaction est effectué sur un autre serveur. Ensuite, lorsqu'un nouveau serveur est créé (en raison d'une mise à l'échelle automatique ou pour remplacer un serveur non sain), nous pouvons garantir que tout serveur joignable et sain peut exécuter la transaction comme prévu avec les bons résultats (valeur renvoyée attendue, changement de données attendu dans la base de données, etc.).

AWS Structure Blu Age d'une application modernisée

Ce document fournit des détails sur la structure des applications modernisées (à l'aide des outils de refactoring de modernisation du AWS mainframe), afin que les développeurs puissent accomplir diverses tâches, telles que :

- navigation fluide dans les applications.
- développer des programmes personnalisés qui peuvent être appelés à partir des applications modernisées.
- refactoriser en toute sécurité des applications modernisées.

Nous supposons que vous possédez déjà des connaissances de base dans les domaines suivants :

- les anciens concepts de codage courants, tels que les enregistrements, les ensembles de données et leurs modes d'accès aux enregistrements (indexés, séquentiels), le VSAM, les unités d'exécution, les scripts jcl, les concepts CICS, etc.
- codage Java à l'aide du framework Spring.
- Tout au long du document, nous l'utilisons `short class names` pour des raisons de lisibilité. Pour plus d'informations, voir AWS Mappages de noms entièrement qualifiés Blu Age pour récupérer les noms complets correspondants pour les éléments d'exécution de AWS Blu Age

et Mappages de noms entièrement qualifiés par des tiers pour récupérer les noms complets correspondants pour les éléments tiers.

- Tous les artefacts et échantillons sont extraits des résultats du processus de modernisation de l'exemple d'application COBOL/CICSCardDemo .

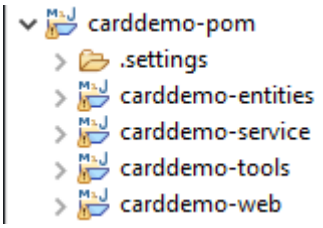
Rubriques

- Organisation des artefacts
- Programmes en cours d'exécution et d'appel
- Écrivez votre propre programme
- Mappages de noms complets

Organisation des artefacts

AWS Les applications modernisées de Blu Age sont regroupées sous forme d'applications Web Java (.war), que vous pouvez déployer sur un serveur JEE. Généralement, le serveur est une instance Tomcat qui intègre le AWS Blu Age Runtime, qui est actuellement basé sur les frameworks Springboot et Angular (pour la partie interface utilisateur).

La guerre regroupe plusieurs artefacts de composants (.jar). Chaque fichier jar est le résultat de la compilation (à l'aide de l'outil maven) d'un projet Java dédié dont les éléments sont le résultat du processus de modernisation.



L'organisation de base repose sur la structure suivante :

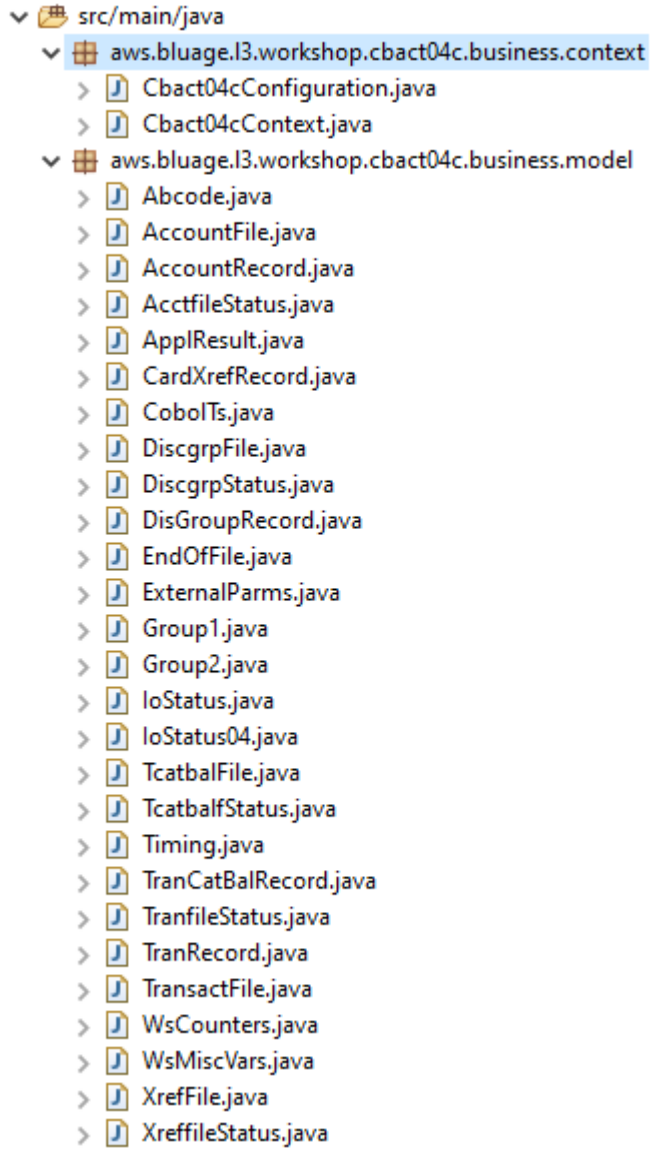
- Projet Entities : contient des éléments de modèle commercial et de contexte. Le nom du projet se termine généralement par « -entities ». Généralement, pour un ancien programme COBOL donné, cela correspond à la modernisation de la section E/S (ensembles de données) et de la division des données. Vous pouvez avoir plusieurs projets d'entités.
- Projet de service : contient des éléments de modernisation de la logique métier héritée. Il s'agit généralement de la division des procédures d'un programme COBOL. Vous pouvez avoir plusieurs projets de service.

- Projet utilitaire : contient des outils et utilitaires communs partagés, utilisés par d'autres projets.
- Projet Web : contient la modernisation des éléments liés à l'interface utilisateur, le cas échéant. Non utilisé pour les projets de modernisation par lots uniquement. Ces éléments d'interface utilisateur peuvent provenir de cartes CICS BMS, de composants IMS MFS et d'autres sources d'interface utilisateur du mainframe. Vous pouvez avoir plusieurs projets Web.

Entités, contenu du projet

> **Note**
>
> Les descriptions suivantes s'appliquent uniquement aux sorties de modernisation COBOL et PL/I. Les sorties de modernisation des RPG sont basées sur une disposition différente.

Avant toute refactorisation, l'organisation des packages dans le projet d'entités est liée aux programmes modernisés. Vous pouvez y parvenir de différentes manières. La méthode préférée consiste à utiliser la boîte à outils Refactoring, qui fonctionne avant que vous ne déclenchiez le mécanisme de génération de code. Il s'agit d'une opération avancée, qui est expliquée dans les BluAge formations. Pour plus d'informations, consultez la section Atelier de refactorisation. Cette approche vous permet de conserver la possibilité de régénérer le code Java ultérieurement, afin de bénéficier de nouvelles améliorations dans le futur, par exemple). L'autre méthode consiste à effectuer une refactorisation Java régulière, directement sur le code source généré, en utilisant n'importe quelle approche de refactorisation Java que vous souhaiteriez appliquer, à vos risques et périls.



Cours liés au programme

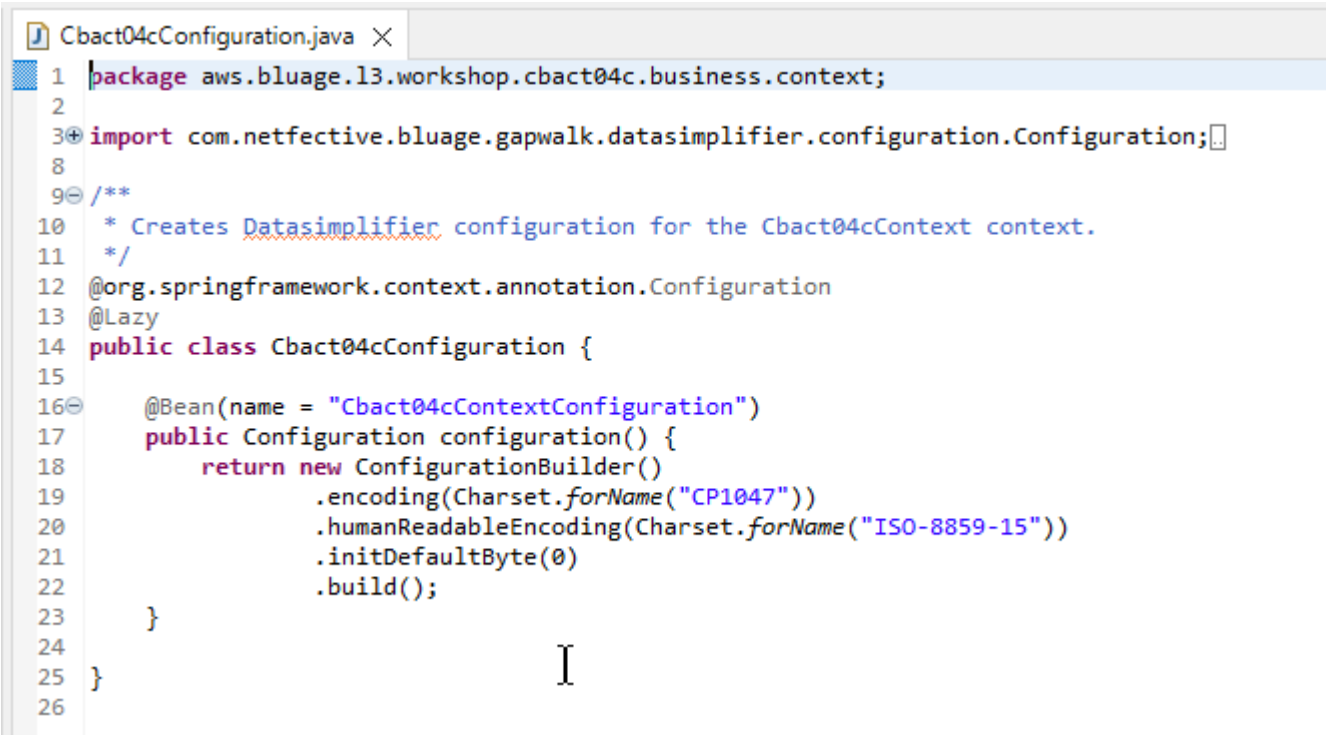
Chaque programme modernisé est associé à deux packages, un package business.context et un package business.model.

- *`base package`*`.`*`program`*`.business.context`

  Le sous-package business.context contient deux classes, une classe de configuration et une classe de contexte.

  - Une classe de configuration pour le programme, qui contient les détails de configuration spécifiques au programme donné, tels que le jeu de caractères à utiliser pour représenter les éléments de données basés sur des caractères, la

valeur d'octet par défaut pour le remplissage des éléments de structure de données, etc. Le nom de la classe se termine par « Configuration ». Il est marqué par l'`@org.springframework.context.annotation.Configuration`annotation et contient une méthode unique qui doit renvoyer un `Configuration` objet correctement configuré.


Cbact04cConfiguration.java

```
1  package aws.bluage.l3.workshop.cbact04c.business.context;
2
3  import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
8
9  /**
10  * Creates Datasimplifier configuration for the Cbact04cContext context.
11  */
12 @org.springframework.context.annotation.Configuration
13 @Lazy
14 public class Cbact04cConfiguration {
15
16     @Bean(name = "Cbact04cContextConfiguration")
17     public Configuration configuration() {
18         return new ConfigurationBuilder()
19                 .encoding(Charset.forName("CP1047"))
20                 .humanReadableEncoding(Charset.forName("ISO-8859-15"))
21                 .initDefaultByte(0)
22                 .build();
23     }
24
25 }
26
```

- Une classe de contexte, qui sert de pont entre les classes de service du programme (voir ci-dessous) et les structures de données (`Record`) et les ensembles de données (`File`) du sous-package du modèle (voir ci-dessous). Le nom de classe se termine par « Context » et est une sous-classe de la `RuntimeContext` classe.

Cbact04cContext.java

```java
@Component("aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext")
@Import({
    aws.bluage.l3.workshop.cbact04c.business.model.TcatbalFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.XrefFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.AccountFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.TransactFile.class
})
@Lazy
@Scope("prototype")
public class Cbact04cContext extends JicsRuntimeContext {

    @Autowired
    private TcatbalFile tcatbalFile;

    @Autowired
    private XrefFile xrefFile;

    @Autowired
    private DiscgrpFile discgrpFile;

    @Autowired
    private AccountFile accountFile;

    @Autowired
    private TransactFile transactFile;

    private IndexedFile tcatbalFileFile;

    private IndexedFile xrefFileFile;

    private IndexedFile discgrpFileFile;

    private IndexedFile accountFileFile;

    private SequentialFile transactFileFile;

    private TranCatBalRecord tranCatBalRecord;
    private TcatbalfStatus tcatbalfStatus;
    private CardXrefRecord cardXrefRecord;
```

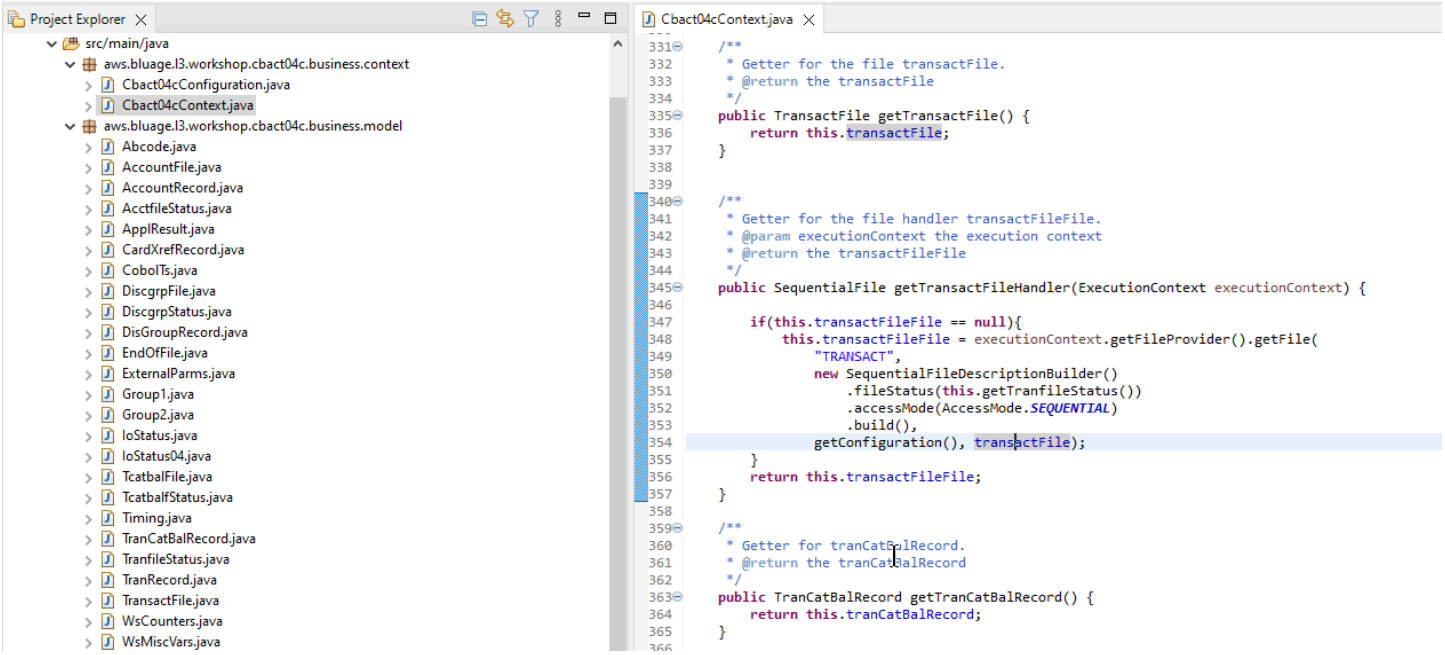
- *base package*.*program*.business.model

  Le sous-package du modèle contient toutes les structures de données que le programme donné peut utiliser. Par exemple, toute structure de données COBOL de niveau 01 correspond à une classe du sous-package du modèle (les structures de données de niveau inférieur sont les propriétés de leur propre structure de niveau 01). Pour plus d'informations sur la façon dont nous modernisons les structures de données 01, consultezQue sont les simplificateurs de données dans AWS Blu Age.

DiscgrpFile.java

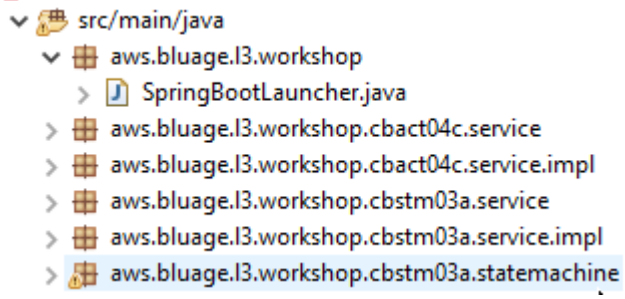
```java
package aws.bluage.l3.workshop.cbact04c.business.model;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group;
import com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

/**
 * Data simplifier file DiscgrpFile.
 *
 * <p>About 'fdDiscgrpRec' field, <br>uml entity: aws.bluage.l3.workshop.cbact04c.business.model.FdDiscgrpRec
 * <br></p>
 *
 */
@Component("aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile")
@Lazy
@Scope("prototype")
public class DiscgrpFile extends RecordEntity {

    private final Group root = new Group(getData());
    private final Group fdDiscgrpRec = new Group(root);
    private final Group fdDiscgrpKey = new Group(fdDiscgrpRec);
    private final Elementary fdDisAcctGroupId = new Elementary(fdDiscgrpKey,new AlphanumericType(10));
    private final Elementary fdDisTranTypeCd = new Elementary(fdDiscgrpKey,new AlphanumericType(2));
    private final Elementary fdDisTranCatCd = new Elementary(fdDiscgrpKey,new ZonedType(4, 0, false));
    private final Elementary fdDiscgrpData = new Elementary(fdDiscgrpRec,new AlphanumericType(34));
```

Toutes les classes étendent la `RecordEntity` classe, qui représente l'accès à une représentation d'enregistrement commercial. Certains enregistrements ont un objectif spécial, car ils sont liés à un`File`. La liaison entre a `Record` et a `File` est établie dans les FileHandler méthodes * correspondantes trouvées dans la classe de contexte lors de la création de l'objet fichier. Par exemple, la liste suivante montre comment le TransactfileFile `File` est lié au TransactFile `Record` (à partir du sous-package du modèle).



Contenu du projet de service

Chaque projet de service est accompagné d'une application Springboot dédiée, qui est utilisée comme colonne vertébrale de l'architecture. Cela se matérialise par le biais de la classe nomméeSpringBootLauncher, située dans le package de base des sources Java du service :



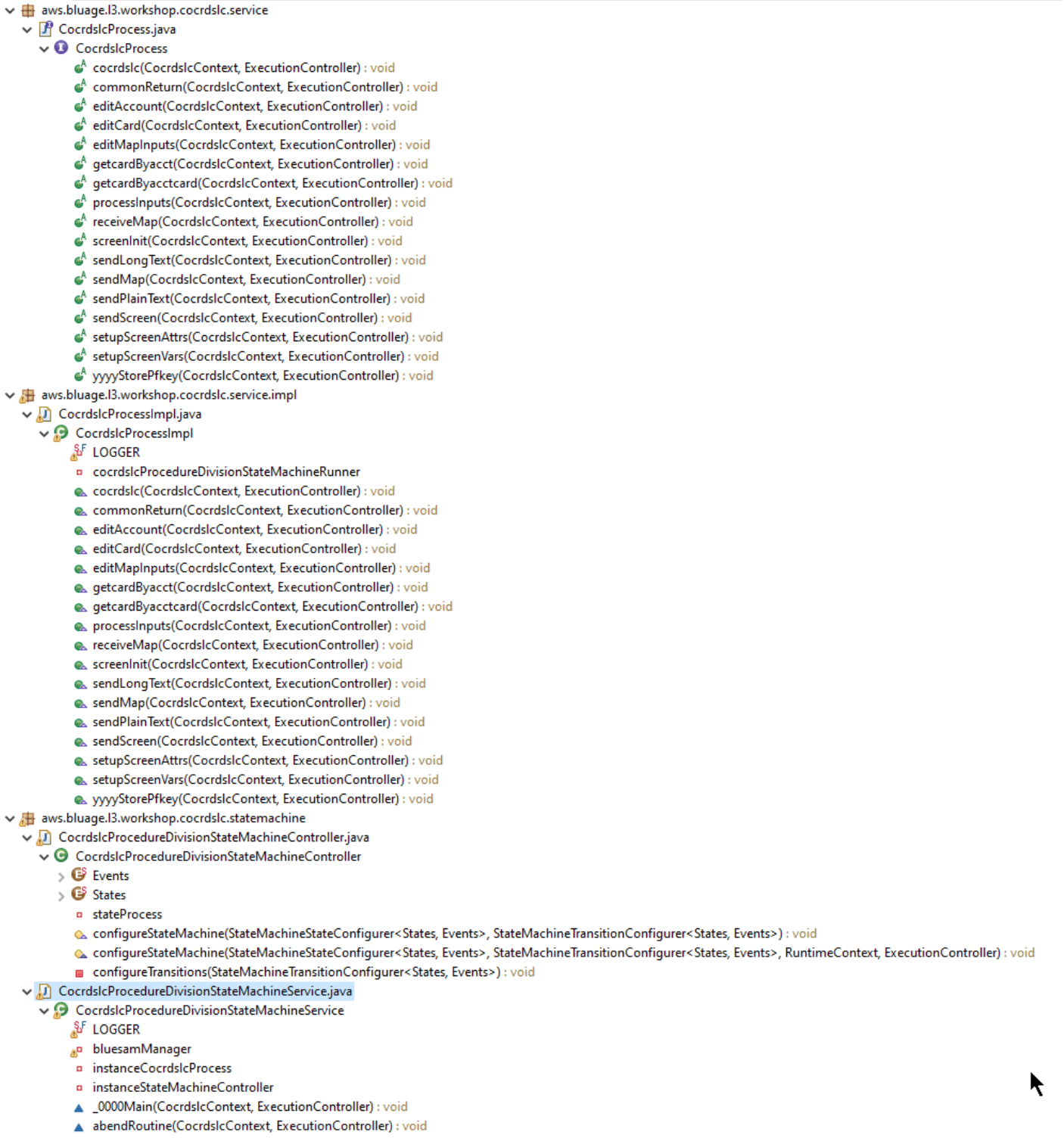
Cette classe est notamment chargée de :

- faire le lien entre les classes du programme et les ressources gérées (sources de données/ gestionnaires de transactions/mappages d'ensembles de données/ etc...).
- fournissant un ou ConfigurableApplicationContext deux programmes.
- découvrir toutes les classes marquées comme spring components (@Component).
- s'assurer que les programmes sont correctement enregistrés dans le ProgramRegistry -- voir la méthode d'initialisation en charge de cet enregistrement.

```
/**
 * Initialization method called when the spring application is ready.
 * Register all programs and services to the gapwalk shared context.
 * @param event the application ready event
 */
@EventListener
public void initialize(ApplicationReadyEvent event) {
    Map<String, ProgramContainer> programContainers = event.getApplicationContext().getBeansOfType(ProgramContainer.class);
    programContainers.values().forEach(ProgramRegistry::registerProgram);
    Map<String, ServiceContainer> serviceContainers = event.getApplicationContext().getBeansOfType(ServiceContainer.class);
    serviceContainers.values().forEach(ServiceRegistry::registerService);
}
```

Artefacts liés au programme

Sans refactorisation préalable, les résultats de la modernisation de la logique métier sont organisés en deux ou trois packages par programme existant :



Le cas le plus exhaustif comportera trois packages :

- *base package*.*program*.service: contient une interface nommée Program Process, qui dispose de méthodes métier pour gérer la logique métier, en préservant le flux de contrôle d'exécution existant.
- *base package*.*program*.service.impl: contient une classe nommée ProgramProcessImpl, qui est l'implémentation de l'interface Process décrite précédemment. C'est ici que les anciennes instructions sont « traduites » en instructions Java, en s'appuyant sur le framework AWS Blu Age :

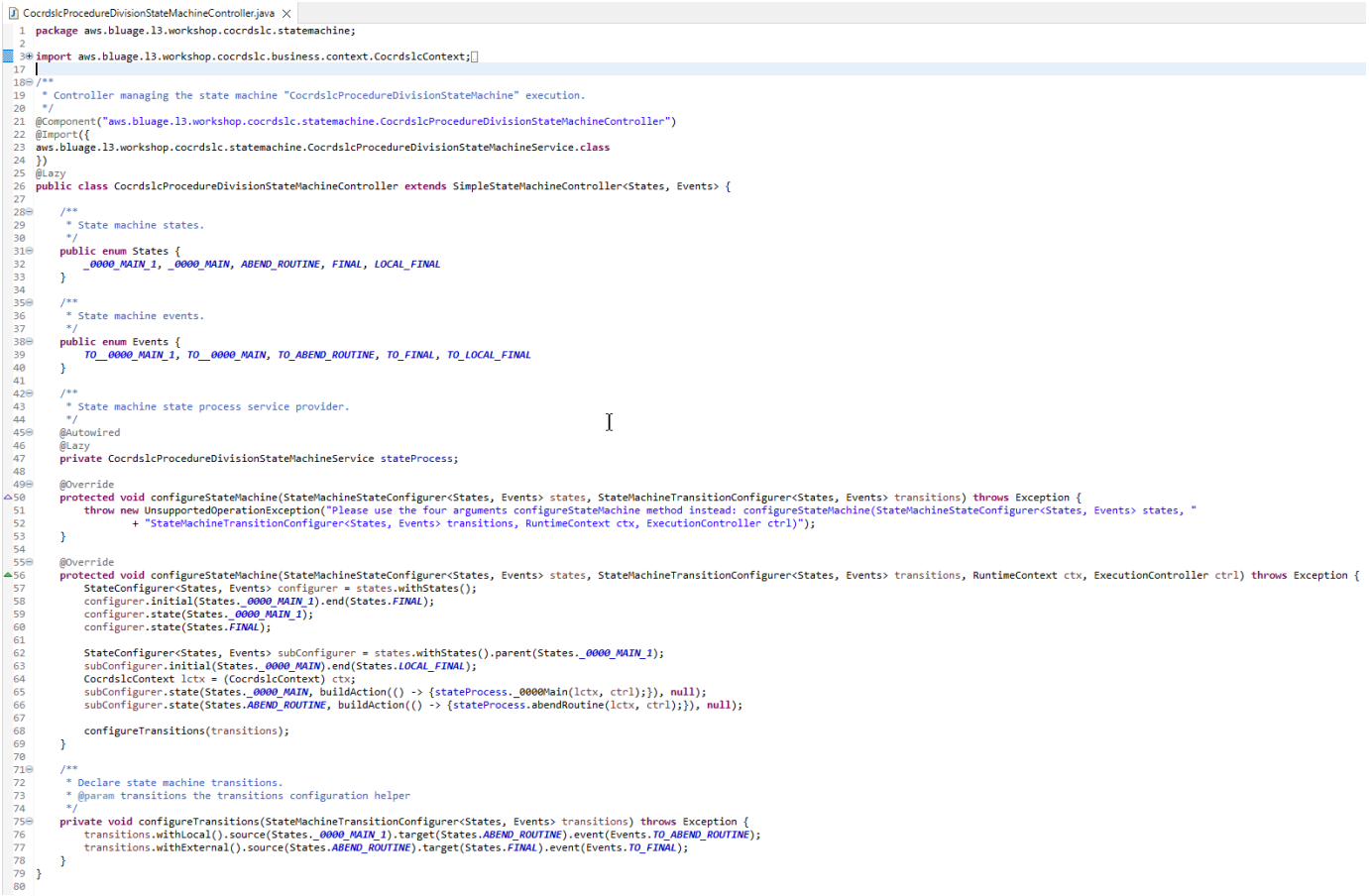
CocrdslcProcessImpl.java

```
210     /**
211      * Process operation sendScreen.
212      *
213      * @param ctx
214      * @param ctrl
215      */
216     @Override
217     public void sendScreen(final CocrdslcContext ctx, final ExecutionController ctrl) {
218         ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
219         ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
220         ctx.getCarddemoCommarea().setCdemoPgmReenter(true);
221         SendMapBuilder.newInstance(ctx.getDfheiblk(), ctx)
222         .withMap(ctx.getCcWorkAreas().getCcardNextMap())
223         .withMapset(ctx.getCcWorkAreas().getCcardNextMapset())
224         .withData(ctx.getGroup1().getCcrdslaoReference())
225         .withCursor()
226         .withErase()
227         .withFreeKB()
228         .execute();
229         ctx.getWsMiscStorage().setWsRespCd(ctx.getDfheiblk().getEibresp());
230     }
231
232     /**
233      * Process operation processInputs.
234      *
235      * @param ctx
236      * @param ctrl
237      */
238     @Override
239     public void processInputs(final CocrdslcContext ctx, final ExecutionController ctrl) {
240         receiveMap(ctx, ctrl);
241         editMapInputs(ctx, ctrl);
242         ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
243         ctx.getCcWorkAreas().setCcardNextProg(ctx.getWsLiterals().getLitThispgm());
244         ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
245         ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
246     }
247
```

- *base package*.*program*.statemachine: ce package n'est peut-être pas toujours présent. Cela est nécessaire lorsque la modernisation de l'ancien flux de contrôle doit utiliser une approche basée sur une machine à états (notamment en utilisant le StateMachine framework Spring) pour couvrir correctement le flux d'exécution existant.

  Dans ce cas, le sous-package statemachine contient deux classes :

- *Program*`ProcedureDivisionStateMachineController`: une classe qui étend une classe implémentant les interfaces `StateMachineController` (définition des opérations nécessaires pour contrôler l'exécution d'une machine à états) et `StateMachineRunner` (définition des opérations requises pour exécuter une machine à états), utilisées pour piloter la mécanique des machines à états Spring ; par exemple, `SimpleStateMachineController` comme dans le cas d'exemple.



```
CocrdslcProcedureDivisionStateMachineController.java
package aws.bluage.l3.workshop.cocrdslc.statemachine;

import aws.bluage.l3.workshop.cocrdslc.business.context.CocrdslcContext;

/**
 * Controller managing the state machine "CocrdslcProcedureDivisionStateMachine" execution.
 */
@Component("aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineController")
@Import({
aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineService.class
})
@Lazy
public class CocrdslcProcedureDivisionStateMachineController extends SimpleStateMachineController<States, Events> {

    /**
     * State machine states.
     */
    public enum States {
        _0000_MAIN_1, _0000_MAIN, ABEND_ROUTINE, FINAL, LOCAL_FINAL
    }

    /**
     * State machine events.
     */
    public enum Events {
        TO__0000_MAIN_1, TO__0000_MAIN, TO_ABEND_ROUTINE, TO_FINAL, TO_LOCAL_FINAL
    }

    /**
     * State machine state process service provider.
     */
    @Autowired
    @Lazy
    private CocrdslcProcedureDivisionStateMachineService stateProcess;

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        throw new UnsupportedOperationException("Please use the four arguments configureStateMachine method instead: configureStateMachine(StateMachineStateConfigurer<States, Events> states, "
                + "StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl)");
    }

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl) throws Exception {
        StateConfigurer<States, Events> configurer = states.withStates();
        configurer.initial(States._0000_MAIN_1).end(States.FINAL);
        configurer.state(States._0000_MAIN_1);
        configurer.state(States.FINAL);

        StateConfigurer<States, Events> subConfigurer = states.withStates().parent(States._0000_MAIN_1);
        subConfigurer.initial(States._0000_MAIN).end(States.LOCAL_FINAL);
        CocrdslcContext lctx = (CocrdslcContext) ctx;
        subConfigurer.state(States._0000_MAIN, buildAction(() -> {stateProcess._0000Main(lctx, ctrl);}), null);
        subConfigurer.state(States.ABEND_ROUTINE, buildAction(() -> {stateProcess.abendRoutine(lctx, ctrl);}), null);

        configureTransitions(transitions);
    }

    /**
     * Declare state machine transitions.
     * @param transitions the transitions configuration helper
     */
    private void configureTransitions(StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        transitions.withLocal().source(States._0000_MAIN_1).target(States.ABEND_ROUTINE).event(Events.TO_ABEND_ROUTINE);
        transitions.withExternal().source(States.ABEND_ROUTINE).target(States.FINAL).event(Events.TO_FINAL);
    }
}
```

Le contrôleur de machine à états définit les différents états possibles et les transitions entre eux, qui reproduisent le flux de contrôle d'exécution existant pour le programme donné.

Lors de la création de la machine à états, le contrôleur fait référence aux méthodes définies dans la classe de service associée située dans le package de machine à états et décrite ci-dessous :

```
subConfigurer.state(States._0000_MAIN, buildAction(() ->
 {stateProcess._0000Main(lctx, ctrl);}), null);
subConfigurer.state(States.ABEND_ROUTINE, buildAction(() ->
 {stateProcess.abendRoutine(lctx, ctrl);}), null);
```

- *Program*ProcedureDivisionStateMachineService: cette classe de service représente une certaine logique métier qui doit être liée à la machine d'état créée par le contrôleur de machine d'état, comme décrit précédemment.

  Le code des méthodes de cette classe utilise les événements définis dans le contrôleur State Machine :

CocrdslcProcedureDivisionStateMachineService.java

```
59    /**
60     * State process operation _0000Main.
61     *
62     * @param ctx
63     * @param ctrl
64     */
65    void _0000Main(CocrdslcContext ctx, ExecutionController ctrl) {
66        ctx.getDfheiblk().bind(ArgUtils.get(ctx, 0));
67        ctx.getDfhcommarea().bind(ArgUtils.get(ctx, 1));
68
69        /*
70        ***************************************************************
71        Program:     COCRDSLC.CBL                                     *
72        Layer:       Business logic                                   *
73        Function:    Accept and process credit card detail request    *
74        ***************************************************************
75        Copyright Amazon.com, Inc. or its affiliates.
76        All Rights Reserved.
77        Licensed under the Apache License, Version 2.0 (the \"License\").
78        You may not use this file except in compliance with the License.
79        You may obtain a copy of the License at
80        http://www.apache.org/licenses/LICENSE-2.0
81        Unless required by applicable law or agreed to in writing,
82        software distributed under the License is distributed on an
83        \"AS IS\" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,
84        either express or implied. See the License for the specific
85        language governing permissions and limitations under the License
86        ***************************************************************
87        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:16:00 CDT */
88        instanceStateMachineController.registerSignalHandler(Events.TO_ABEND_ROUTINE, "!ABEND");
89        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).execute().handleException();
90        ctx.getCcWorkAreas().getCcWorkAreaReference().getField().initialize();
91        ctx.getWsMiscStorage().getField().initialize();
92        DataUtils.initialize(ctx.getWsCommarea().getWsCommareaReference());
93
```

CocrdslcProcedureDivisionStateMachineService.java

```
221      *
222      * @param ctx
223      * @param ctrl
224      */
225     void abendRoutine(CocrdslcContext ctx, ExecutionController ctrl) {
226         if (DataUtils.isLowValue(ctx.getAbendData().getAbendMsgReference())) {
227             ctx.getAbendData().setAbendMsg("UNEXPECTED ABEND OCCURRED.");
228         }
229         ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
230         SendTextBuilder.newInstance(ctx.getDfheiblk(), ctx)
231         .withData(ctx.getAbendData())
232         .withLength(134)
233         .execute();
234         HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).cancel().execute().handleException();
235         AbendBuilder.newInstance(ctx.getDfheiblk(), ctx).withAbendCode("9999").execute().handleException();
236
237         /*
238         Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:12:33 CDT */
239         instanceStateMachineController.sendEvent(Events.TO_FINAL);
240
241     }
242 }
243
```

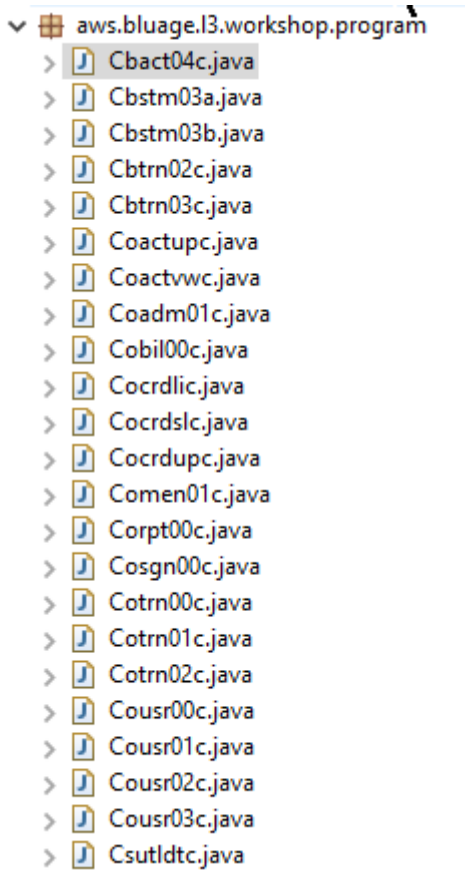
Le service statemachine appelle également l'implémentation du service de processus décrite précédemment :

CocrdslcProcedureDivisionStateMachineService.java

```
166
167         /*
168         ************************************************************
169         COMING FROM CREDIT CARD LIST SCREEN
170         SELECTION CRITERIA ALREADY VALIDATED
171         ************************************************************ */
172     } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter() && DataUtils.compare(ctx.getCarddemoCommarea().getCdemoFromProgramReference(), ctx.getWsLiterals().getLitCclistpgmReference()) == 0) {
173         ctx.getWsMiscStorage().setInputOk(true);
174         ctx.getCcWorkAreas().setCcAcctIdN(ctx.getCarddemoCommarea().getCdemoAcctId());
175         ctx.getCcWorkAreas().setCcCardNumN(ctx.getCarddemoCommarea().getCdemoCardNum());
176         instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
177         instanceCocrdslcProcess.sendMap(ctx, ctrl);
178         instanceCocrdslcProcess.commonReturn(ctx, ctrl);
179     } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter()) {
180
181         /*
182         ************************************************************
183         COMING FROM SOME OTHER CONTEXT
184         SELECTION CRITERIA TO BE GATHERED
185         ************************************************************ */
186         instanceCocrdslcProcess.sendMap(ctx, ctrl);
187         instanceCocrdslcProcess.commonReturn(ctx, ctrl);
188     } else if (ctx.getCarddemoCommarea().isCdemoPgmReenter()) {
189         instanceCocrdslcProcess.processInputs(ctx, ctrl);
190         if (ctx.getWsMiscStorage().isInputError()) {
191             instanceCocrdslcProcess.sendMap(ctx, ctrl);
192             instanceCocrdslcProcess.commonReturn(ctx, ctrl);
193         } else {
194             instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
195             instanceCocrdslcProcess.sendMap(ctx, ctrl);
196             instanceCocrdslcProcess.commonReturn(ctx, ctrl);
197         }
198     } else {
199         ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
200         ctx.getAbendData().setAbendCode("0001");
201         DataUtils.setToBlank(ctx.getAbendData().getAbendReasonReference());
202         ctx.getWsMiscStorage().setWsReturnMsg("UNEXPECTED DATA SCENARIO");
203         instanceCocrdslcProcess.sendPlainText(ctx, ctrl);
204     }
205
206     /*
207     If we had an error setup error message that slipped through
208     Display and return */
209     if (ctx.getWsMiscStorage().isInputError()) {
210         ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
211         instanceCocrdslcProcess.sendMap(ctx, ctrl);
212         instanceCocrdslcProcess.commonReturn(ctx, ctrl);
213     }
214     instanceCocrdslcProcess.commonReturn(ctx, ctrl);
```

De plus, un package nommé *`base package`*`.program` joue un rôle important, car il regroupe une classe par programme, qui servira de point d'entrée au programme (plus de détails à ce sujet

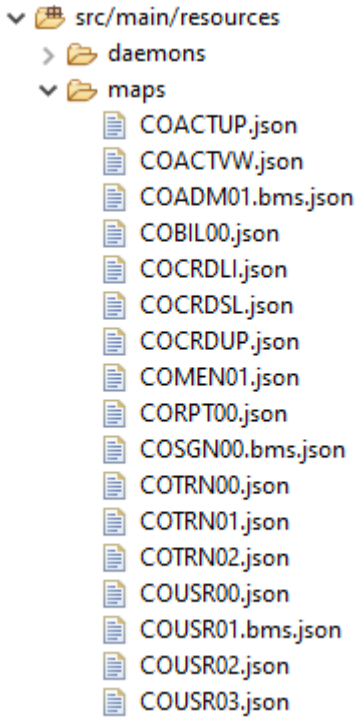
ultérieurement). Chaque classe implémente l'`Program`interface, marqueur d'un point d'entrée du programme.



Autres artefacts

- Compagnons BMS MAPs

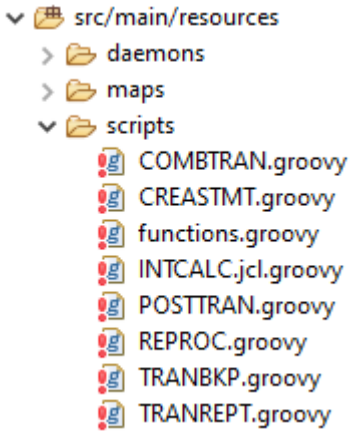
  Outre les artefacts liés au programme, le projet de service peut contenir d'autres artefacts à des fins diverses. Dans le cas de la modernisation d'une application en ligne CICS, le processus de modernisation produit un fichier json et place dans le dossier map du the /src/main/resources dossier :



Le moteur d'exécution Blu Age utilise ces fichiers json pour lier les enregistrements utilisés par l'instruction SEND MAP aux champs d'écran.

- Scripts géniaux

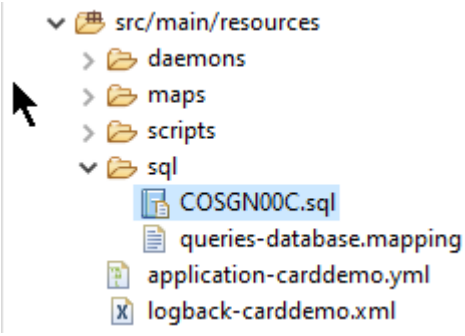
  Si l'ancienne application comportait des scripts JCL, ceux-ci ont été modernisés en scripts groovy, stockés dans un the /src/main/resources/scripts dossier (nous reviendrons sur cet emplacement spécifique ultérieurement) :



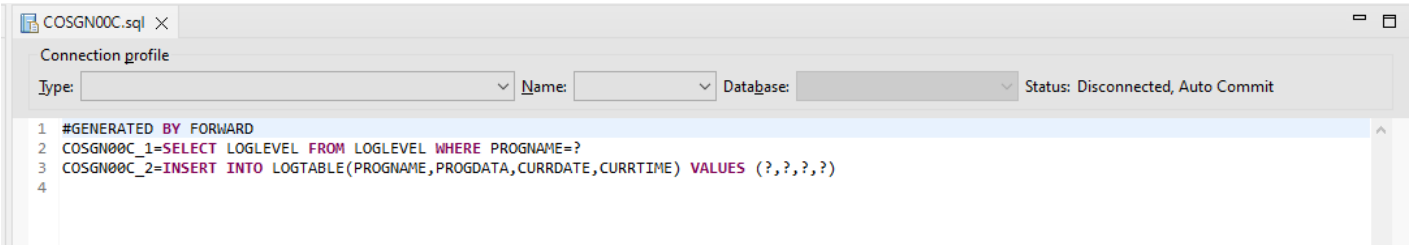
  Ces scripts sont utilisés pour lancer des tâches par lots (charges de travail de traitement de données dédiées, non interactives et gourmandes en ressources processeur).

- fichiers SQL

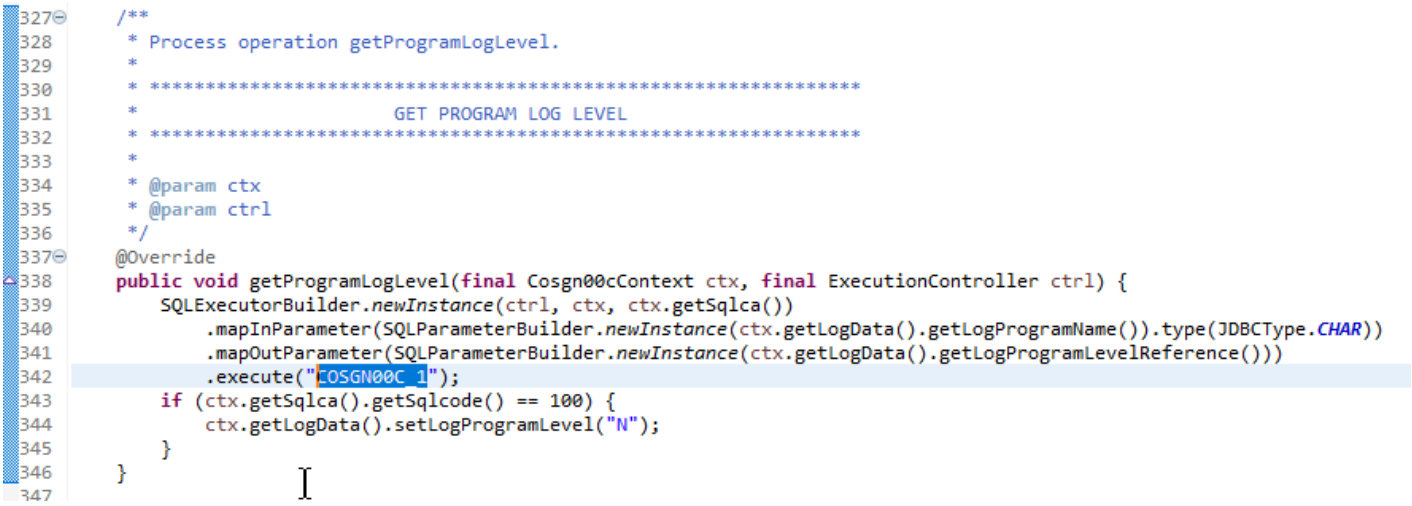
Si l'ancienne application utilisait des requêtes SQL, les requêtes SQL modernisées correspondantes ont été rassemblées dans des fichiers de propriétés dédiés, avec le modèle de dénomination program .sql, où program est le nom du programme utilisant ces requêtes.



Le contenu de ces fichiers SQL est une collection d'entrées (clé=requête), où chaque requête est associée à une clé unique, que le programme modernisé utilise pour exécuter la requête donnée :



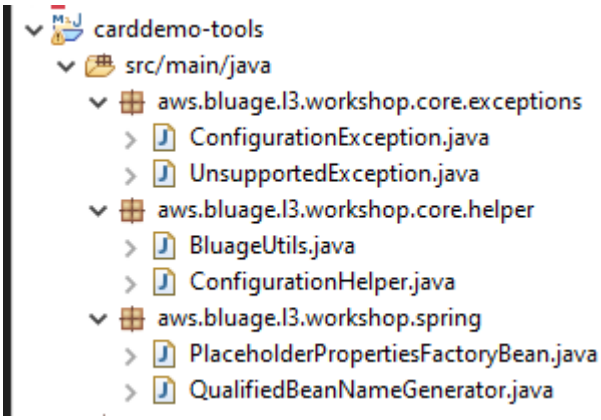
Par exemple, le programme COSGN00C exécute la requête avec la clé « COSGN00C_1 » (la première entrée du fichier SQL) :



```
    /**
     * Process operation getProgramLogLevel.
     *
     * ****************************************************************
     *                       GET PROGRAM LOG LEVEL
     * ****************************************************************
     *
     * @param ctx
     * @param ctrl
     */
    @Override
    public void getProgramLogLevel(final Cosgn00cContext ctx, final ExecutionController ctrl) {
        SQLExecutorBuilder.newInstance(ctrl, ctx, ctx.getSqlca())
            .mapInParameter(SQLParameterBuilder.newInstance(ctx.getLogData().getLogProgramName()).type(JDBCType.CHAR))
            .mapOutParameter(SQLParameterBuilder.newInstance(ctx.getLogData().getLogProgramLevelReference()))
            .execute("COSGN00C_1");
        if (ctx.getSqlca().getSqlcode() == 100) {
            ctx.getLogData().setLogProgramLevel("N");
        }
    }
```

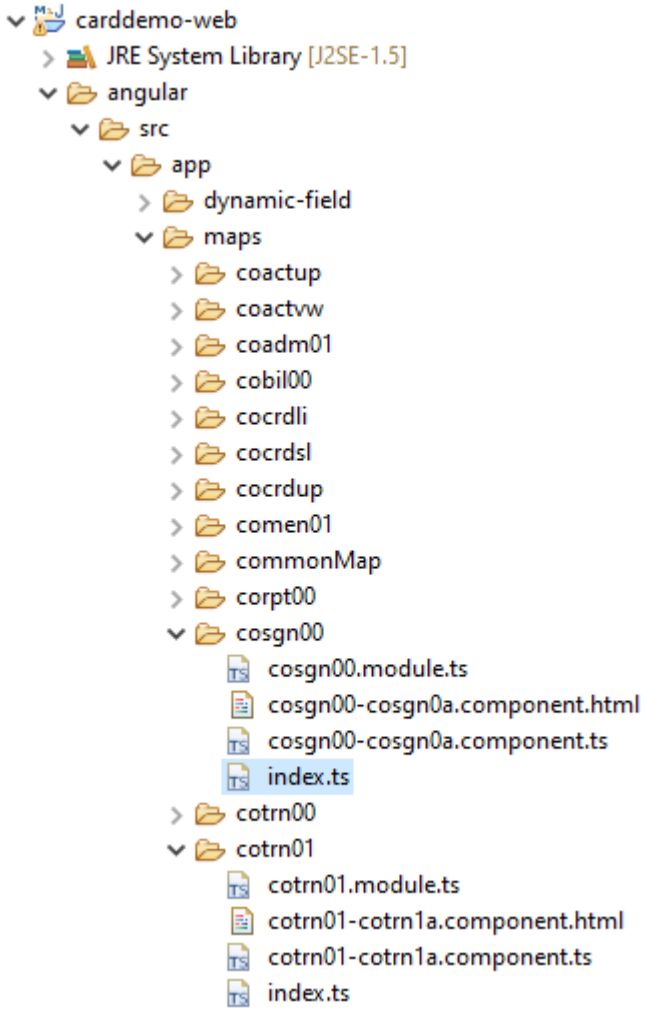
Contenu du projet Utilities

Le projet d'utilitaires, dont le nom se termine par « -tools », contient un ensemble d'utilitaires techniques qui peuvent être utilisés par tous les autres projets.



Contenu du (des) projet (s) Web

Le projet Web n'est présent que lors de la modernisation des anciens éléments de l'interface utilisateur. Les éléments d'interface utilisateur modernes utilisés pour créer le front-end d'application modernisé sont basés sur Angular. L'exemple d'application utilisé pour présenter les artefacts de modernisation est une application COBOL/CICS exécutée sur un ordinateur central. Le système CICS utilise MAPs pour représenter les écrans de l'interface utilisateur. Les éléments modernes correspondants seront, pour chaque carte, un fichier html accompagné de fichiers Typescript :



Le projet Web ne prend en charge que l'aspect frontal de l'application. Le projet de service, qui repose sur les projets d'utilitaires et d'entités, fournit les services principaux. Le lien entre le front-end et le backend est établi via l'application Web nommée Gapwalk-Application, qui fait partie de la distribution d'exécution standard AWS de Blu Age.

Programmes en cours d'exécution et d'appel

Sur les anciens systèmes, les programmes sont compilés sous forme d'exécutables autonomes qui peuvent s'appeler eux-mêmes via un mécanisme CALL, tel que l'instruction COBOL CALL, en transmettant des arguments lorsque cela est nécessaire. Les applications modernisées offrent les mêmes fonctionnalités mais utilisent une approche différente, car la nature des artefacts concernés est différente de celle des anciens artefacts.

Du côté modernisé, les points d'entrée du programme sont des classes spécifiques qui implémentent l'`Program`interface, sont des composants Spring (@Component) et sont situés dans des projets de service, dans un package nommé*`base package`*`.program`.

Inscription aux programmes

Chaque fois que le serveur Tomcat hébergeant les applications modernisées est démarré, le service Springboot est également lancé, ce qui déclenche l'enregistrement des programmes. Un registre dédié nommé `ProgramRegistry` est rempli d'entrées de programme, chaque programme étant enregistré à l'aide de ses identifiants, une entrée par identifiant de programme connu, ce qui signifie que si un programme est connu sous plusieurs identifiants différents, le registre contient autant d'entrées qu'il y a d'identifiants.

L'enregistrement d'un programme donné repose sur la collection d'identifiants renvoyés par la méthode getProgramIdentifiers () :

Cbact04c.java

```java
package aws.bluage.l3.workshop.program;

import aws.bluage.l3.workshop.SpringBootLauncher;

/**
 * Reference the spring application of program CBACT04C.
 *  Provides an access to the contained program for the run unit.
 */
@Component
@Import({
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cConfiguration.class,
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext.class,
    aws.bluage.l3.workshop.cbact04c.service.impl.Cbact04cProcessImpl.class
})
public class Cbact04c implements Program {
    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("CBACT04C").collect(Collectors.toSet()));

    /**
     * Main program identifier for the contained program.
     */
    private static final String programIdentifier = "CBACT04C";
    @Autowired
    PlatformTransactionManager transactionManager;

    @Autowired
    Map<String, DataSource> datasources;
    @Autowired
    BeanFactory beanFactory;
    /**
     * {@inheritDoc}
     */
    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public void updateExecutionContext(ExecutionContext executionContext) {
        executionContext.setDatasources(datasources);
        executionContext.setDatabaseSupport(ExecutionContext.DatabaseSupport.POSTGRE);
        executionContext.setSqlcaVersion(ExecutionContext.SqlcaVersion.getEnum("ansi-comp5"));
        executionContext.setTransactionManager(transactionManager);
        executionContext.setUseSQLDateNewParadigm(true);
        executionContext.setUseSQLTrimStringType(false);
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Dans cet exemple, le programme est enregistré une seule fois, sous le nom « CBACT04C » (regardez le contenu de la collection ProgramIdentifiers). Les journaux Tomcat indiquent chaque enregistrement de programme. L'enregistrement du programme dépend uniquement des identificateurs de programme déclarés et non du nom de classe de programme lui-même (bien que les identifiants de programme et les noms de classe de programme soient généralement alignés).

Le même mécanisme d'enregistrement s'applique aux programmes utilitaires fournis par les différentes applications Web utilitaires AWS Blu Age, qui font partie de la distribution d'exécution

AWS Blu Age. Par exemple, l' Gapwalk-Utility-Pgmapplication Web fournit les équivalents fonctionnels des utilitaires système z/OS (IDCAMS, ICEGENER, SORT, etc.) et peut être appelée par des programmes ou des scripts modernisés. Tous les programmes utilitaires disponibles enregistrés au démarrage de Tomcat sont enregistrés dans les journaux Tomcat.

Enregistrement des scripts et des démons

Un processus d'enregistrement similaire, au moment du démarrage de Tomcat, se produit pour les scripts groovy situés dans the /src/main/resources/scripts la hiérarchie des dossiers. La hiérarchie des dossiers de scripts est parcourue et tous les scripts groovy découverts (à l'exception du script réservé special functions.groovy) sont enregistrés dans le`ScriptRegistry`, en utilisant leur nom court (la partie du nom du fichier de script située avant le premier point) comme clé de récupération.

> **Note**
>
> - Si plusieurs scripts ont des noms de fichiers qui produiront la même clé d'enregistrement, seule la dernière est enregistrée, remplaçant ainsi toute inscription précédemment rencontrée pour cette clé donnée.
> - Compte tenu de ce qui précède, faites attention lorsque vous utilisez des sous-dossiers, car le mécanisme d'enregistrement aplatit la hiérarchie et peut entraîner des remplacements inattendus. La hiérarchie ne compte pas dans le processus d'enregistrement : typically /scripts/A/myscript.groovy and /scripts/B/myscript.groovy will lead to /scripts/B/myscript.groovy overwriting /scripts/A/myscript .groovy.

Les scripts groovy du the /src/main/resources/daemons dossier sont gérés un peu différemment. Ils sont toujours enregistrés en tant que scripts classiques, mais en plus, ils ne sont lancés qu'une seule fois, directement au moment du démarrage de Tomcat, de manière asynchrone.

Une fois les scripts enregistrés dans le`ScriptRegistry`, un appel REST peut les lancer, en utilisant les points de terminaison dédiés exposés par l'application Gapwalk. Pour plus d'informations, consultez la documentation correspondante.

Programmes d'appel de programmes

Chaque programme peut appeler un autre programme en tant que sous-programme et lui transmettre des paramètres. Pour ce faire, les programmes utilisent une implémentation de l'`ExecutionController`interface (la plupart du temps, il s'agira d'une

`ExecutionControllerImpl` instance), ainsi qu'un mécanisme d'API fluide appelé le `CallBuilder` pour générer les arguments d'appel du programme.

Toutes les méthodes des programmes prennent à la fois a `RuntimeContext` et an `ExecutionController` comme arguments de méthode, de sorte que an `ExecutionController` est toujours disponible pour appeler d'autres programmes.

Voir, par exemple, le schéma suivant, qui montre comment le programme CBST03A appelle le programme CBST03B en tant que sous-programme, en lui transmettant des paramètres :

Cbstm03aProcessImpl.java

```
67      /**
68       * Process operation xreffileGetNext.
69       *
70       * ----------------------------------------------------------------*
71       *
72       * @param ctx
73       * @param ctrl
74       */
75      @Override
76      public void xreffileGetNext(final Cbstm03aContext ctx, final ExecutionController ctrl) {
77          ctx.getWsM03bArea().setWsM03bDd("XREFFILE");
78          ctx.getWsM03bArea().setM03bRead(true);
79          DataUtils.setToZeroes(ctx.getWsM03bArea().getWsM03bRcReference());
80          DataUtils.setToBlank(ctx.getWsM03bArea().getWsM03bFldtReference());
81          ctrl.callSubProgram("CBSTM03B", CallBuilder.newInstance()
82              .byReference(ctx.getWsM03bArea())
83              .getArguments(), ctx);
84          if (DataUtils.compare(ctx.getWsM03bArea().getWsM03bRcReference(), "00") == 0) {
85
86              /*
87              Do nothing */
88          } else if (DataUtils.compare(ctx.getWsM03bArea().getWsM03bRcReference(), "10") == 0) {
89              ctx.getMiscVariables().setEndOfFile("Y");
90          } else {
91              if (LOGGER.isInfoEnabled()) LOGGER.info("ERROR READING XREFFILE");
92              if (LOGGER.isInfoEnabled()) LOGGER.info("{}{}" , "RETURN CODE: " , ctx.getWsM03bArea().getWsM03bRc());
93              abendProgram(ctx, ctrl);
94          }
95          ctx.getCardXrefRecord().setBytes(ctx.getWsM03bArea().getWsM03bFldtReference().getBytes());
96      }
97
```

- Le premier argument de `ExecutionController.callSubProgram` est un identifiant du programme à appeler (c'est-à-dire l'un des identifiants utilisés pour l'enregistrement du programme, voir paragraphes ci-dessus).
- Le deuxième argument, qui est le résultat de la construction sur le`CallBuilder`, est un tableau de`Record`, correspondant aux données transmises de l'appelant à l'appelé.
- Le troisième et dernier argument est l'`RuntimeContext`instance de l'appelant.

Les trois arguments sont obligatoires et ne peuvent pas être nuls, mais le second argument peut être un tableau vide.

L'appelé ne pourra traiter les paramètres transmis que s'il a été initialement conçu pour le faire. Pour un ancien programme COBOL, cela signifie avoir une section LINKAGE et une clause USING pour la division des procédures afin d'utiliser les éléments LINKAGE.

Par exemple, consultez le fichier source COBOL CBSTM03B.CBL correspondant :

github.com/aws-samples/aws-mainframe-modernization-carddemo/blob/main/app/cbl/CBSTM03B.CBL

```
 98
 99          LINKAGE SECTION.
100          01  LK-M03B-AREA.
101              05  LK-M03B-DD          PIC X(08).
102              05  LK-M03B-OPER        PIC X(01).
103                88  M03B-OPEN       VALUE 'O'.
104                88  M03B-CLOSE      VALUE 'C'.
105                88  M03B-READ       VALUE 'R'.
106                88  M03B-READ-K     VALUE 'K'.
107                88  M03B-WRITE      VALUE 'W'.
108                88  M03B-REWRITE    VALUE 'Z'.
109              05  LK-M03B-RC          PIC X(02).
110              05  LK-M03B-KEY         PIC X(25).
111              05  LK-M03B-KEY-LN      PIC S9(4).
112              05  LK-M03B-FLDT        PIC X(1000).
113
114          PROCEDURE DIVISION USING LK-M03B-AREA.
115
```

Le programme CBSTM03B prend donc un seul `Record` comme paramètre (un tableau de taille 1). C'est ce qu'ils sont en train de construire, en utilisant les méthodes de chaînage byReference () et getArguments (). `CallBuilder`

La classe d'API `CallBuilder` Fluent dispose de plusieurs méthodes pour remplir le tableau d'arguments à transmettre à un appelé :

- AsPointer (RecordAdaptable) : ajoute un argument de type pointeur, par référence. Le pointeur représente l'adresse d'une structure de données cible.
- ByReference (RecordAdaptable) : ajoute un argument par référence. L'appelant verra les modifications qu'il effectue.
- ByReference (RecordAdaptable) : variante varargs de la méthode précédente.
- ByValue (Object) : ajoute un argument, transformé en a`Record`, par valeur. L'appelant ne verra pas les modifications effectuées par l'appelé.

- ByValue (RecordAdaptable) : identique à la méthode précédente, mais l'argument est directement disponible sous forme de`RecordAdaptable`.
- byValueWithBounds (Object, int, int) : ajoutez un argument, transformé en a, en extrayant la partie du tableau d'octets définie par les limites données, par valeur. `Record`

Enfin, la méthode getArguments collectera tous les arguments ajoutés et les renverra sous forme de tableau de`Record`.

> **Note**
>
> Il est de la responsabilité de l'appelant de s'assurer que le tableau d'arguments a la taille requise, que les éléments sont correctement ordonnés et compatibles, en termes de disposition de la mémoire, avec les dispositions attendues pour les éléments de liaison.

Scripts appelant des programmes

L'appel de programmes enregistrés à partir de scripts groovy nécessite l'utilisation d'une instance de classe implémentant l'`MainProgramRunner`interface. Habituellement, l'obtention d'une telle instance est obtenue grâce à l' ApplicationContext utilisation de Spring :

REPROC.groovy

```groovy
// Import
import com.netfective.bluage.gapwalk.rt.provider.ScriptRegistry
import com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
import com.netfective.bluage.gapwalk.io.support.FileConfigurationUtils
import com.netfective.bluage.gapwalk.rt.job.support.DefaultJobContext
import com.netfective.bluage.gapwalk.rt.utils.GroovyUtils
import com.netfective.bluage.gapwalk.rt.io.support.FileConfiguration
import com.netfective.bluage.gapwalk.rt.shared.AbendException
import com.netfective.bluage.gapwalk.rt.call.exception.GroovyExecutionException
// Variables
mpr = applicationContext.getBean("com.netfective.bluage.gapwalk.rt.call.ExecutionController", MainProgramRunner.class)
TreeMap mapTransfo = [:]
```

Une fois qu'une `MainProgramRunner` interface est disponible, utilisez la méthode RunProgram pour appeler un programme et transmettre l'identifiant du programme cible en tant que paramètre :

REPROC.groovy

```
//*******************************************************************
//*                              STEPS                              *
//*******************************************************************
// STEP PRC001 - PGM - IDCAMS*****************************************
def stepPRC001(Object shell, Map params, Map programResults){
    shell.with {
        if (checkValidProgramResults(programResults)) {
            return execStep("PRC001", "IDCAMS", programResults, {
                mpr
                    .withFileConfigurations(new FileConfigurationUtils()
                        .systemOut("SYSPRINT")
                        .output("*")
                        .build()
                        .bluesam("FILEIN")
                        .dataset("NULLFILE")
                        .disposition("SHR")
                        .build()
                        .bluesam("FILEOUT")
                        .dataset("NULLFILE")
                        .disposition("SHR")
                        .build()
                        .fileSystem("SYSIN")
                        .path("&CNTLLIB(REPROCT)")
                        .disposition("SHR")
                        .build()
                        .getFileConfigurations(fcmap))
                    .withParameters(params)
                    .runProgram("IDCAMS")
                })
        }
    }
}
```

Dans l'exemple précédent, une étape de travail appelle IDCAMS (programme utilitaire de gestion de fichiers), fournissant un mappage entre les définitions réelles des ensembles de données et leurs identifiants logiques.

Lorsqu'il s'agit d'ensembles de données, les anciens programmes utilisent principalement des noms logiques pour identifier les ensembles de données. Lorsque le programme est appelé à partir d'un script, celui-ci doit associer les noms logiques aux ensembles de données physiques réels. Ces ensembles de données peuvent se trouver sur le système de fichiers, dans un stockage Blusam ou même définis par un flux en ligne, la concaténation de plusieurs ensembles de données ou la génération d'un GDG.

Utilisez withFileConfiguration cette méthode pour créer une carte logique-physique des ensembles de données et la mettre à la disposition du programme appelé.

Écrivez votre propre programme

L'écriture de votre propre programme pour des scripts ou d'autres programmes modernisés à appeler est une tâche courante. Généralement, dans le cadre de projets de modernisation, vous écrivez vos propres programmes lorsqu'un ancien programme exécutable est écrit dans un langage que le processus de modernisation ne prend pas en charge, ou lorsque les sources ont été perdues (oui, cela peut arriver), ou lorsque le programme est un utilitaire dont les sources ne sont pas disponibles.

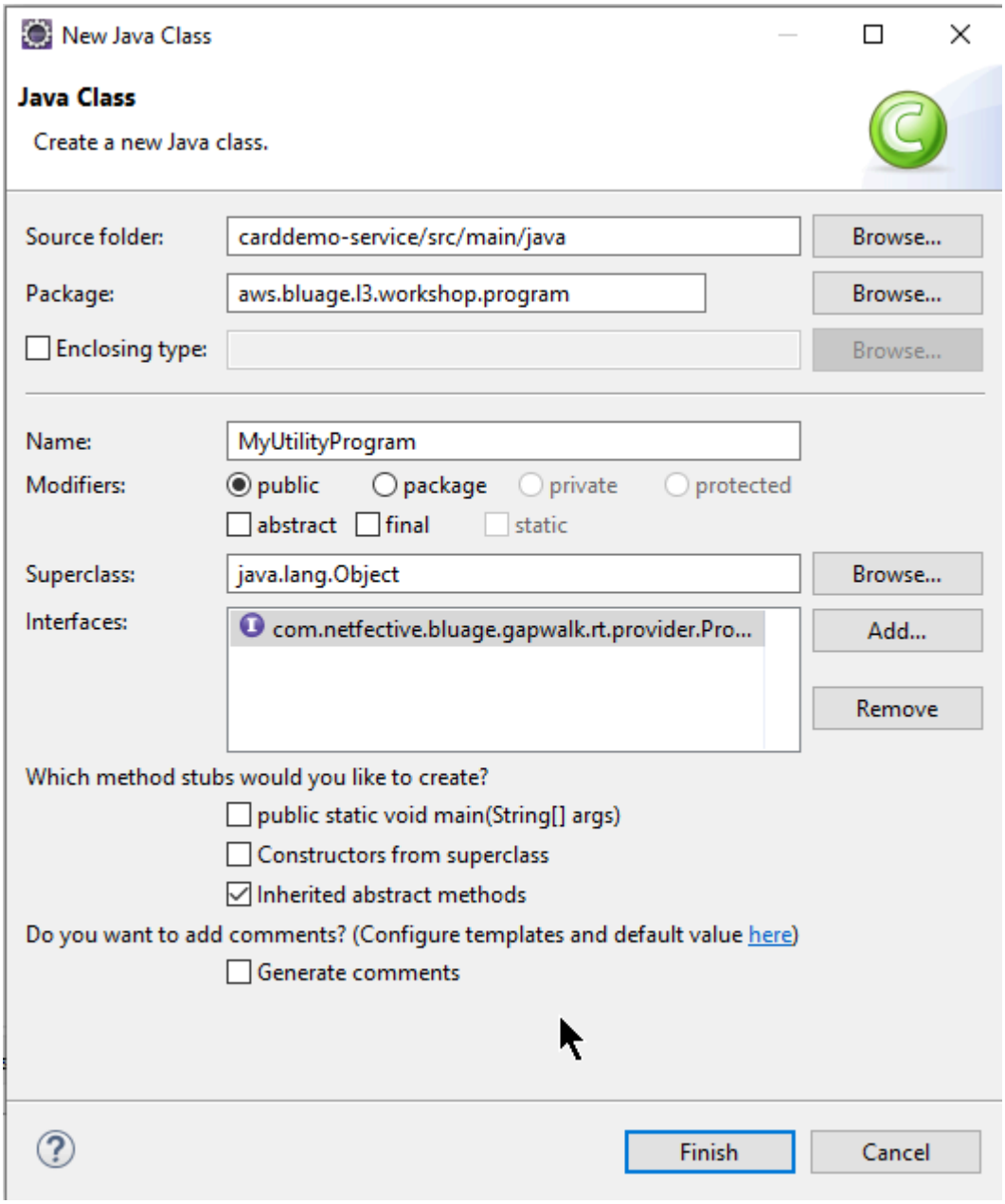
Dans ce cas, vous devrez peut-être écrire vous-même le programme manquant, en Java (en supposant que vous sachiez suffisamment quel devrait être le comportement attendu du programme, la disposition en mémoire des arguments du programme, le cas échéant, etc.) Votre programme Java doit respecter les mécanismes du programme décrits dans ce document, afin que d'autres programmes et scripts puissent l'exécuter.

Pour vous assurer que le programme est utilisable, vous devez effectuer deux étapes obligatoires :

- Écrivez une classe qui implémente correctement l'`Program`interface, afin qu'elle puisse être enregistrée et appelée.
- Assurez-vous que votre programme est correctement enregistré, afin qu'il soit visible depuis d'autres programmes/scripts.

Rédaction de l'implémentation du programme

Utilisez votre IDE pour créer une nouvelle classe Java qui implémente l'`Program`interface :



L'image suivante montre l'IDE Eclipse, qui se charge de créer toutes les méthodes obligatoires à implémenter :

MyUtilityProgram.java

```
1  package aws.bluage.l3.workshop.program;
2
3  import java.util.Set;
10
11 public class MyUtilityProgram implements Program {
12
13     @Override
14     public ConfigurableApplicationContext getSpringApplication() {
15         // TODO Auto-generated method stub
16         return null;
17     }
18
19     @Override
20     public Set<String> getProgramIdentifiers() {
21         // TODO Auto-generated method stub
22         return null;
23     }
24
25     @Override
26     public Context getContext() {
27         // TODO Auto-generated method stub
28         return null;
29     }
30
31     @Override
32     public void run(ExecutionController ctrl) {
33         // TODO Auto-generated method stub
34
35     }
36
37 }
38
```

Intégration Spring

Tout d'abord, la classe doit être déclarée en tant que composant Spring. Annotez la classe avec l'`@Componentannotation` suivante :

```
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.rt.call.ExecutionController;
import com.netfective.bluage.gapwalk.rt.context.Context;
import com.netfective.bluage.gapwalk.rt.provider.Program;

import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
public class MyUtilityProgram implements Program {
```

Ensuite, implémentez correctement les méthodes requises. Dans le contexte de cet exemple, nous avons ajouté le `MyUtilityProgram` package qui contient déjà tous les programmes

modernisés. Cet emplacement permet au programme d'utiliser l'application Springboot existante pour fournir les éléments requis `ConfigurableApplicationContext` pour l'implémentation de la getSpringApplication méthode :

```
public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }
```

Vous pouvez choisir un autre emplacement pour votre propre programme. Par exemple, vous pouvez localiser le programme en question dans un autre projet de service dédié. Assurez-vous que le projet de service donné possède sa propre application Springboot, qui permet de récupérer le ApplicationContext (qui devrait être un`ConfigurableApplicationContext`).

Donner une identité au programme

Pour pouvoir être appelé par d'autres programmes et scripts, le programme doit recevoir au moins un identifiant, qui ne doit entrer en collision avec aucun autre programme enregistré existant dans le système. Le choix de l'identifiant peut être dicté par la nécessité de couvrir le remplacement d'un ancien programme existant ; dans ce cas, vous devrez utiliser l'identifiant attendu, tel que défini dans les occurrences CALL trouvées dans les anciens programmes. La plupart des identificateurs de programmes comportent 8 caractères dans les anciens systèmes.

La création d'un ensemble non modifiable d'identifiants dans le programme est un moyen d'y parvenir. L'exemple suivant montre comment choisir « MYUTILPG » comme identifiant unique :

```
@Component
public class MyUtilityProgram implements Program {

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));


    public ConfigurableApplicationContext getSpringApplication() {

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Associer le programme à un contexte

Le programme a besoin d'une `RuntimeContext` instance associée. Pour les programmes modernisés, AWS Blu Age génère automatiquement le contexte associé, en utilisant les structures de données qui font partie de l'ancien programme.

Si vous écrivez votre propre programme, vous devez également écrire le contexte associé.

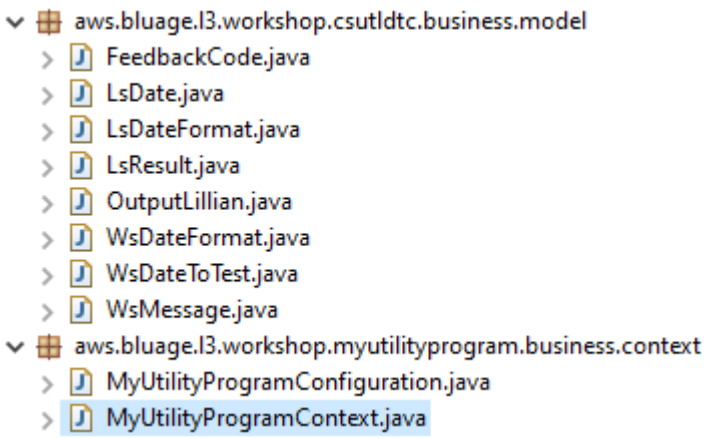
En référence à ce qui suitCours liés au programme, vous pouvez constater qu'un programme nécessite au moins deux cours complémentaires :

- une classe de configuration.
- une classe de contexte qui utilise la configuration.

Si le programme utilitaire utilise une structure de données supplémentaire, elle doit également être écrite et utilisée par le contexte.

Ces classes doivent se trouver dans un package faisant partie d'une hiérarchie de packages qui sera analysée au démarrage de l'application, afin de s'assurer que le composant contextuel et la configuration seront gérés par le framework Spring.

Écrivons une configuration et un contexte minimaux, dans le *`base package`*`.myutilityprogram.business.context` package, fraîchement créé dans le projet entities :



Voici le contenu de la configuration. Il utilise une version de configuration similaire à celle d'autres programmes (modernisés) situés à proximité. Vous devrez probablement le personnaliser en fonction de vos besoins spécifiques.

MyUtilityProgramConfiguration.java

```java
package aws.bluage.l3.workshop.myutilityprogram.business.context;

import java.nio.charset.Charset;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Lazy;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder;

/**
 * Creates Datasimplifier configuration for the MyUtilityProgram context.
 */
@org.springframework.context.annotation.Configuration
@Lazy
public class MyUtilityProgramConfiguration {

    @Bean(name = "MyUtilityProgramContextConfiguration")
    public Configuration configuration() {
        return new ConfigurationBuilder()
                .encoding(Charset.forName("CP1047"))
                .humanReadableEncoding(Charset.forName("ISO-8859-15"))
                .initDefaultByte(0)
                .build();
    }
}
```

Remarques :

- La convention de dénomination générale est ProgramNameConfiguration.
- Il doit utiliser les annotations @org .springframework.context.annotation.Configuration et @Lazy.
- Le nom du haricot suit généralement la ProgramNameContextConfiguration convention, mais cela n'est pas obligatoire. Veillez à éviter les collisions de noms de beans dans le projet.
- La seule méthode à implémenter doit renvoyer un `Configuration` objet. Utilisez l'API `ConfigurationBuilder` Fluent pour vous aider à en créer une.

Et le contexte associé :

MyUtilityProgramContext.java

```java

import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.rt.context.RuntimeContext;

@Component("aws.bluage.l3.workshop.mytuilityprogram.business.context.MyUtilityProgramContext")
@Lazy
@Scope("prototype")
public class MyUtilityProgramContext extends RuntimeContext{

    protected MyUtilityProgramContext(@Qualifier("MyUtilityProgramContextConfiguration") Configuration configuration) {
        super(configuration);
    }

    @Override
    public void cleanUp() {
        // TODO implement clean-up of associated data structures if any
    }

    @Override
    protected void doReset() {
        // TODO implement reset of associated data structures if any
    }

}

```

Remarques

- La classe de contexte doit étendre une implémentation d'`Context`interface existante (`RuntimeContext`soit`JicsRuntimeContext`, soit améliorée `RuntimeContext` avec des éléments spécifiques à JICS).
- La convention de dénomination générale est ProgramNameContext.
- Vous devez le déclarer en tant que composant du prototype et utiliser l'annotation @Lazy.
- Le constructeur fait référence à la configuration associée en utilisant l'annotation @Qualifier pour cibler la classe de configuration appropriée.
- Si le programme utilitaire utilise des structures de données supplémentaires, celles-ci doivent être :
  - écrit et ajouté au *`base package`*`.business.model` package.
  - référencé dans le contexte. Examinez d'autres classes de contexte existantes pour voir comment référencer les classes de structures de données et adapter les méthodes de contexte (constructeur/nettoyage/réinitialisation) selon les besoins.

Maintenant qu'un contexte dédié est disponible, laissez le nouveau programme l'utiliser :

MyUtilityProgram.java

```java
import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
@Import({
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramConfiguration.class,
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class
})
public class MyUtilityProgram implements Program {

    @Autowired
    BeanFactory beanFactory;

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));

    private static final String programIdentifier = "MYUTILPG";

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public String getProgramMainIdentifier() {
        return programIdentifier;
    }


    @Override
    public Context getContext() {
        return ProgramContextStore.getOrCreate(
        getProgramMainIdentifier(),
        aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class,
        beanFactory);
    }

```

Remarques :

- La méthode GetContext doit être implémentée strictement comme indiqué, en utilisant une délégation à la getOrCreate méthode de la `ProgramContextStore` classe et le Spring `BeanFactory` câblé automatiquement. Un identifiant de programme unique est utilisé pour stocker le contexte du programme dans le `ProgramContextStore` ; cet identifiant est référencé comme étant « l'identifiant principal du programme ».
- La configuration associée et les classes de contexte doivent être référencées à l'aide de l'annotation `@Import` Spring.

Mise en œuvre de la logique métier

Lorsque le squelette du programme est terminé, implémentez la logique métier du nouveau programme utilitaire.

Faites-le dans la `run` méthode du programme. Cette méthode sera exécutée chaque fois que le programme est appelé, que ce soit par un autre programme ou par un script.

Bon codage !

Gestion de l'enregistrement du programme

Enfin, assurez-vous que le nouveau programme est correctement enregistré dans le`ProgramRegistry`. Si vous avez ajouté le nouveau programme au package qui contient déjà d'autres programmes, il n'y a plus rien à faire. Le nouveau programme est sélectionné et enregistré auprès de tous les programmes voisins au démarrage de l'application.

Si vous avez choisi un autre emplacement pour le programme, vous devez vous assurer qu'il est correctement enregistré au démarrage de Tomcat. Pour vous inspirer sur la manière de procéder, examinez la méthode d'initialisation des SpringbootLauncher classes générées dans le ou les projets de service (voirContenu du projet de service).

Consultez les journaux de démarrage de Tomcat. Chaque inscription au programme est enregistrée. Si votre programme est enregistré avec succès, vous trouverez l'entrée de journal correspondante.

Lorsque vous êtes certain que votre programme est correctement enregistré, vous pouvez commencer à itérer sur le codage logique métier.

Mappages de noms complets

Cette section contient des listes de mappages de noms complets de AWS Blu Age et de tiers à utiliser dans vos applications modernisées.

AWS Mappages de noms entièrement qualifiés Blu Age

Nom court	Nom entièrement qualifié
`CallBuilder`	`com.netfective.bluage.gapwalk.runtime.statements.CallBuilder`
`Configuration`	`com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration`

Nom court	Nom entièrement qualifié
ConfigurationBuilder	com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder
ExecutionController	com.netfective.bluage.gapwalk.rt.call.ExecutionController
ExecutionControllerImpl	com.netfective.bluage.gapwalk.rt.call.internal.ExecutionControllerImpl
File	com.netfective.bluage.gapwalk.rt.io.File
MainProgramRunner	com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
Program	com.netfective.bluage.gapwalk.rt.provider.Program
ProgramContextStore	com.netfective.bluage.gapwalk.rt.context.ProgramContextStore
ProgramRegistry	com.netfective.bluage.gapwalk.rt.provider.ProgramRegistry
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordEntity	com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity
RuntimeContext	com.netfective.bluage.gapwalk.rt.context.RuntimeContext

Nom court	Nom entièrement qualifié
SimpleStateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.SimpleStateMachineController
StateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineController
StateMachineRunner	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineRunner

Mappages de noms entièrement qualifiés par des tiers

Nom court	Nom entièrement qualifié
@Autowired	org.springframework.beans.factory.annotation.Autowired
@Bean	org.springframework.context.annotation.Bean
BeanFactory	org.springframework.beans.factory.BeanFactory
@Component	org.springframework.stereotype.Component
ConfigurableApplicationContext	org.springframework.context.ConfigurableApplicationContext
@Import	org.springframework.context.annotation.Import
@Lazy	org.springframework.context.annotation.Lazy

Que sont les simplificateurs de données dans AWS Blu Age

Sur les systèmes mainframe et de milieu de gamme (appelés systèmes « anciens » dans la rubrique suivante), les langages de programmation fréquemment utilisés tels que COBOL, PL/I ou RPG fournissent un accès de bas niveau à la mémoire. Cet accès se concentre sur la disposition de la mémoire accessible via des types natifs tels que zonée, compressée ou alphanumérique, éventuellement agrégée via des groupes ou des tableaux.

Un mélange d'accès à un élément de mémoire donné, à la fois via des champs dactylographiés et sous forme d'accès direct aux octets (mémoire brute), coexiste dans un programme donné. Par exemple, les programmes COBOL transmettent des arguments aux appelants sous forme d'ensembles d'octets contigus (LINKAGE) ou lisent et écrivent des données à partir de fichiers de la même manière (enregistrements), tout en interprétant ces plages de mémoire à l'aide de champs typés organisés dans des cahiers.

Ces combinaisons d'accès brut et structuré à la mémoire, le recours à une disposition précise de la mémoire au niveau des octets et les types existants, tels que zoné ou compressé, sont des fonctionnalités qui ne sont ni nativement ni facilement disponibles dans l'environnement de programmation Java.

Faisant partie de la solution AWS Blu Age pour moderniser les anciens programmes vers Java, la bibliothèque Data Simplifier fournit de telles constructions aux programmes Java modernisés et les expose d'une manière aussi familière que possible aux développeurs Java (getters/setters, tableaux d'octets, basés sur des classes). Il s'agit d'une dépendance essentielle du code Java modernisé généré à partir de tels programmes.

Pour des raisons de simplicité, la plupart des explications suivantes sont basées sur des constructions COBOL, mais vous pouvez utiliser la même API pour les deux PL1 et pour la modernisation de la mise en page des données RPG, car la plupart des concepts sont similaires.

Rubriques

- Classes principales
- Liaison des données et accès
- FQN des types Java discutés

Classes principales

Pour faciliter la lecture, ce document utilise les noms abrégés Java des interfaces et des classes de l'API AWS Blu Age. Pour de plus amples informations, veuillez consulter FQN des types Java discutés.

Représentation de la mémoire de bas niveau

Au niveau le plus bas, la mémoire (une plage contiguë d'octets accessible de manière rapide et aléatoire) est représentée par l'`Record`interface. Cette interface est essentiellement une abstraction d'un tableau d'octets de taille fixe. En tant que tel, il fournit des setters et des getters capables d'accéder aux octets sous-jacents ou de les modifier.

Représentation de données structurées

Pour représenter des données structurées, telles que « 01 éléments de données » ou « 01 copybooks », comme on le trouve dans COBOL DATA DIVISION, des sous-classes de la `RecordEntity` classe sont utilisées. Ils ne sont généralement pas écrits à la main, mais générés par les outils de modernisation de AWS Blu Age à partir des anciennes constructions correspondantes. Il est toujours utile de connaître leur structure principale et leur API afin de comprendre comment le code d'un programme modernisé les utilise. Dans le cas de COBOL, ce code est généré par Java à partir de leur DIVISION PROCEDURE.

Le code généré représente chaque « 01 élément de données » avec une `RecordEntity` sous-classe ; chaque champ élémentaire ou agrégat qui le compose est représenté sous la forme d'un champ Java privé, organisé sous forme d'arbre (chaque élément a un parent, à l'exception de la racine).

À des fins d'illustration, voici un exemple d'élément de données COBOL, suivi du code généré par AWS Blu Age correspondant qui le modernise :

```
01 TST2.
 02 FILLER PIC X(4).
 02 F1     PIC 9(2) VALUE 42.
 02 FILLER PIC X.
 02        PIC 9(3) VALUE 123.
 02 F2     PIC X VALUE 'A'.
```

```
public class Tst2 extends RecordEntity {

    private final Group root = new Group(getData()).named("TST2");
```

```
    private final Filler filler = new Filler(root,new AlphanumericType(4));
    private final Elementary f1 = new Elementary(root,new ZonedType(2, 0, false),new
BigDecimal("42")).named("F1");
    private final Filler filler1 = new Filler(root,new AlphanumericType(1));
    private final Filler filler2 = new Filler(root,new ZonedType(3, 0, false),new
BigDecimal("123"));
    private final Elementary f2 = new Elementary(root,new
AlphanumericType(1),"A").named("F2");



    /**
     * Instantiate a new Tst2 with a default record.
     * @param configuration the configuration
     */
    public Tst2(Configuration configuration) {
        super(configuration);
        setupRoot(root);
    }
    /**
     * Instantiate a new Tst2 bound to the provided record.
     * @param configuration the configuration
     * @param record the existing record to bind
     */
    public Tst2(Configuration configuration, RecordAdaptable record) {
        super(configuration);
        setupRoot(root, record);
    }

    /**
     * Gets the reference for attribute f1.
     * @return the f1 attribute reference
     */
    public ElementaryRangeReference getF1Reference() {
        return f1.getReference();
    }

    /* *
     * Getter for f1 attribute.
     * @return f1 attribute
     */
    public int getF1() {
        return f1.getValue();
    }
```

```
    /**
     * Setter for f1 attribute.
     * @param f1 the new value of f1
     */
    public void setF1(int f1) {
        this.f1.setValue(f1);
    }
    /**
     * Gets the reference for attribute f2.
     * @return the f2 attribute reference
     */
    public ElementaryRangeReference getF2Reference() {
        return f2.getReference();
    }

    /**
     * Getter for f2 attribute.
     * @return f2 attribute
     */
    public String getF2() {
        return f2.getValue();
    }

    /**
     * Setter for f2 attribute.
     * @param f2 the new value of f2
     */
    public void setF2(String f2) {
        this.f2.setValue(f2);
    }
}
```

Domaines élémentaires

Les champs de classe `Elementary` (ou`Filler`, lorsqu'ils ne sont pas nommés) représentent une « feuille » de l'ancienne structure de données. Ils sont associés à une plage contiguë d'octets sous-jacents (« plage ») et ont généralement un type (éventuellement paramétré) indiquant comment interpréter et modifier ces octets (en « décodant » et en « codant » respectivement une valeur depuis/vers un tableau d'octets).

Tous les types élémentaires sont des sous-classes de. `RangeType` Les types courants sont les suivants :

Type COBOL	Type de simplificateur de données
`PIC X(n)`	`AlphanumericType`
`PIC 9(n)`	`ZonedType`
`PIC 9(n) COMP-3`	`PackedType`
`PIC 9(n) COMP-5`	`BinaryType`

Champs agrégés

Les champs d'agrégation organisent la disposition en mémoire de leur contenu (autres agrégats ou champs élémentaires). Ils n'ont pas eux-mêmes de type élémentaire.

`Group`les champs représentent des champs contigus en mémoire. Chacun de leurs champs contenus est disposé dans le même ordre en mémoire, le premier champ étant `0` décalé par rapport à la position du champ de groupe en mémoire, le second champ étant décalé`0 + (size in bytes of first field)`, etc. Ils sont utilisés pour représenter des séquences de champs COBOL sous le même champ conteneur.

`Union`les champs représentent plusieurs champs accédant à la même mémoire. Chacun de leurs champs contenus est disposé de manière `0` décalée par rapport à la position du champ d'union en mémoire. Ils sont par exemple utilisés pour représenter la construction COBOL « REDEFINES » (les premiers enfants de l'Union étant l'élément de données redéfini, les seconds étant sa première redéfinition, etc.).

Les champs matriciels (sous-classes de`Repetition`) représentent la répétition, en mémoire, de la disposition de leur champ enfant (qu'il s'agisse d'un agrégat lui-même ou d'un élément élémentaire). Ils mettent en mémoire un certain nombre de mises en page pour enfants de ce type, chacune étant `index * (size in bytes of child)` décalée. Ils sont utilisés pour représenter les constructions COBOL « SURCIS ».

Primitives

Dans certains cas de modernisation, les « primitives » peuvent également être utilisées pour présenter des éléments de données « racines » indépendants. Leur utilisation est très similaire, `RecordEntity` mais ils ne proviennent pas de celui-ci et ne sont pas basés sur le code généré. Au lieu de cela, ils sont directement fournis par le moteur d'exécution AWS Blu Age en tant que

sous-classes de l'`Primitive`interface. Des exemples de ces cours fournis sont `Alphanumeric` ou`ZonedDecimal`.

Liaison des données et accès

L'association entre les données structurées et les données sous-jacentes peut se faire de différentes manières.

Une interface importante à cette fin est `RecordAdaptable` celle qui est utilisée pour obtenir une `Record` « vue inscriptible » des données `RecordAdaptable` sous-jacentes. Comme nous le verrons ci-dessous, plusieurs classes sont implémentées`RecordAdaptable`. Réciproquement, AWS Blu Age APIs et le code manipulant de la mémoire de bas niveau (tels que les arguments des programmes, les enregistrements d'E/S de fichiers, la zone de communication CICS, la mémoire allouée...) s'attendent souvent à un `RecordAdaptable` identifiant pour cette mémoire.

Dans le cas de la modernisation COBOL, la plupart des éléments de données sont associés à une mémoire qui sera corrigée pendant la durée d'exécution du programme correspondant. À cette fin, les `RecordEntity` sous-classes sont instanciées une seule fois dans un objet parent généré (le programme Context) et se chargeront d'instancier leur sous-jacent`Record`, en fonction de la taille en octets. `RecordEntity`

Dans d'autres cas COBOL, tels que l'association d'éléments LINKAGE à des arguments de programme ou la modernisation de la construction SET ADDRESS OF, une `RecordEntity` instance doit être associée à un paramètre fourni. `RecordAdaptable` Pour cela, deux mécanismes existent :

- si l'`RecordEntity`instance existe déjà, la `RecordEntity.bind(RecordAdaptable)` méthode (héritée de`Bindable`) peut être utilisée pour que cette instance « pointe » vers celle-ci`RecordAdaptable`. Tout getter ou setter appelé sur le `RecordEntity` sera ensuite sauvegardé (octets en lecture ou en écriture) par les octets sous-jacents`RecordAdaptable`.
- si le `RecordEntity` doit être instancié, un constructeur généré acceptant a `RecordAdaptable` est disponible.

Inversement, les données `Record` actuellement liées aux données structurées sont accessibles. Pour cela`RecordAdaptable`, `RecordEntity` implémente donc `getRecord()` n'importe quelle instance de ce type.

Enfin, de nombreux verbes COBOL ou CICS nécessitent l'accès à un seul champ, à des fins de lecture ou d'écriture. La `RangeReference` classe est utilisée pour représenter un tel

accès. Ses instances peuvent être obtenues à partir de `getXXXReference()` méthodes `RecordEntity` générées (`XXX`étant le champ accédé) et transmises aux méthodes d'exécution. `RangeReference`est généralement utilisé pour accéder à l'ensemble `RecordEntity` ou`Group`, tandis que sa sous-classe `ElementaryRangeReference` représente les accès aux `Elementary` champs.

Notez que la plupart des observations ci-dessus s'appliquent aux `Primitive` sous-classes, car elles visent à implémenter un comportement similaire à celui fourni par le moteur d'exécution AWS Blu Age (au lieu du code généré). `RecordEntity` À cette fin, toutes les sous-classes d'`Primitive`implémentent `ElementaryRangeReference` et `RecordAdaptable` les `Bindable` interfaces de manière à être utilisables à la fois à la place des `RecordEntity` sous-classes et des champs élémentaires.

FQN des types Java discutés

Le tableau suivant indique les noms complets des types Java décrits dans cette section.

Nom court	Nom entièrement qualifié
`Alphanumeric`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.Alphanumeric`
`AlphanumericType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType`
`BinaryType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.BinaryType`
`Bindable`	`com.netfective.bluage.gapwalk.datasimplifier.data.Bindable`
`Elementary`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary`

Nom court	Nom entièrement qualifié
ElementaryRangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference
Filler	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Filler
Group	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group
PackedType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.PackedType
Primitive	com.netfective.bluage.gapwalk.datasimplifier.elementary.Primitive
RangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference
RangeType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.RangeType
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordAdaptable	com.netfective.bluage.gapwalk.datasimplifier.data.RecordAdaptable

Nom court	Nom entièrement qualifié
`RecordEntity`	`com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity`
`Repetition`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Repetition`
`Union`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Union`
`ZonedDecimal`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.ZonedDecimal`
`ZonedType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType`

AWS Fard à joues Blue Age

Sur les systèmes mainframe (appelés « anciens » dans la rubrique suivante), les données commerciales sont souvent stockées à l'aide de la méthode VSAM (Virtual Storage Access Method). Les données sont stockées dans des « enregistrements » (tableaux d'octets) appartenant à un « ensemble de données ».

Il existe quatre organisations chargées des ensembles de données :

- KSDS : ensembles de données séquencés par touches : les enregistrements sont indexés par une clé primaire (aucune clé dupliquée n'est autorisée) et, éventuellement, par des clés « alternatives » supplémentaires. Toutes les valeurs de clé sont des sous-ensembles du tableau d'octets d'enregistrement, chaque clé étant définie par :
  - un décalage (basé sur 0, 0 étant le début du contenu du tableau d'octets d'enregistrement, mesuré en octets)

  - une longueur (exprimée en octets)
  - s'il tolère ou non les valeurs dupliquées
- ESDS : ensembles de données séquencés par entrée : les enregistrements sont accessibles principalement de manière séquentielle (en fonction de leur ordre d'insertion dans l'ensemble de données) mais peuvent être consultés à l'aide de clés alternatives supplémentaires ;
- RRDS : ensembles de données relatifs aux enregistrements : les enregistrements sont accessibles par « sauts », en utilisant des numéros d'enregistrement relatifs ; les sauts peuvent être effectués en avant ou en arrière ;
- LDS : ensembles de données linéaires : aucun enregistrement, simplement un flux d'octets, organisé en pages. Principalement utilisé à des fins internes sur les anciennes plateformes.

Lors de la modernisation des applications existantes, à l'aide de l'approche de refactorisation AWS Blu Age, les applications modernisées ne sont plus destinées à accéder aux données stockées par VSAM, tout en préservant la logique d'accès aux données. Le composant Blusam est la solution : il permet d'importer des données à partir d'anciens ensembles de données VSAM exportés, fournit une API permettant à l'application modernisée de les manipuler, ainsi qu'une application Web d'administration dédiée. Voir the section called “Console d'administration Blusam”.

> **Note**
>
> Blusam ne prend en charge que KSDS, ESDS et RRDS.

L'API Blusam permet de préserver la logique d'accès aux données (lectures séquentielles, aléatoires et relatives ; insertion, mise à jour et suppression d'enregistrements), tandis que l'architecture des composants, qui repose sur une combinaison de stratégies de mise en cache et de stockage basé sur des SGBDR, permet des opérations d'E/S à haut débit avec des ressources limitées.

Infrastructures Blusam

Blusam s'appuie sur le RDBMS PostgreSQL pour le stockage des ensembles de données, à la fois pour les données d'enregistrements brutes et les index de clés (le cas échéant). L'option préférée consiste à utiliser le moteur compatible avec Amazon Aurora PostgreSQL. Les exemples et illustrations présentés dans cette rubrique sont basés sur ce moteur.

> **Note**
>
> Au démarrage du serveur, le moteur d'exécution Blusam vérifie la présence de certaines tables techniques obligatoires et les crée si elles sont introuvables. Par conséquent, le rôle utilisé dans la configuration pour accéder à la base de données Blusam doit être autorisé à créer, mettre à jour et supprimer les tables de base de données (à la fois les lignes et les définitions des tables elles-mêmes). Pour plus d'informations sur la désactivation de Blusam, consultez. the section called "Configuration de Blusam"

mise en cache

Outre le stockage lui-même, Blusam fonctionne plus rapidement lorsqu'il est associé à une implémentation de cache.

Deux moteurs de cache sont actuellement pris en charge, EhCache et Redis, chacun ayant son propre cas d'utilisation :

- EhCache : cache local volatil intégré autonome
  - NON éligible au déploiement de l'environnement géré AWS Mainframe Modernization.
  - Généralement utilisé lorsqu'un seul nœud, tel qu'un seul serveur Apache Tomcat, est utilisé pour exécuter les applications modernisées. Par exemple, le nœud peut être dédié à l'hébergement de tâches par lots.
  - Volatile : l'instance de EhCache cache est volatile ; son contenu sera perdu lors de l'arrêt du serveur.
  - Embarqué : Le serveur EhCache et le serveur partagent le même espace mémoire JVM (à prendre en compte lors de la définition des spécifications de la machine d'hébergement).
- Redis : cache persistant partagé
  - Éligibilité au déploiement de l'environnement géré de modernisation du AWS mainframe.
  - Généralement utilisé dans des situations impliquant plusieurs nœuds, en particulier lorsque plusieurs serveurs se trouvent derrière un équilibreur de charge. Le contenu du cache est partagé entre tous les nœuds.
  - Le Redis est persistant et n'est pas lié au cycle de vie des nœuds. Il fonctionne sur sa propre machine ou service dédié (par exemple, Amazon ElastiCache). Le cache est distant de tous les nœuds.

Verrouillage

Pour gérer l'accès simultané aux ensembles de données et aux enregistrements, Blusam s'appuie sur un système de verrouillage configurable. Le verrouillage peut être appliqué aux deux niveaux : ensembles de données et enregistrements :

- Le verrouillage d'un ensemble de données à des fins d'écriture empêchera tous les autres clients d'effectuer des opérations d'écriture sur celui-ci, à n'importe quel niveau (ensemble de données ou enregistrement).
- Le verrouillage au niveau de l'enregistrement pour l'écriture empêchera les autres clients d'effectuer des opérations d'écriture uniquement sur l'enregistrement donné.

La configuration du système de verrouillage Blusam doit être effectuée conformément à la configuration du cache :

- S'il EhCache est choisi comme implémentation du cache, aucune autre configuration de verrouillage n'est requise car le système de verrouillage en mémoire par défaut doit être utilisé.
- Si Redis est choisi comme implémentation du cache, une configuration de verrouillage basée sur Redis est requise pour permettre un accès simultané à partir de plusieurs nœuds. Le cache Redis utilisé pour les verrous ne doit pas nécessairement être le même que celui utilisé pour les ensembles de données. Pour plus d'informations sur la configuration d'un système de verrouillage basé sur Redis, consultez. the section called “Configuration de Blusam”

Les caractéristiques intrinsèques de Blusam et la migration des données depuis l'ancienne version

Stockage des ensembles de données : enregistrements et index

Chaque ensemble de données existant, une fois importé dans Blusam, sera stocké dans une table dédiée ; chaque ligne de la table représente un enregistrement, en utilisant deux colonnes :

- La colonne d'identification numérique, de type grand entier, qui est la clé primaire de la table, est utilisée pour stocker l'adresse d'octet relative (RBA) de l'enregistrement. Le RBA représente le décalage en octets par rapport au début de l'ensemble de données et commence à 0.
- La colonne d'enregistrement du tableau d'octets, utilisée pour stocker le contenu de l'enregistrement brut.

Voir par exemple le contenu d'un ensemble de données KSDS utilisé dans l' CardDemo application :

```
1  SELECT * FROM public.aws_m2_carddemo_acctdata_vsam_ksds
2  ORDER BY id ASC
```

Data output | Messages | Notifications

	id [PK] bigint	record bytea
1	0	[binary data]
2	300	[binary data]
3	600	[binary data]
4	900	[binary data]
5	1200	[binary data]
6	1500	[binary data]
7	1800	[binary data]
8	2100	[binary data]
9	2400	[binary data]
10	2700	[binary data]
11	3000	[binary data]
12	3300	[binary data]
13	3600	[binary data]

- Cet ensemble de données particulier comporte des enregistrements de longueur fixe, la longueur étant de 300 octets (la collection d'identifiants étant donc des multiples de 300).
- Par défaut, l'outil pgAdmin utilisé pour interroger les bases de données PostgreSQL n'affiche pas le contenu des colonnes d'un tableau d'octets, mais affiche plutôt une étiquette [données binaires].
- Le contenu brut de l'enregistrement correspond à l'ensemble de données brutes exporté à partir de l'ancien, sans aucune conversion. En particulier, aucune conversion de jeu de caractères n'a lieu ; cela implique que les parties alphanumériques de l'enregistrement devront être décodées par des applications modernisées utilisant le jeu de caractères existant, probablement une variante EBCDIC.

En ce qui concerne les métadonnées des ensembles de données et les index des clés : chaque ensemble de données est associé à deux lignes dans la table nommée`metadata`. Il s'agit de la

convention de dénomination par défaut. Pour savoir comment le personnaliser, voirthe section called "Configuration de Blusam".

	name [PK] text	metadata oid
1	AWS_M2_CARDDEMO_ACCTDATA_VSAM_KSDS	66320
2	AWS_M2_CARDDEMO_ACCTDATA_VSAM_KSDS__internal	66321

- La première ligne contient le nom de l'ensemble de données comme valeur de la colonne de nom. La colonne de métadonnées est une colonne binaire qui contient une sérialisation binaire des métadonnées générales de l'ensemble de données donné. Pour plus d'informations, consultez the section called "Attributs de métadonnées généraux des ensembles de données".
- La deuxième ligne contient le nom de l'ensemble de données avec le suffixe `__internal'` comme valeur de la colonne de nom. Le contenu binaire de la colonne de métadonnées dépend du « poids » de l'ensemble de données.
  - Pour les ensembles de données de petite ou moyenne taille, le contenu est une sérialisation compressée de :
    - définition des clés utilisées par l'ensemble de données ; définition de la clé primaire (pour KSDS) et définitions des clés alternatives, le cas échéant (pour KSDS/ ESDS)
    - les index clés, le cas échéant (KSDS/ ESDS avec définitions de clés alternatives) : utilisés pour la navigation indexée des enregistrements ; l'index clé fait correspondre une valeur clé au RBA d'un enregistrement ;
    - carte de longueur des enregistrements : utilisée pour la navigation séquentielle/relative des enregistrements ;
  - Pour les ensembles de données de grande ou très grande taille, le contenu est une sérialisation compressée de :
    - définition des clés utilisées par l'ensemble de données ; définition de la clé primaire (pour KSDS) et définitions des clés alternatives, le cas échéant (pour KSDS/ ESDS)

De plus, les index des ensembles de données de grande ou très grande taille (le cas échéant) sont stockés à l'aide d'un mécanisme de pagination ; les sérialisations binaires des pages d'index sont stockées sous forme de lignes d'une table dédiée (une table par clé d'ensemble de données). Chaque page d'index est stockée dans une ligne comportant les colonnes suivantes :

- id : identifiant technique de la page d'index (clé primaire numérique) ;
- firstkey : valeur binaire de la première valeur clé (la plus basse) stockée dans la page des index ;
- lastkey : valeur binaire de la dernière valeur clé (la plus élevée) stockée dans la page d'index ;

- métadonnées : sérialisation compressée binaire de la page d'index (mappage des valeurs clés aux enregistrements RBAs).

	id [PK] bigint	firstkey bytea	lastkey bytea	metadata oid
1	1	[binary data]	[binary da...	6458928
2	2	[binary data]	[binary da...	6458929
3	3	[binary data]	[binary da...	6458930
4	4	[binary data]	[binary da...	6458931
5	5	[binary data]	[binary da...	6458932
6	6	[binary data]	[binary da...	6458933
7	7	[binary data]	[binary da...	6458934
8	8	[binary data]	[binary da...	6458935
9	9	[binary data]	[binary da...	6458936

Le nom de la table est une concaténation du nom de l'ensemble de données et du nom interne de la clé, qui contient des informations sur la clé, telles que le décalage de touche, si la clé accepte les doublons (défini sur true pour autoriser les doublons) et la longueur de la clé. Par exemple, considérez un ensemble de données nommé « AWS_LARGE _KSDS » qui possède les deux clés définies suivantes :

- clé primaire [décalage : 0, doublons : faux, longueur : 18]
- clé alternative [décalage : 3, doublons : vrai, longueur : 6]

Dans ce cas, les tables suivantes stockent les index relatifs aux deux clés.

Optimisation du débit d'E/S à l'aide du mécanisme d'écriture différée

Pour optimiser les performances des opérations d'insertion/de mise à jour/de suppression, le moteur Blusam s'appuie sur un mécanisme d'écriture secondaire configurable. Le mécanisme repose sur un pool de threads dédiés qui traitent les opérations de persistance à l'aide de requêtes de mise à jour groupées, afin de maximiser le débit d'E/S vers le stockage Blusam.

Le moteur Blusam collecte toutes les opérations de mise à jour effectuées sur les enregistrements par les applications et crée des lots d'enregistrements qui sont envoyés pour traitement aux threads

dédiés. Les lots sont ensuite conservés dans le stockage Blusam, à l'aide de requêtes de mise à jour groupées, en évitant le recours à des opérations de persistance atomique et en garantissant la meilleure utilisation possible de la bande passante du réseau.

Le mécanisme utilise un délai configurable (une seconde par défaut) et une taille de lot configurable (10 000 enregistrements par défaut). Les requêtes de persistance du build sont exécutées dès que la première des deux conditions suivantes est remplie :

- Le délai configuré est expiré et le lot n'est pas vide
- Le nombre d'enregistrements du lot à traiter atteint la limite configurée

Pour savoir comment configurer le mécanisme d'écriture différée, consultez. the section called "Propriétés facultatives"

Choisir le bon plan de stockage

Comme indiqué dans la section précédente, la manière dont les ensembles de données sont stockés dépend de leur « poids ». Mais qu'est-ce qui est considéré comme petit, moyen ou grand pour un ensemble de données ? Quand choisir la stratégie de stockage paginée plutôt que la stratégie classique ?

La réponse à cette question dépend des éléments suivants.

- La quantité de mémoire disponible sur chacun des serveurs hébergeant les applications modernisées qui utiliseront ces ensembles de données.
- La quantité de mémoire disponible sur l'infrastructure de cache (le cas échéant).

Lorsque vous utilisez un système de stockage d'index non paginé, les collections complètes d'index clés et de tailles d'enregistrements seront chargées dans la mémoire du serveur au moment de l'ouverture du jeu de données, pour chaque ensemble de données. En outre, si la mise en cache est impliquée, tous les enregistrements des ensembles de données peuvent être préchargés dans le cache selon l'approche normale, ce qui peut entraîner un épuisement des ressources de mémoire du côté de l'infrastructure du cache.

En fonction du nombre de clés définies, de la longueur des valeurs clés, du nombre d'enregistrements et du nombre d'ensembles de données ouverts en même temps, la quantité de mémoire consommée peut être évaluée approximativement pour les cas d'utilisation connus donnés.

Pour en savoir plus, consultez la section the section called "Estimation de l'empreinte mémoire pour un ensemble de données donné".

Migration de Blusam

Une fois que le schéma de stockage approprié a été sélectionné pour un ensemble de données donné, le stockage blusam doit être rempli en migrant les anciens ensembles de données.

Pour ce faire, il faut utiliser des exportations binaires brutes des anciens ensembles de données, sans qu'aucune conversion de jeu de caractères ne soit utilisée pendant le processus d'exportation. Lorsque vous transférez des exportations d'ensembles de données depuis l'ancien système, veillez à ne pas corrompre le format binaire. Par exemple, appliquez le mode binaire lors de l'utilisation du protocole FTP.

Les exportations binaires brutes contiennent uniquement les enregistrements. Le mécanisme d'importation n'a pas besoin que la keys/indexes exports as all keys/indexes zone soit recalculée à la volée par le mécanisme d'importation.

Une fois qu'une exportation binaire d'un ensemble de données est disponible, plusieurs options existent pour la migrer vers Blusam :

À propos de l'environnement géré de modernisation du AWS mainframe :

- Importez des ensembles de données à l'aide de la fonctionnalité dédiée. Voir the section called "Importer des ensembles de données pour les applications".

or

- Utilisez la fonction d'importation groupée d'ensembles de données. Consultez the section called "Référence de définition de l'ensemble de données" et the section called "Exemple de format de demande d'ensemble de données pour VSAM".

or

- Utilisez un script groovy pour importer des ensembles de données à l'aide de services de chargement dédiés.

> **Note**
>
> Pour le moment, l'importation de LargeKSD et LargeESD dans les environnements gérés Mainframe Modernization n'est possible qu'à l'aide de scripts groovy.

À propos de AWS Blu Age Runtime sur Amazon EC2 :

- Importez un ensemble de données à l'aide duthe section called "Console d'administration Blusam".

or

- Utilisez un script groovy pour importer des ensembles de données à l'aide de services de chargement dédiés.

Importer des ensembles de données à l'aide de scripts Groovy

Cette section vous aidera à écrire des scripts géniaux pour importer des ensembles de données existants dans Blusam.

Cela commence par quelques importations obligatoires :

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList; //used for alternate keys if any
```

Ensuite, pour chaque ensemble de données à importer, le code est construit sur le modèle donné :

1. créer ou effacer un objet cartographique
2. remplissez la carte avec les propriétés requises (cela varie en fonction du type d'ensemble de données - voir ci-dessous pour plus de détails)
3. récupérer le service de chargement approprié à utiliser pour le type d'ensemble de données dans le registre des services
4. exécuter le service en utilisant la carte comme argument

Cinq implémentations de service peuvent être extraites du registre des services à l'aide des identifiants suivants :

- `"BluesamKSDSFileLoader"`: pour les KDS de petite et moyenne taille
- `"BluesamESDSFileLoader"`pour les systèmes ESDS de petite et moyenne taille
- `"BluesamRRDSFileLoader"`: pour RDS
- `"BluesamLargeKSDSFileLoader"`: pour les grands KDS
- `"BluesamLargeESDSFileLoader"`: pour les disques SSD de grande taille

Le choix de la version standard ou de la version étendue du service pour KSDS/ESDS dépend de la taille des ensembles de données et de la stratégie de stockage que vous souhaitez appliquer. Pour savoir comment choisir la bonne stratégie de stockage, voirthe section called "Choisir le bon plan de stockage".

Pour pouvoir importer correctement l'ensemble de données dans Blusam, les propriétés appropriées doivent être fournies au service de chargement.

Propriétés communes :

- Obligatoire (pour tous les types d'ensembles de données)
  - « BlueSamManager » : la valeur attendue est `applicationContext.getBean(BluesamManager.class)`
  - « DataSetName » : nom de l'ensemble de données, sous forme de chaîne
  - «inFilePath" : chemin vers l'ancienne exportation de l'ensemble de données, sous forme de chaîne
  - « recordLength » : la longueur d'enregistrement fixe ou 0 pour un ensemble de données de longueur d'enregistrement variable, sous forme d'entier
- Facultatif
  - Non pris en charge pour les grands ensembles de données :
    - « IsAppend » : indicateur booléen indiquant que l'importation s'effectue en mode ajout (ajout d'enregistrements à un ensemble de données blusam existant).
    - « useCompression » : un indicateur booléen, indiquant que la compression sera utilisée pour stocker les métadonnées.
  - Uniquement pour les grands ensembles de données :
    - « indexingPageSizeInMb" : la taille en mégaoctets de chaque page d'index, pour chacune des clés de l'ensemble de données, sous la forme d'un entier strictement positif

Propriétés dépendantes du type de jeu de données :

- KSDS/Grand KDS :
  - mandatory
    - « PrimaryKey » : la définition de la clé primaire, à l'aide d'un appel au `com.netfective.bluage.gapwalk.bluesam.metadata.Key` constructeur.
  - optionnel :
    - « AlternateKeys » : une liste (`java.util.List`) de définitions de clés alternatives, construite à l'aide d'appels de `com.netfective.bluage.gapwalk.bluesam.metadata.Key` constructeurs.
- SDS ou grands disques SSD :
  - optionnel :
    - « AlternateKeys » : une liste (`java.util.List`) de définitions de clés alternatives, construite à l'aide d'appels de `com.netfective.bluage.gapwalk.bluesam.metadata.Key` constructeurs.
- ROUGES :
  - aucun.

Les principaux appels du constructeur sont les suivants :

- `new Key(int offset, int length)`: crée un objet Key, avec des attributs clés donnés (décalage et longueur) et aucun doublon n'est autorisé. Cette variante doit être utilisée pour définir une clé primaire.
- `new Key(boolean allowDuplicates, int offset, int length)`: crée un objet Key, avec des attributs clés donnés (décalage et longueur) et duplique le drapeau d'autorisation.

Les exemples Groovy suivants illustrent différents scénarios de chargement.

Chargement d'un KSDS de grande taille, avec deux clés alternatives :

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList;

// Loading a large KSDS into Blusam
```

```
def map = [:]
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", "largeKsdsSample");
map.put("inFilePath", "/work/samples/largeKsdsSampleExport");
map.put("recordLength", 49);
map.put("primaryKey", new Key(0, 18));
ArrayList altKeys = [new Key(true, 10, 8), new Key(false, 0, 9)]
map.put("alternateKeys", altKeys);
map.put("indexingPageSizeInMb", 25);
def service = ServiceRegistry.getService("BluesamLargeKSDSFileLoader");
service.runService(map);
```

Chargement d'un ESDS de longueur d'enregistrement variable, sans clé alternative :

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry

// Loading an ESDS into Blusam
def map = [:]
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", "esdsSample");
map.put("inFilePath", "/work/samples/esdsSampleExport");
map.put("recordLength", 0);
def service = ServiceRegistry.getService("BluesamESDSFileLoader");
service.runService(map);
```

Les ensembles de données de longueur d'enregistrement variable exportés contiendront les informations RDW (Record Descriptor Word) obligatoires pour permettre le fractionnement des enregistrements au moment de la lecture.

Chargement d'un RRDS de longueur d'enregistrement fixe :

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry

// Loading a RRDS into Blusam
def map = [:]
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", "rrdsSample");
map.put("inFilePath", "/work/samples/rrdsSampleExport");
```

```
map.put("recordLength", 180);
def service = ServiceRegistry.getService("BluesamRRDSFileLoader");
service.runService(map);
```

Chargement d'ensembles de données en mode multi-schéma :

Mode multi-schéma : dans certains systèmes existants, les fichiers VSAM sont organisés en ensembles de fichiers, ce qui permet aux programmes d'accéder aux données des partitions spécifiées et de les modifier. Les systèmes modernes traitent chaque ensemble de fichiers comme un schéma, ce qui permet un partitionnement des données et un contrôle d'accès similaires.

Pour activer le mode multi-schémas dans le `application-main.yml` fichier, reportez-vous àthe section called "Configuration de Blusam". Dans ce mode, les ensembles de données peuvent être chargés dans un schéma spécifique à l'aide d'un contexte partagé, qui est un registre en mémoire pour les informations d'exécution. Pour charger un ensemble de données dans un schéma spécifique, préfixez le nom du jeu de données par le nom du schéma correspondant.

Chargement d'un fichier KSDS dans un schéma spécifique pour le mode multi-schémas :

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList;
import com.netfective.bluage.gapwalk.rt.shared.SharedContext;

// Loading a KSDS into Blusam
def map = [:]
String schema = "schema1";
String datasetName = schema+"|"+"ksdsSample";
SharedContext.get().setCurrentBlusamSchema(schema);
schema = SharedContext.get().getCurrentBlusamSchema();
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", datasetName);
map.put("inFilePath", "/work/samples/ksdsSampleExport");
map.put("recordLength", 49);
map.put("primaryKey", new Key(0, 18));
map.put("indexingPageSizeInMb", 25);
def service = ServiceRegistry.getService("BluesamKSDSFileLoader");
service.runService(map);
```

Chargement d'un fichier KSDS volumineux dans un schéma spécifique pour le mode multi-schémas :

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList;
import com.netfective.bluage.gapwalk.rt.shared.SharedContext;

// Loading a Large KSDS into Blusam
def map = [:]
String schema = "schema1";
String datasetName = schema+"|"+"largeKsdsSample";
SharedContext.get().setCurrentBlusamSchema(schema);
schema = SharedContext.get().getCurrentBlusamSchema();
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", datasetName);
map.put("inFilePath", "/work/samples/LargeKsdsSampleExport");
map.put("recordLength", 49);
map.put("primaryKey", new Key(0, 18));
map.put("indexingPageSizeInMb", 25);
def service = ServiceRegistry.getService("BluesamLargeKSDSFileLoader");
service.runService(map);
```

En outre, une entrée de configuration (à définir dans le fichier de `application-main.yml` configuration) peut être utilisée pour affiner le processus d'importation :

- `bluesam.fileLoading.commitInterval`: un entier strictement positif, définissant l'intervalle de validation pour le mécanisme ESDS/KSDS/RRDS d'importation normal. Ne s'applique pas aux importations de grands ensembles de données. La valeur par défaut est 100 000.

Configuration de Blusam

La configuration de Blusam se fait dans le fichier de `application-main.yml` configuration (ou dans le fichier de `application-bac.yml` configuration pour le déploiement autonome de l'application Blusam Administration Console -- BAC --).

Blusam doit être configuré sur deux points :

- Configuration du stockage et de l'accès aux caches Blusam
- Configuration du moteur Blusam

Configuration du stockage et de l'accès aux caches Blusam

Pour plus d'informations sur la façon de configurer l'accès au stockage et aux caches Blusam à l'aide de gestionnaires de secrets ou de sources de données, consultez. the section called “AWS Configuration de Blu Age Runtime”

> **Note**
>
> En ce qui concerne l'accès au stockage Blusam, les informations d'identification utilisées indiqueront un rôle de connexion, avec les privilèges correspondants. Pour que le moteur Blusam puisse fonctionner comme prévu, le rôle de connexion doit disposer des privilèges suivants :

- se connecter à la base de données
- créer/supprimer/modifier/tronquer des tables et des vues
- sélectionner/insérer/supprimer/mettre à jour les lignes dans les tables et les vues
- exécuter des fonctions ou des procédures

Configuration du moteur Blusam

Désactiver le support Blusam

Tout d'abord, mentionnons qu'il est possible de désactiver complètement le support de Blusam, en définissant la `bluesam.disabled` propriété sur. `true` Un message d'information sera affiché dans les journaux du serveur au démarrage de l'application pour rappeler la désactivation de Blusam :

```
BLUESAM is disabled. No operations allowed.
```

Aucune autre configuration concernant Blusam n'est requise dans ce cas et toute tentative d'utilisation des fonctionnalités associées à Blusam (par programmation ou par le biais d'appels REST) déclenchera une exécution du code Java, avec un `UnsupportedOperationException` message d'explication pertinent concernant la désactivation de Blusam.

Propriétés du moteur Blusam

Les propriétés de configuration du moteur Blusam sont regroupées sous le préfixe bluesam key :

Propriétés obligatoires

- `cache`: à évaluer avec l'implémentation de cache choisie. Les valeurs valides sont :
  - `ehcache`: Pour une utilisation locale de l'ehcache intégré. Consultez les restrictions relatives aux cas d'utilisation connexes ci-dessus.
  - `redis`: Pour l'utilisation partagée du cache Redis à distance. Il s'agit de l'option préférée pour le cas d'utilisation gérée de la modernisation du AWS mainframe.
  - `none`: pour désactiver la mise en cache du stockage
- `persistence`: à évaluer avec pgsql (moteur PostgreSQL : version minimale 10.0 — version recommandée >=14.0
- référence de source de données : `<persistence engine>.dataSource` pointera vers la définition de la source de données pour la connexion au stockage Blusam, définie ailleurs dans le fichier de configuration. Il est communément nommé`bluesamDs`.

> **Note**
>
> Chaque fois que Redis est utilisé comme mécanisme de cache, que ce soit pour les données ou pour les verrous (voir ci-dessous), l'accès aux instances Redis doit être configuré. Pour plus d'informations, consultez the section called "Propriétés du cache Redis disponibles".

Propriétés facultatives

Blusam Locks : les propriétés sont préfixées par `locks`

- `cache`: la seule valeur utilisable est `redis` de spécifier que le mécanisme de verrouillage basé sur Redis sera utilisé (à utiliser lorsque le cache de stockage Blusam est également basé sur Redis). Si la propriété est absente ou n'est pas définie sur`redis`, le mécanisme de verrouillage en mémoire par défaut sera utilisé à la place.
- `lockTimeOut`: une valeur entière longue positive, indiquant le délai d'attente exprimé en millisecondes avant qu'une tentative de verrouillage d'un élément déjà verrouillé ne soit marquée comme ayant échoué. La valeur par défaut est. `500`
- `locksDeadTime`: une valeur entière longue positive, représentant la durée maximale, exprimée en millisecondes, pendant laquelle une application peut maintenir un verrou. Les verrous sont

automatiquement marqués comme expirés et relâchés une fois ce délai écoulé. La valeur par défaut est ; `1000`

- `locksCheck`: une chaîne, utilisée pour définir la stratégie de vérification du verrouillage utilisée par le gestionnaire de verrous Blusam actuel, concernant la suppression des verrous expirés. À sélectionner parmi les valeurs suivantes :
  - `off`: aucun contrôle n'est effectué. Déconseillé, car des blocages peuvent se produire.
  - `reboot`: les vérifications sont effectuées au redémarrage ou au démarrage de l'application. Tous les verrous expirés sont alors libérés. Il s'agit de l'option par défaut.
  - `timeout`: les vérifications sont effectuées au redémarrage ou au démarrage de l'application, ou lorsqu'un délai expire lors d'une tentative de verrouillage d'un ensemble de données. Les verrous expirés sont immédiatement libérés.

Mécanisme d'écriture secondaire : les propriétés sont préfixées par la clé : `write-behind`

- `enabled`: `true` (valeur par défaut et recommandée) ou `false` pour activer ou désactiver le mécanisme d'écriture différée. La désactivation du mécanisme aura un impact important sur les performances d'écriture et est déconseillée.
- `maxDelay`: durée maximale pendant laquelle les threads doivent être déclenchés. La valeur par défaut est `"1s"` (une seconde). Conserver la valeur par défaut est généralement une bonne idée, sauf si des conditions spécifiques nécessitent un ajustement de cette valeur. Dans tous les cas, la valeur doit rester faible (moins de 3 secondes). Le format de la chaîne de retard est le suivant : `<integer value><optional whitespace><time unit>` où `<time unit>` doit être choisi parmi les valeurs suivantes :
  - `"ns"`: nanosecondes
  - `"µs"`: microsecondes
  - `"ms"`: millisecondes
  - `"s"`: secondes
- `threads`: le nombre de threads dédiés à l'écriture différée. La valeur par défaut est5. Vous devez ajuster cette valeur en fonction de la puissance de calcul de l'hôte exécutant le moteur Blusam. Il n'est pas pertinent d'utiliser une valeur beaucoup plus élevée dans l'espoir d'améliorer les performances, car le facteur limitant sera la capacité des SGBDR de stockage à traiter de nombreuses requêtes par lots simultanées. Les valeurs recommandées sont généralement comprises entre 4 et 8.

- `batchSize`: un entier positif représentant le nombre maximal d'enregistrements d'un lot qui seront envoyés pour traitement en vrac vers un fil de discussion. Sa valeur doit être comprise entre 1 et 32 767. La valeur par défaut est. `10000` L'utilisation 1 comme valeur va à l'encontre de l'objectif du mécanisme qui est d'éviter d'utiliser des requêtes de mise à jour atomiques ; la valeur minimale appropriée à utiliser est d'environ`1000`.

EhCache Réglage fin intégré : les propriétés sont préfixées par la clé : `ehcache`

- `resource-pool`:
  - `size`: taille de mémoire allouée pour le cache intégré, exprimée sous forme de chaîne. La valeur par défaut est `"1024MB"` (1 gigaoctet). À ajuster en fonction de la mémoire disponible de la machine hébergeant le moteur Blusam et de la taille des ensembles de données utilisés par l'application. Le format de la chaîne de taille est le suivant : `<integer value><optional whitespace><memory unit>` où `<memory-unit>` doit être choisi parmi les valeurs suivantes :
    - `B` : octets
    - `KB`: kilo-octets
    - `MB`: mégaoctets
    - `GB`: gigaoctets
    - `TB`: téraoctets
  - `heap`: `true` ou`false`, pour indiquer si le cache consommera ou non de la mémoire tampon de la JVM. La valeur par défaut est `true` (option la plus rapide en termes de performances du cache, mais le stockage en cache consomme de la mémoire RAM sur le tas de la JVM). Si vous définissez cette propriété sur, `false` vous passez à la mémoire Off-Heap, qui sera plus lente en raison des échanges nécessaires avec le tas de mémoire JVM.
- `timeToLiveMillis`: durée (en millisecondes) pendant laquelle une entrée de cache reste dans le cache avant d'être considérée comme expirée et supprimée. Si cette propriété n'est pas spécifiée, les entrées du cache n'expireront pas automatiquement par défaut.

Propriétés de configuration multi-schémas

- `multiSchema`: false (valeur par défaut) ou true, pour désactiver ou activer le mode multi-schémas pour Blusam - Disponible à partir de la version 4.4.0.
- `pgsql`:

- `schemas`: liste des noms de schéma que l'application utilisera en mode multi-schémas pour Blusam.
- `fallbackSchema`: nom du schéma de remplacement à utiliser en mode multi-schémas. Si aucun ensemble de données n'est trouvé dans le contexte du schéma actuel, ce schéma sera utilisé pour les opérations liées à Blusam sur cet ensemble de données.

Exemple d'extrait de configuration :

```
dataSource:
  bluesamDs:
    driver-class-name: org.postgresql.Driver
    ...
    ...
bluesam:
  locks:
    lockTimeOut: 700
  cache: ehcache
  persistence: pgsql
  ehcache:
    resource-pool:
      size: 8GB
  write-behind:
    enabled: true
    threads: 8
    batchsize: 5000
  pgsql:
    dataSource : bluesamDs
```

Exemple d'extrait de configuration (avec le mode multi-schéma activé pour Blusam) :

```
dataSource:
  bluesamDs:
    driver-class-name: org.postgresql.Driver
    ...
    ...
bluesam:
  locks:
    lockTimeOut: 700
  cache: ehcache
  persistence: pgsql
  ehcache:
```

```
    resource-pool:
      size: 8GB
  write-behind:
    enabled: true
    threads: 8
    batchsize: 5000
  multiSchema: true
  pgsql:
    dataSource : bluesamDs
    schemas:
      - "schema1"
      - "schema2"
      - "schema3"
    fallbackSchema: schema3
```

> **Note**
>
> Les schémas de métadonnées Blusam, y compris les schémas répertoriés dans le `application-main.yml` fichier pour le mode multi-schémas, sont créés dans la base de données Blusam s'ils n'existent pas et si l'utilisateur dispose de privilèges suffisants.

Console d'administration Blusam

La console d'administration Blusam (BAC) est une application Web utilisée pour administrer le stockage Blusam. Pour plus d'informations sur le BAC, voirthe section called "Console d'administration Blusam".

Annexe

Attributs de métadonnées généraux des ensembles de données

Liste générale des attributs de sérialisation des métadonnées des ensembles de données :

- nom (de l'ensemble de données)
- type (KSDS, LargeKSDS, ESDS, LargeESDS ou RRDS)
- indicateur de préchauffage du cache (si l'ensemble de données doit être préchargé dans le cache au démarrage du serveur ou non)
- indicateur d'utilisation de la compression (s'il faut stocker les enregistrements dans un format compressé ou brut)

- date de création
- date de dernière modification
- indicateur d'enregistrement de longueur fixe (que les enregistrements de l'ensemble de données aient tous la même longueur ou non)
- longueur d'enregistrement -- uniquement significative pour une longueur d'enregistrement fixe
- taille de page (utilisée pour personnaliser les requêtes SQL paginées utilisées pour précharger le cache si nécessaire)
- taille (taille de l'ensemble de données - longueur cumulée des enregistrements)
- dernier décalage (décalage, c'est-à-dire le RBA du dernier enregistrement ajouté à l'ensemble de données)
- décalage suivant (prochain décalage disponible pour ajouter un nouvel enregistrement à l'ensemble de données)
- si cela est significatif, définition des clés utilisées par l'ensemble de données ; chaque clé étant définie par son type (clé principale ou faisant partie de la collection de clés secondaires) et par trois attributs :
  - offset : position dans l'enregistrement de l'octet de départ de la valeur clé ;
  - longueur : longueur en octets de la valeur clé. Ainsi, la valeur clé est le tableau d'octets qui est le sous-ensemble de l'enregistrement commençant à `key offset` et se terminant à la position ; `key offset + length - 1`
  - indicateur autorisé des doublons : indique si la clé accepte les doublons ou non (défini sur true pour autoriser les doublons).

Estimation de l'empreinte mémoire pour un ensemble de données donné

Pour les ensembles de données de petite ou moyenne taille, les métadonnées (tailles et index des différentes clés) seront entièrement chargées en mémoire. L'allocation des ressources appropriées à la machine hébergeant le serveur utilisé pour exécuter les applications modernisées nécessite de déterminer la consommation de mémoire induite par les ensembles de données Blusam, en particulier en ce qui concerne les métadonnées. Cette section fournit des réponses pratiques aux opérateurs concernés.

Les formules données ne s'appliquent qu'aux ensembles de données de petite ou moyenne taille de Blusam, sans utiliser la stratégie de stockage « grande ».

Métadonnées de l'ensemble de données Blusam

Pour un ensemble de données Blusam, les métadonnées sont divisées en deux parties :

- métadonnées de base : contient des informations globales sur l'ensemble de données. Son encombrement mémoire peut être considéré comme négligeable par rapport aux métadonnées internes.
- métadonnées internes : contiennent des informations sur la taille des enregistrements et les index clés ; lorsqu'un ensemble de données n'est pas vide, c'est ce qui consomme de la mémoire lorsqu'il est chargé sur le serveur d'applications hébergeant des applications modernisées. Les sections ci-dessous expliquent comment la mémoire consommée augmente avec le nombre d'enregistrements.

Calcul de l'empreinte interne des métadonnées

Carte des tailles d'enregistrements

Tout d'abord, les métadonnées internes stockent une carte contenant la taille de chaque enregistrement (sous forme d'entier) en fonction de son RBA (adresse d'octet relative, stockée sous forme de nombre long).

L'empreinte mémoire de cette structure de données est, en octets : `80 * number of records`

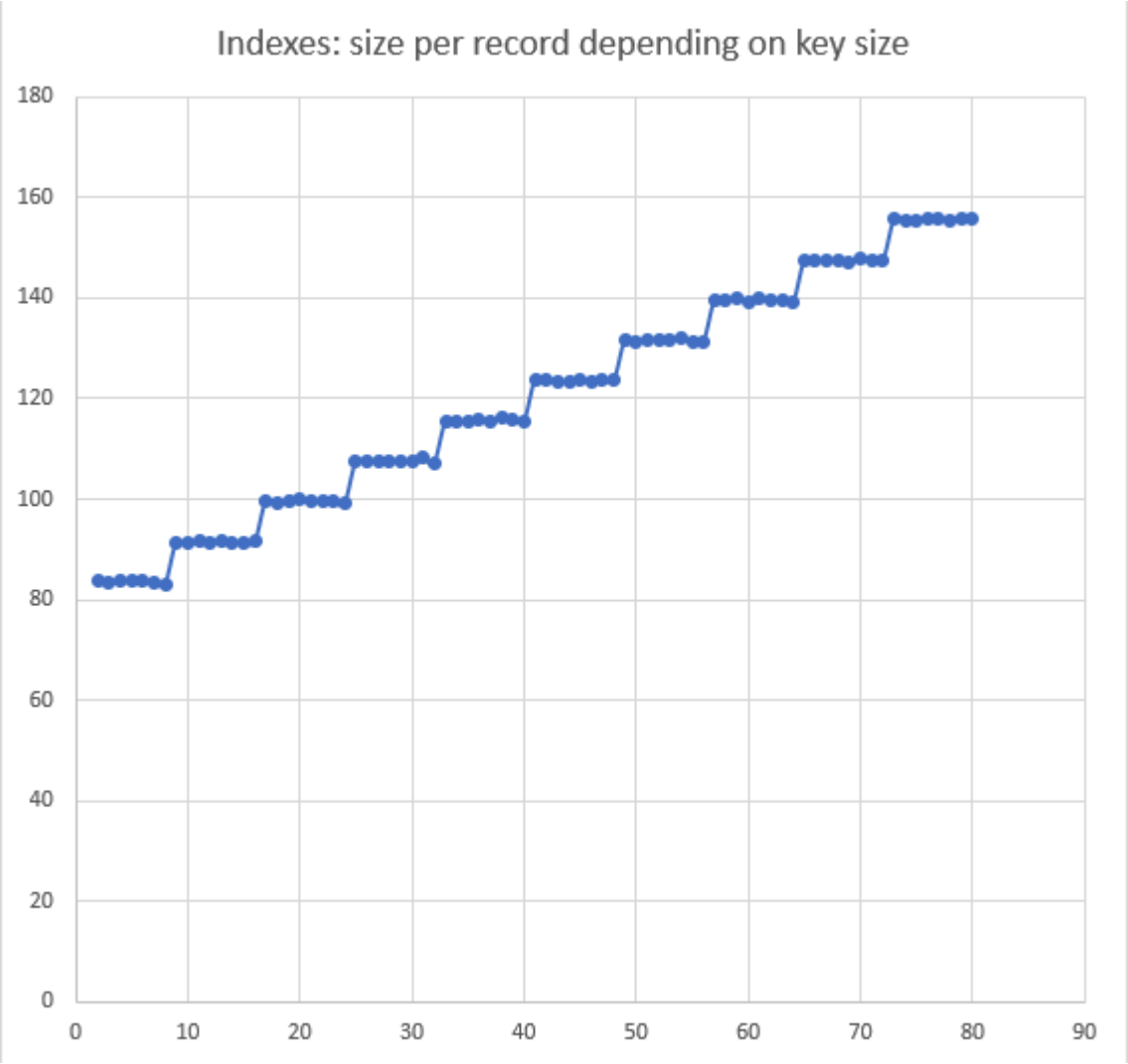
Cela s'applique à tous les types d'ensembles de données.

Index

En ce qui concerne les index de la clé primaire du KSDS ou des clés secondaires à la fois sur l'ESDS et le KSDS, le calcul de l'empreinte dépend de deux facteurs :

- le nombre d'enregistrements contenus dans l'ensemble de données ;
- la taille de la clé, en octets.

Le graphique ci-dessous montre la taille de l'index clé par enregistrement (axe Y) en fonction de la taille de la clé (axe X).



La formule correspondante pour évaluer l'empreinte d'un index clé donné d'un ensemble de données est la suivante :

```
index footprint = number of records * ( 83 + 8 (key length / 8))
```

où «/» représente la division entière.

Exemples :

- ensemble de données 1 :
  - nombre d'enregistrements = 459 996
  - longueur de clé = 15 donc (longueur de clé/8) = 1
  - empreinte de l'indice = 459 996 * (83 + (8*1)) = 41 859 636 octets (= 39 Mo environ)

- ensemble de données 2 :
  - nombre d'enregistrements = 13 095 783
  - longueur de clé = 18 donc (longueur de clé/8) = 2
  - empreinte de l'indice = 13 095 783 * (83 + (8*2)) = 1 296 482 517 octets (= 1,2 Go environ)

L'empreinte totale d'un ensemble de données donné est la somme de toutes les empreintes de tous les index clés et de l'empreinte de la carte des tailles d'enregistrements.

Par exemple, en prenant l'exemple de l'ensemble de données 2, qui ne possède qu'une seule clé, l'empreinte globale est la suivante :

- Carte des tailles d'enregistrements : 13 095 783 * 80 = 1 047 662 640 octets
- Index clés : 1 296 482 517 octets (voir ci-dessus)
- Encombrement total = 2 344 145 157 octets (= 2,18 Go environ)

Programmes disponibles dans l'application Web utilitaire

L'application Web utilitaire prend en charge divers programmes utilitaires de plate-forme existants, tels que IDCAMS, INFUTILB, SORT, etc. Pour configurer l'accès à l'application, voirConfiguration de l'accès aux utilitaires pour les applications gérées.

Liste des programmes

- Utilitaire JCLBCICS- Utilisé par lots pour définir le statut du jeu de données Bluesam sur. open/enabled or closed/disabled

Utilitaire JCLBCICS

JCLBCICS est un utilitaire JCL conçu pour définir le open/enabled or closed/disabled. An open/enabled status will block access to the dataset from batch programs while a closed/disabled statut du jeu de données bluesam afin de le rendre indisponible pour accéder aux services en ligne JICS.

Utilisation

- JCLBCICS modifie la colonne STATUS de la table FILE_TABLE de Jics et la colonne OPEN_STATUS de la table Bluesam BLUESAM_STATUS en fonction de la configuration groovy du nom du DD.

```
.open(ddName) -> ENABLED in Jics FILE_TABLE table, OPEN in Bluesam BLUESAM_STATUS
 table
.close(ddName) -> DISABLED in Jics FILE_TABLE table, CLOSED in Bluesam BLUESAM_STATUS
 table
```

- La taille du nom DD est configurable globalement dans le fichier `application-utility-pgm.yml` de configuration.

```
jclbcics.ddname.size: 7
```

- La taille globale du nom DD peut être remplacée à chaque étape en fournissant la taille remplacée avec les lignes suivantes en groovy, puis en utilisant StepParams comme paramètres pour cette étape.

```
TreeMap stepMapTransfo = [:]
Map stepParams = ["MapTransfo":stepMapTransfo]
stepParams["MapTransfo"]["JCLBCICS_OVERRIDDEN_SIZE"] = '7'
...
.withParameters(stepParams)
.runProgram("JCLBCICS")
```

- Lorsque vous définissez la taille du nom DD, la taille maximale effective du nom DD est de 8.
- Si la longueur du DDname est supérieure à la taille du nom DD fournie, elle sera tronquée à partir de la fin pour correspondre à la taille du nom DD.
- Les caractères génériques sont pris en charge dans DDName si * (astérisque) est ajouté à la fin du DDName ou si la longueur du DDName est inférieure à 8.

```
.open("DTSNAME*")
```

Exemple de code

```
// DD name with overridden size of 7 bytes
 def stepSTEP007(Object shell, Map params, Map programResults) {
   shell.with {
     if (checkValidProgramResults(programResults)) {
       TreeMap stepMapTransfo = [:]
       Map stepParams = ["MapTransfo":stepMapTransfo]
       stepParams["MapTransfo"]["JCLBCICS_OVERRIDDEN_SIZE"] = '7'
       return execStep("STEP007", "JCLBCICS", programResults, {
```

```
          mpr
            .withDatasetsConfiguration(new DatasetsConfiguration()
            .close("DTSNAME"))
            .withParameters(stepParams)
            .runProgram("JCLBCICS")
         })
      }
   }
}
```

AWS Console d'administration Blu Age Blusam

La console d'administration Blusam (BAC) est une application Web sécurisée permettant de gérer les ensembles de données Blusam. Ce guide couvre l'interface utilisateur du BAC. Pour la gestion à distance via les points de terminaison REST, voirthe section called "Points de terminaison REST de la console d'applications Blusam ".

Rubriques

- Déploiement du BAC
- Utilisation du BAC
- Format JSON LISTCAT

Déploiement du BAC

Le BAC est disponible sous la forme d'une application Web unique sécurisée, utilisant le format d'archive Web (.war). Il est destiné à être déployé parallèlement à l' BluAge application Gapwalk, sur un serveur d'applications Apache Tomcat, mais peut également être déployé en tant qu'application autonome. Le BAC hérite de l'accès au stockage Blusam de la configuration de l'application Gapwalk, le cas échéant.

Le BAC possède son propre fichier de configuration dédié, nomméapplication-bac.yml. Pour plus de détails sur la configuration, consultezthe section called "Fichier de configuration dédié au BAC".

Le BAC est sécurisé. Pour plus de détails sur la configuration de sécurité, consultezthe section called "Configuration de la sécurité pour le BAC".

Fichier de configuration dédié au BAC

Déploiement autonome : si le BAC est déployé seul sur l'application Gapwalk, la connexion au stockage Blusam doit être configurée dans le fichier de configuration application-bac.yml.

Les valeurs par défaut pour la configuration des ensembles de données utilisée pour parcourir les enregistrements des ensembles de données doivent être définies dans le fichier de configuration. Voir the section called "Parcourir les enregistrements d'un ensemble de données". La page de navigation des enregistrements peut utiliser un mécanisme de masque optionnel qui permet d'afficher une vue structurée du contenu d'un enregistrement. Certaines propriétés ont un impact sur l'affichage des enregistrements lorsque des masques sont utilisés.

Les propriétés configurables suivantes doivent être définies dans le fichier de configuration. L'application BAC ne prend aucune valeur par défaut pour ces propriétés.

Clé	Type	Description
`bac.crud.limit`	integer	Une valeur entière positive indiquant le nombre maximum d'enregistrements renvoyés lors de la consultation des enregistrements. Utiliser `0` signifie illimité. Valeur recommandée : `10` (puis ajustez la valeur définie par ensemble de données sur la page de navigation, en fonction de vos besoins).
`bac.crud.encoding`	chaîne	Le nom du jeu de caractère s par défaut, utilisé pour décoder les octets des enregistrements sous forme de contenu alphanumérique. Le nom du jeu de caractère s fourni doit être compatible avec Java (veuillez consulter la documentation Java pour

Clé	Type	Description
		les jeux de caractères pris en charge). Valeur recommandée : le jeu de caractères existant utilisé sur l'ancienne plate-forme d'où proviennent les ensembles de données ; il s'agira la plupart du temps d'une variante EBCDIC.
`bac.crud.initCharacter`	chaîne	Le caractère par défaut (octet) utilisé pour initialiser les éléments de données. Deux valeurs spéciales peuvent être utilisées : `"LOW-VALUE"` l'octet 0x00 (valeur recommandée) et `"HI-VALUE"` l'octet 0xFF. Utilisé lorsque des masques sont appliqués.
`bac.crud.defaultCharacter`	chaîne	Le caractère par défaut (octet), sous forme de chaîne d'un caractère, utilisé pour le remplissage des enregistrements (à droite). Valeur recommandée : `" "` (espace) Utilisé lorsque des masques sont appliqués.

Clé	Type	Description
`bac.crud.blankCharacter`	chaîne	Le caractère par défaut (octet), sous forme de chaîne d'un caractère, utilisé pour représenter les blancs dans les enregistrements. Valeur recommandée : `"  "` (espace). Utilisé lorsque des masques sont appliqués.
`bac.crud.strictZoned`	boolean	Un drapeau pour indiquer le mode zoné utilisé pour l'enregistrement. Dans le cas contraire`true`, le mode zone stricte sera utilisé ; dans le cas contraire`false`, le mode zoné modifié sera utilisé. Valeur recommandée :`true`. Utilisé lorsque des masques sont appliqués.
`bac.crud.decimalSeparator`	chaîne	Le caractère utilisé comme séparateur décimal dans les champs édités numériquement (utilisé lorsque des masques sont appliqués).
`bac.crud.currencySign`	chaîne	Le caractère par défaut, sous forme de chaîne d'un caractère, utilisé pour représenter la devise dans les champs édités numérique ment, lors de l'application du formatage (utilisé lorsque des masques sont appliqués).

Clé	Type	Description
`bac.crud.pictureCurrencySign`	chaîne	Le caractère par défaut, sous forme de chaîne d'un caractère, utilisé pour représenter la devise dans les images de champs édités numériquement (utilisé lorsque des masques sont appliqués).

L'exemple suivant est un extrait de fichier de configuration.

```
bac.crud.limit: 10
bac.crud.encoding: ascii
bac.crud.initCharacter: "LOW-VALUE"
bac.crud.defaultCharacter: " "
bac.crud.blankCharacter: " "
bac.crud.strictZoned: true
bac.crud.decimalSeparator: "."
bac.crud.currencySign: "$"
bac.crud.pictureCurrencySign: "$"
```

Configuration de la sécurité pour le BAC

La configuration de la sécurité pour le BAC repose sur les mécanismes détaillés dans cette page de documentation. Le schéma d'authentification est le suivant OAuth2, et les détails de configuration pour Amazon Cognito ou Keycloak sont fournis.

Bien que la configuration générale puisse être appliquée, certaines informations concernant le BAC doivent être détaillées ici. L'accès aux fonctionnalités du BAC est protégé par une politique basée sur les rôles et repose sur les rôles suivants.

- UTILISATEUR_RÔLE :
  - Rôle d'utilisateur de base
  - Aucune importation, exportation, création ou suppression d'ensembles de données n'est autorisée
  - Aucun contrôle sur les politiques de mise en cache
  - Aucune fonctionnalité d'administration n'est autorisée

- ROLE_ADMIN :
  - Hérite des autorisations ROLE_USER
  - Toutes les opérations sur les ensembles de données sont autorisées
  - Administration des politiques de mise en cache autorisée

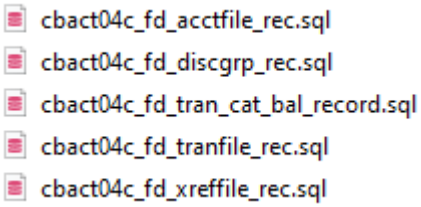
Installation des masques

Dans le stockage Blusam, les enregistrements des ensembles de données sont stockés dans une colonne de tableau d'octets de la base de données, pour des raisons de polyvalence et de performances. L'accès à une vue structurée, à l'aide de champs, des dossiers commerciaux, basée sur le point de vue de l'application est une fonctionnalité pratique du BAC. Cela repose sur les masques SQL produits au cours du processus de modernisation BluAge piloté.

Pour que les masques SQL soient générés, assurez-vous de définir l'option appropriée (`export.SQL.masks`) sur true dans la configuration du centre de BluInsights transformation :

Les masques font partie des artefacts de modernisation qui peuvent être téléchargés BluInsights pour un projet donné. Ce sont des scripts SQL, organisés par des programmes modernisés, donnant un point de vue applicatif sur les enregistrements de jeux de données.

Par exemple, à l'aide de l' CardDemo exemple d'application AWS, vous pouvez trouver dans les artefacts téléchargés à partir du résultat de modernisation de cette application, les masques SQL suivants pour le programme CBact04c.cbl :



Chaque nom de masque SQL est la concaténation du nom du programme et du nom de la structure d'enregistrement pour un ensemble de données donné au sein du programme.

Par exemple, en regardant le programme [CBact04C.cbl], l'entrée de contrôle de fichier donnée :

```
    FILE-CONTROL.
        SELECT TCATBAL-FILE ASSIGN TO TCATBALF
               ORGANIZATION IS INDEXED
               ACCESS MODE  IS SEQUENTIAL
               RECORD KEY   IS FD-TRAN-CAT-KEY
               FILE STATUS  IS TCATBALF-STATUS.
```

est associé à la définition d'enregistrement FD donnée

```
       FILE SECTION.
       FD  TCATBAL-FILE.
       01  FD-TRAN-CAT-BAL-RECORD.
           05 FD-TRAN-CAT-KEY.
              10 FD-TRANCAT-ACCT-ID             PIC 9(11).
              10 FD-TRANCAT-TYPE-CD             PIC X(02).
              10 FD-TRANCAT-CD                  PIC 9(04).
           05 FD-FD-TRAN-CAT-DATA               PIC X(33).
```

Le masque SQL correspondant nommé `cbact04c_fd_tran_cat_bal_record.SQL` est le masque qui donne le point de vue du programme CBact04c.cbl sur l'enregistrement FD nommé. `FD-TRAN-CAT-BAL-RECORD`

Son contenu est le suivant :

```
-- Generated by Blu Age Velocity
-- Mask : cbact04c_fd_tran_cat_bal_record

INSERT INTO mask (name, length) VALUES ('cbact04c_fd_tran_cat_bal_record', 50);
```

```
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk) VALUES
 ('fd_trancat_acct_id', 1, 11, false, 'zoned', 'integerSize=11!fractionalSize=0!
signed=false', (SELECT MAX(id) FROM mask));
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk) VALUES
 ('fd_trancat_type_cd', 12, 2, false, 'alphanumeric', 'length=2', (SELECT MAX(id) FROM
 mask));
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk)
 VALUES ('fd_trancat_cd', 14, 4, false, 'zoned', 'integerSize=4!fractionalSize=0!
signed=false', (SELECT MAX(id) FROM mask));
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk) VALUES
 ('fd_fd_tran_cat_data', 18, 33, false, 'alphanumeric', 'length=33', (SELECT MAX(id)
 FROM mask));
```

Les masques sont stockés dans le magasin Blusam à l'aide de deux tables :

- masque : utilisé pour identifier les masques. Les colonnes du tableau de masse sont les suivantes :
  - nom : utilisé pour stocker l'identification du masque (utilisé comme clé primaire, il doit donc être unique)
  - longueur : taille en octets du masque d'enregistrement
- mask_item : utilisé pour stocker les détails du masque. Chaque champ élémentaire d'une définition d'enregistrement FD produira une ligne dans la table mask_item, avec des détails sur la façon d'interpréter la partie d'enregistrement donnée. Les colonnes de la table mask_item sont les suivantes :
  - nom : nom du champ d'enregistrement, basé sur le nom élémentaire, en minuscules et en remplaçant le tiret par un trait de soulignement
  - c_offset : décalage basé sur 1 de la sous-partie de l'enregistrement, utilisé pour le contenu du champ
  - longueur : longueur en octets de la sous-partie de l'enregistrement, utilisée pour le contenu du champ
  - skip : drapeau pour indiquer si la partie de l'enregistrement donnée doit être ignorée ou non, dans la présentation de la vue
  - type : le type de champ (basé sur sa clause d'image existante)
  - options : options de type supplémentaires, dépendantes du type
  - mask_fk : référence à l'identifiant du masque auquel associer cet élément

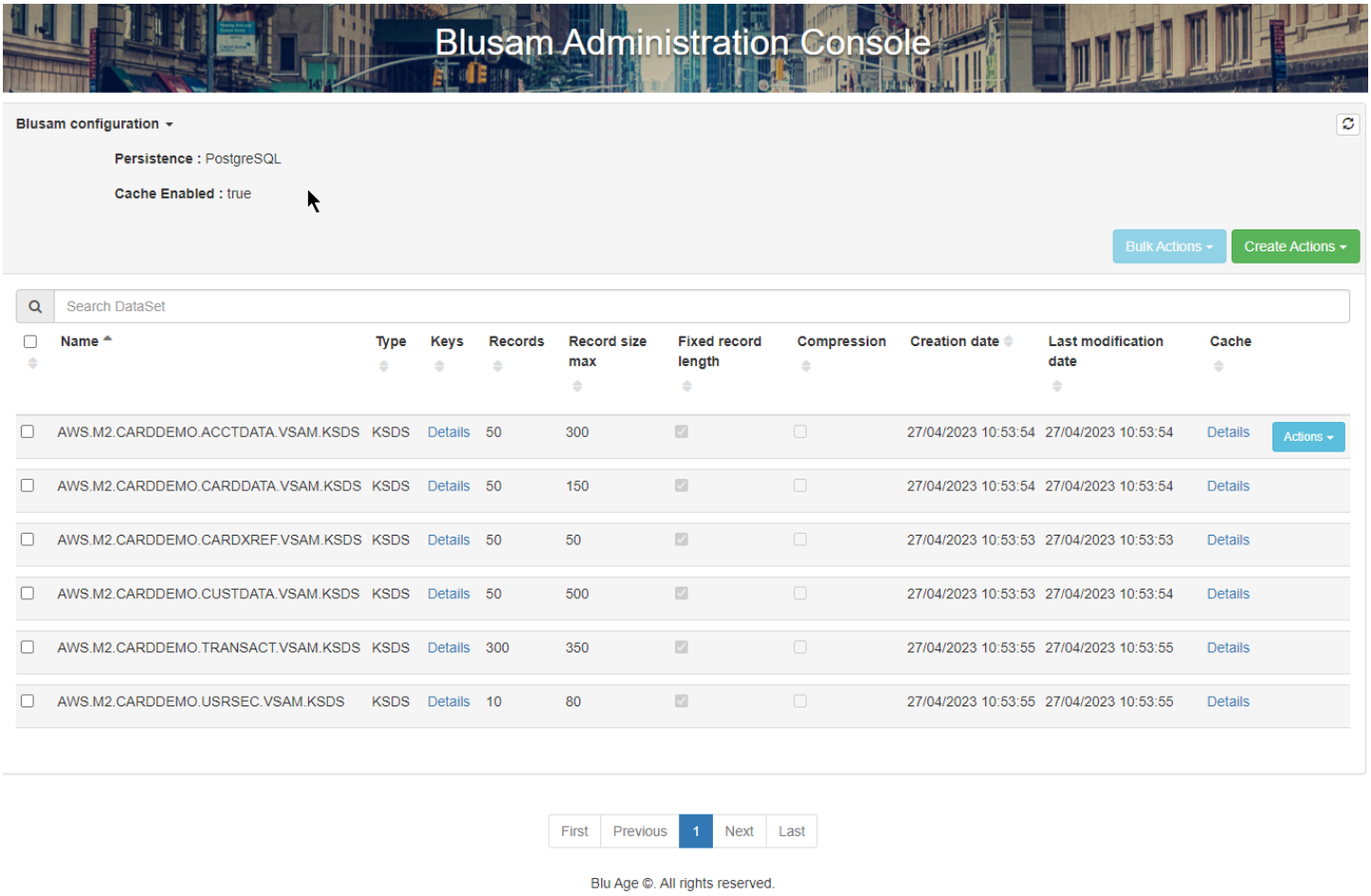
Remarques :

- Les masques SQL représentent le point de vue d'un programme sur les enregistrements d'un ensemble de données : plusieurs programmes peuvent avoir un point de vue différent sur un ensemble de données donné ; installez uniquement les masques que vous jugez pertinents pour votre objectif.
- Un masque SQL peut également représenter le point de vue d'un programme basé sur une structure de données 01 issue de la section WORKING STORAGE, et pas seulement d'un enregistrement FD. Les masques SQL sont organisés en sous-dossiers en fonction de leur nature :
  - Les masques basés sur les enregistrements FD seront situés dans le sous-dossier nommé `file`
  - 01 les masques basés sur la structure de données seront situés dans le sous-dossier nommé `working`

  Alors que les définitions des enregistrements FD correspondent toujours au contenu des enregistrements d'un ensemble de données, les structures de données 01 peuvent ne pas être alignées ou ne représenter qu'un sous-ensemble d'un enregistrement d'ensemble de données. Avant de les utiliser, inspectez le code et comprenez les éventuelles lacunes.

Utilisation du BAC

Le BAC étant sécurisé et autorisant l'utilisation des fonctionnalités en fonction du rôle de l'utilisateur, la première étape pour accéder à l'application consiste à vous authentifier. Après l'étape d'authentification, vous serez redirigé vers la page d'accueil. La page d'accueil présente la liste paginée des ensembles de données trouvés dans le stockage Blusam :



Pour revenir à la page d'accueil avec la liste des ensembles de données, choisissez le logo Blu Age dans le coin supérieur gauche de n'importe quelle page de l'application. L'image suivante montre le logo.

L'en-tête pliable, intitulé « BluSam configuration », contient des informations sur la configuration de BluSam stockage utilisée :

- `Persistence`: le moteur de stockage persistant (PostgreSQL)
- `Cache Enabled`: si le cache de stockage est activé

Sur le côté droit de l'en-tête, deux listes déroulantes répertoriant chacune les opérations liées aux ensembles de données :

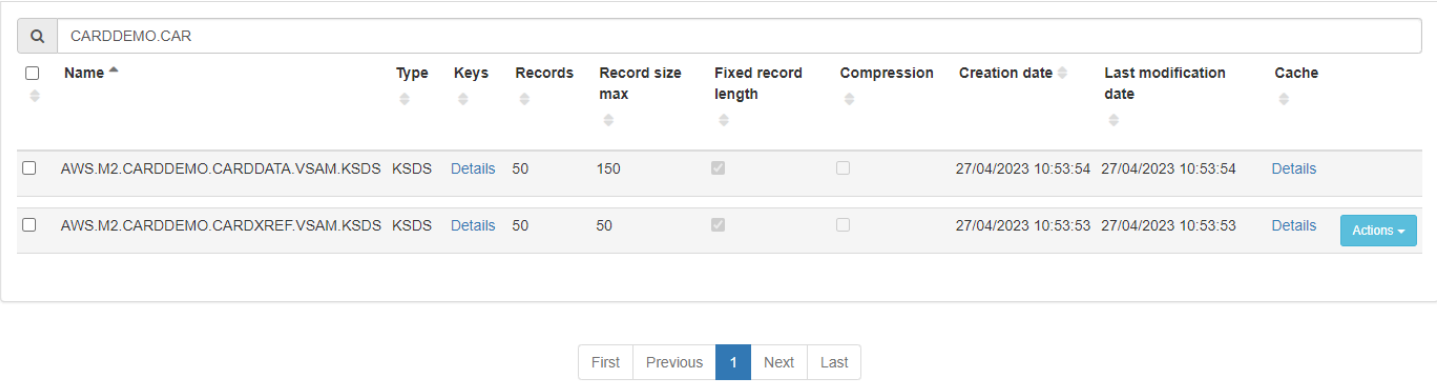
- Actions groupées

- Créez des actions

Pour en savoir plus sur le contenu détaillé de ces listes, consultezthe section called "Opérations d'ensembles de données existantes".

Le bouton Actions groupées est désactivé lorsqu'aucun ensemble de données n'a été sélectionné.

Vous pouvez utiliser le champ de recherche pour filtrer la liste en fonction des noms des ensembles de données :

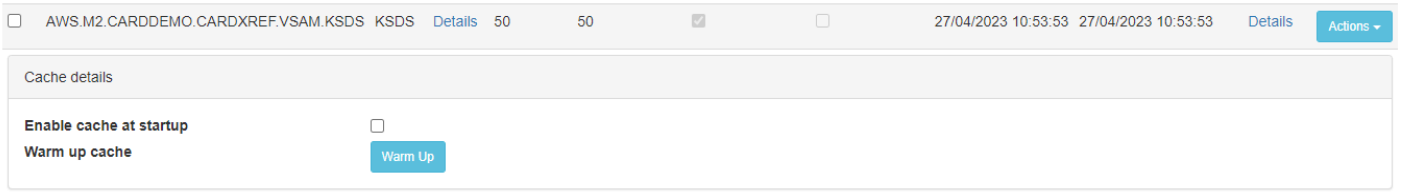


La liste paginée qui suit montre un ensemble de données par ligne de tableau, avec les colonnes suivantes :

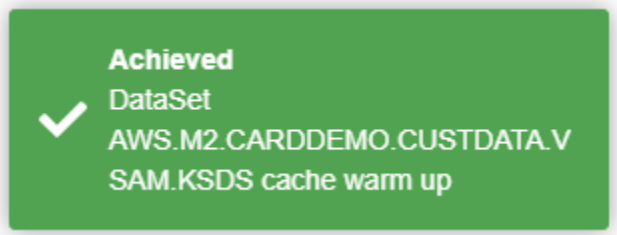
- Case à cocher : case à cocher pour sélectionner l'ensemble de données actuel.
- Nom : nom de l'ensemble de données.
- Type : type de jeu de données, l'un des suivants :
  - KSDS
  - ESDS
  - RDS
- Clés : lien permettant d'afficher ou de masquer les détails relatifs aux clés (le cas échéant). Par exemple, le KSDS donné possède la clé primaire obligatoire et une clé alternative.

  Il y a une ligne par touche, avec les colonnes suivantes. Aucun des champs n'est modifiable.

  - Nature de la clé : clé primaire ou clé alternative
  - Nom : le nom de la clé
  - Unique : si la clé accepte les entrées en double
  - Décalage : décalage de la touche de départ dans l'enregistrement
  - Longueur : longueur en octets de la partie clé de l'enregistrement
- Enregistrements : nombre total d'enregistrements de l'ensemble de données.
- Taille maximale des enregistrements : taille maximale des enregistrements, exprimée en octets.
- Longueur d'enregistrement fixe : case à cocher qui indique si les enregistrements sont de longueur fixe (sélectionnée) ou variable (non sélectionnée).
- Compression : case à cocher qui indique si la compression est appliquée (sélectionnée) ou non (désélectionnée) aux index stockés.
- Date de création : date à laquelle l'ensemble de données a été créé dans le stockage Blusam.
- Date de dernière modification : date à laquelle l'ensemble de données a été mis à jour pour la dernière fois dans le stockage Blusam.
- Cache : lien permettant d'afficher ou de masquer les détails de la stratégie de mise en cache appliquée à cet ensemble de données.



  - Activer le cache au démarrage : case à cocher pour spécifier la stratégie de mise en cache de démarrage pour cet ensemble de données. Si cette option est sélectionnée, l'ensemble de données sera chargé dans le cache au démarrage.
  - Réchauffer le cache : bouton permettant de charger l'ensemble de données donné dans le cache, en commençant immédiatement (mais l'hydratation du cache prend un certain temps, en fonction de la taille de l'ensemble de données et du nombre de clés). Une fois que l'ensemble de données est chargé dans le cache, une notification semblable à la suivante apparaît.

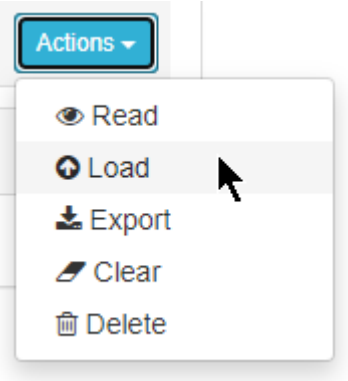


- Actions : liste déroulante des opérations possibles sur les ensembles de données. Pour plus d'informations, consultez the section called "Opérations d'ensembles de données existantes".

Au bas de la page, un widget de navigation paginé normal permet de parcourir les pages de la liste des ensembles de données.

Opérations d'ensembles de données existantes

Pour chaque ensemble de données de la liste paginée, il existe une liste déroulante Actions dont le contenu est le suivant :



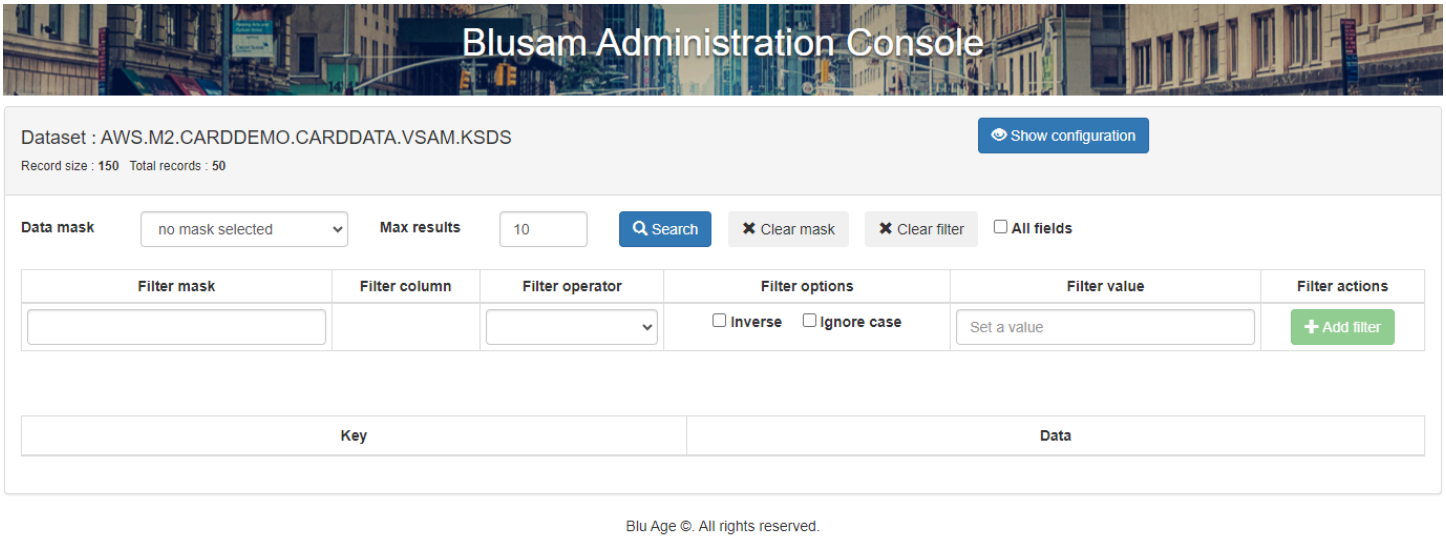
Chaque élément de la liste est un lien actif qui permet d'effectuer l'action spécifiée sur l'ensemble de données :

- Lire : parcourir les enregistrements des ensembles de données
- Charger : importer des enregistrements à partir d'un ancien fichier d'ensemble de données
- Exportation : exportez les enregistrements vers un fichier plat (compatible avec les anciens systèmes)
- Effacer : supprimer tous les enregistrements de l'ensemble de données
- Supprimer : supprimer l'ensemble de données du stockage

Les détails de chaque action sont fournis dans les sections suivantes.

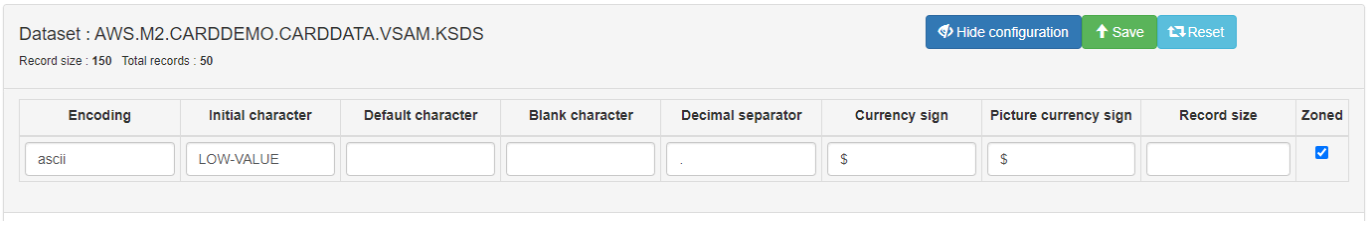
Parcourir les enregistrements d'un ensemble de données

Lorsque vous choisissez l'action Lire pour un ensemble de données donné, vous obtenez la page suivante.



La page est composée de :

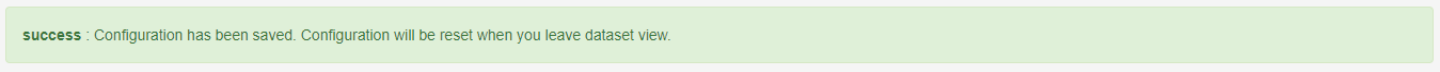
- un en-tête, avec :
  - Ensemble de données : nom de l'ensemble de données
  - Taille d'enregistrement : longueur d'enregistrement fixe, exprimée en octets
  - Nombre total d'enregistrements : le nombre total d'enregistrements stockés pour cet ensemble de données
  - Afficher le bouton de configuration (sur le côté droit) : un bouton à bascule pour afficher/masquer la configuration de l'ensemble de données. Dans un premier temps, la configuration est masquée. Lorsque vous utilisez le bouton, la configuration s'affiche, comme indiqué dans l'image suivante.



    Lorsque la configuration est affichée, deux nouveaux boutons : Enregistrer et Réinitialiser, utilisés respectivement pour :

    - enregistrer la configuration de cet ensemble de données et de la session de travail en cours
    - rétablir les valeurs par défaut de la configuration pour tous les champs.
  - Liste de propriétés configurables pour adapter l'expérience de navigation à un ensemble de données donné.

Les propriétés configurables correspondent aux propriétés de configuration décrites dansthe section called "Fichier de configuration dédié au BAC". Reportez-vous à cette section pour comprendre la signification de chaque colonne et les valeurs applicables. Chaque valeur peut être redéfinie ici pour l'ensemble de données et enregistrée pour la session de travail (à l'aide du bouton Enregistrer). Après avoir enregistré la configuration, une bannière similaire à celle illustrée dans l'image suivante apparaît.



La bannière indique que la session de travail se termine lorsque vous quittez la page en cours.

Il existe une propriété configurable supplémentaire qui n'est pas documentée dans la section de configuration : Taille de l'enregistrement. Ceci est utilisé pour spécifier une taille d'enregistrement donnée, exprimée en octets, qui filtrera les masques applicables à cet ensemble de données : seuls les masques dont la longueur totale correspond à la taille d'enregistrement donnée seront répertoriés dans la liste déroulante des masques de données.

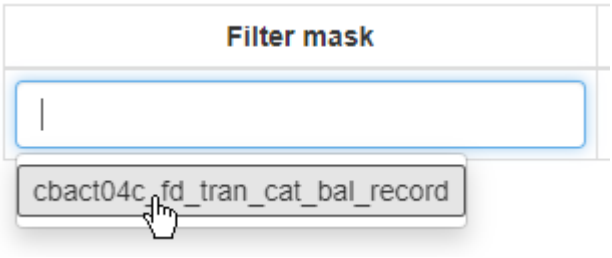
La récupération des enregistrements de l'ensemble de données est déclenchée par le bouton Rechercher, en utilisant toutes les options et tous les filtres à proximité.

Première ligne d'options :

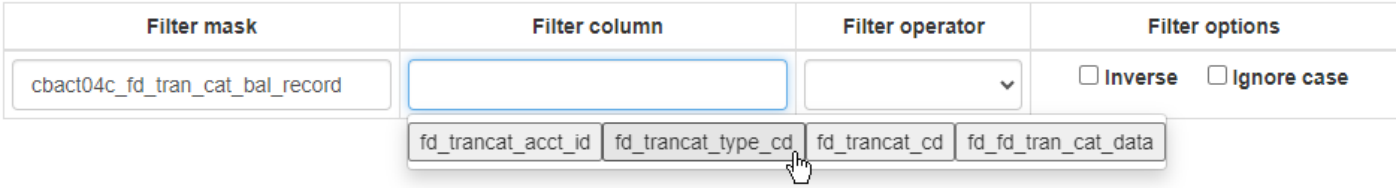
- la liste déroulante des masques de données indique les masques applicables (en respectant la taille de l'enregistrement). Veuillez noter qu'il ne suffit pas de faire correspondre la taille de l'enregistrement pour obtenir un masque applicable efficace. La définition du masque doit également être compatible avec le contenu des enregistrements. Le masque de données sélectionné ici a
- Nombre maximum de résultats : limite le nombre d'enregistrements extraits par la recherche. Défini sur 0 pour un nombre illimité de résultats (paginés) à partir de l'ensemble de données.
- Bouton de recherche : lancez la récupération des enregistrements à l'aide de filtres et d'options
- Bouton Effacer le masque : efface le masque utilisé, le cas échéant, et fait revenir la page de résultats à une présentation des clés/données brutes.
- Bouton Effacer le filtre : efface le ou les filtres utilisés, le cas échéant, et met à jour la page de résultats en conséquence.
- Tous les champs basculent : lorsque cette option est sélectionnée, les éléments de masque définis avec `skip = true` sont affichés de toute façon, sinon les éléments de masque avec `skip = true` sont masqués.

Lignes de filtres suivantes : Il est possible de définir une liste de filtres, basée sur l'utilisation des conditions de filtrage appliquées aux champs (colonnes) d'un masque donné, comme le montre l'image suivante.

- Masque de filtre : nom du masque à partir duquel sélectionner la colonne de filtrage. Lorsque vous sélectionnez le champ, la liste des masques applicables s'affiche. Vous pouvez choisir le masque que vous souhaitez dans cette liste.

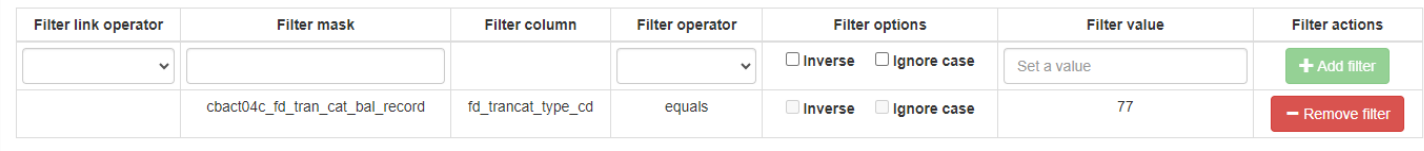


- Colonne de filtre : nom du champ (colonne) du masque, utilisé pour filtrer les enregistrements. Lorsque vous choisissez le champ, la liste des colonnes du masque apparaît. Pour remplir le champ de la colonne Filtre, sélectionnez la cellule souhaitée.



- Opérateur de filtre : opérateur à appliquer à la colonne sélectionnée. Les opérateurs suivants sont disponibles.
  - égal à : la valeur de colonne de l'enregistrement doit être égale à la valeur du filtre
  - commence par : la valeur de colonne de l'enregistrement doit commencer par la valeur du filtre
  - se termine par : la valeur de colonne de l'enregistrement doit se terminer par la valeur du filtre
  - contient : la valeur de colonne de l'enregistrement doit contenir la valeur du filtre
- Options de filtre :
  - Inverse : appliquez la condition inverse pour l'opérateur de filtre ; par exemple, « égal à » est remplacé par « non égal à » ;
  - Ignorer le cas : ignorer le cas sur les comparaisons alphanumériques pour l'opérateur de filtre
- Valeur du filtre : valeur utilisée pour la comparaison par l'opérateur du filtre avec la colonne du filtre.

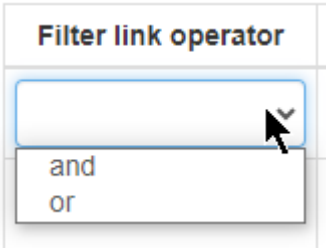
Une fois le nombre minimal d'éléments de filtre défini (au moins : le masque de filtre, la colonne de filtre, l'opérateur de filtre et la valeur du filtre doivent être définis), le bouton Ajouter un filtre est activé et le fait de cliquer dessus crée une nouvelle condition de filtre sur les enregistrements récupérés. Une autre ligne de condition de filtre vide est ajoutée en haut et la condition de filtre ajoutée comporte un bouton Supprimer le filtre qui peut être utilisé pour supprimer la condition de filtre donnée :



Lorsque vous lancez la recherche, les résultats filtrés apparaissent dans un tableau paginé.

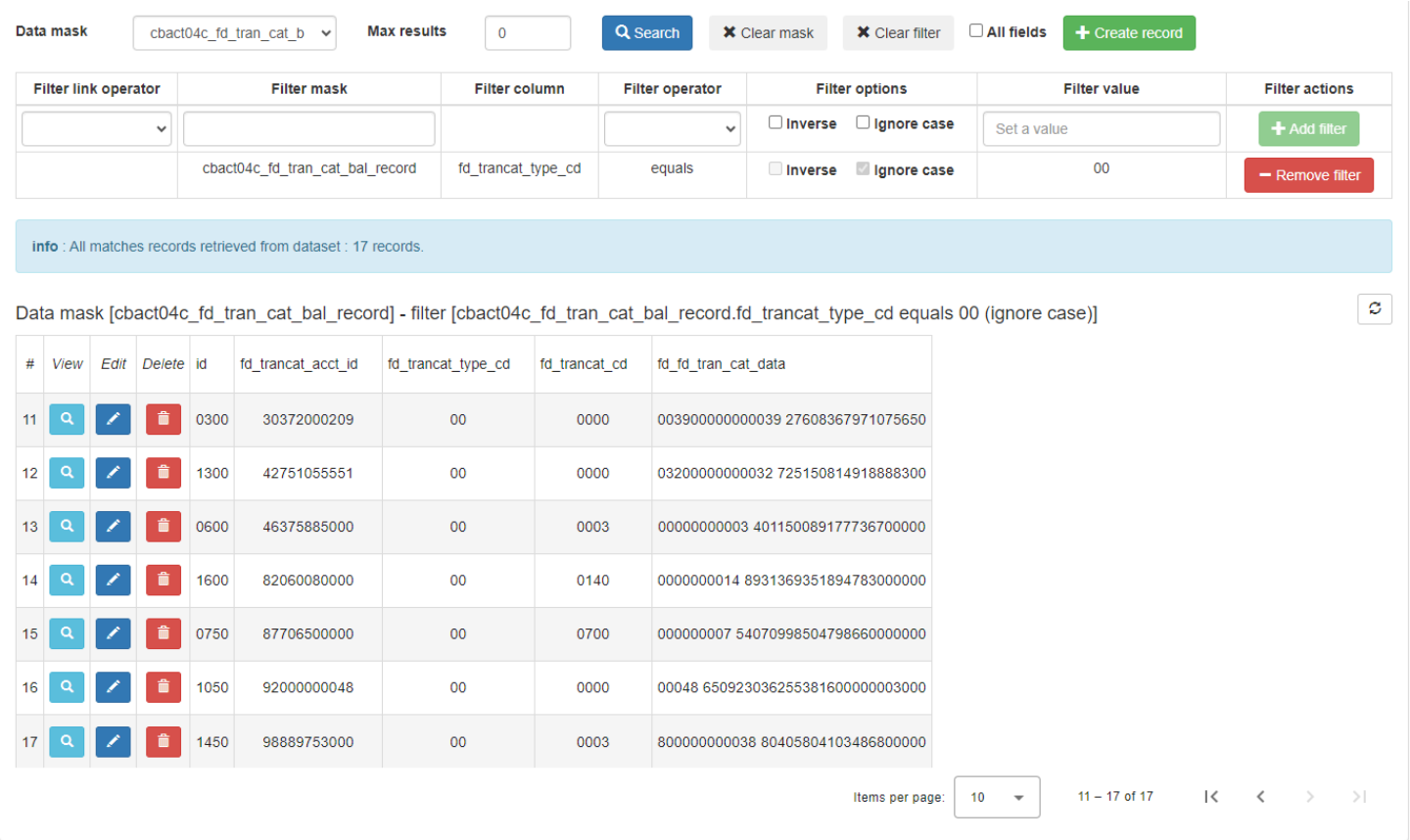
Remarque

- Les filtres successifs sont liés par un et ou un ou. Chaque nouvelle définition de filtre commence par définir l'opérateur de lien, comme illustré dans l'image suivante.



- Il se peut qu'aucun enregistrement ne corresponde aux conditions de filtre données.

Dans le cas contraire, le tableau des résultats ressemble à celui de l'image suivante.



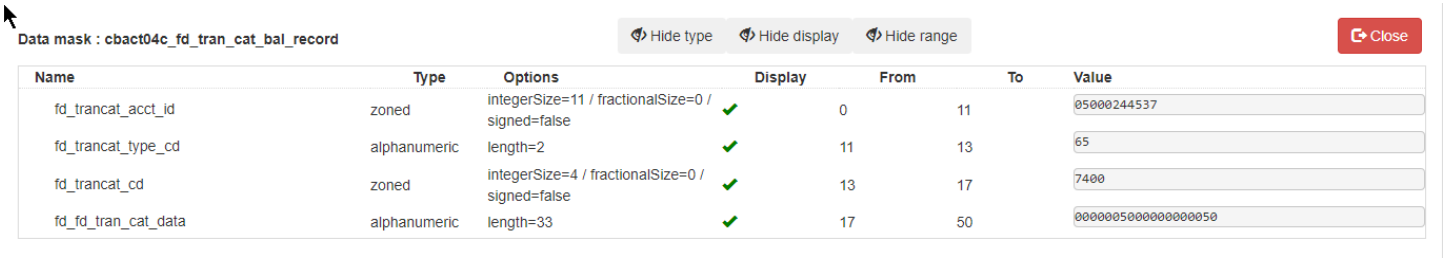
Un en-tête indique le nombre total d'enregistrements qui répondent aux conditions du filtre. Après l'en-tête, vous pouvez voir ce qui suit.

- Rappel du masque de données utilisé (le cas échéant) et des conditions du filtre.
- Un bouton d'actualisation que vous pouvez utiliser pour déclencher l'actualisation de l'ensemble du tableau des résultats avec les dernières valeurs du stockage Blusam (tel qu'il aurait pu être mis à jour par un autre utilisateur par exemple).

Pour chaque enregistrement extrait, le tableau comporte une ligne qui indique le résultat de l'application du masque de données au contenu des enregistrements. Chaque colonne est l'interprétation de la sous-partie de l'enregistrement en fonction du type de colonne (et en utilisant le codage sélectionné). Trois boutons se trouvent à gauche de chaque ligne :

- un bouton en forme de loupe : mène à une page dédiée affichant le contenu détaillé de l'enregistrement
- un bouton stylet : mène à une page d'édition dédiée au contenu de l'enregistrement :
- un bouton de corbeille : utilisé pour supprimer l'enregistrement donné du stockage de blusam

Afficher le contenu de l'enregistrement en détail :



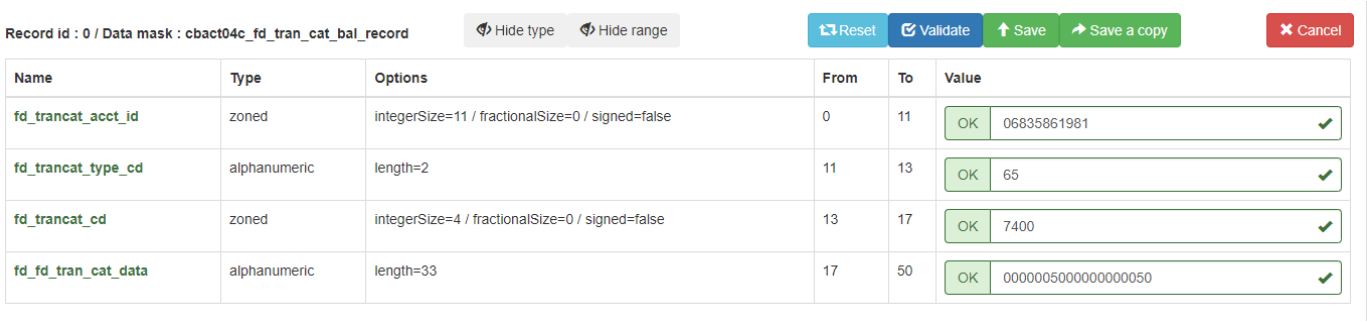
- Trois boutons permettant de masquer ou d'afficher certaines colonnes :
  - Masquer/afficher le type
  - Masquer/afficher le drapeau d'affichage
  - Masquer/afficher la gamme
- Pour quitter cette page dédiée et revenir au tableau des résultats, choisissez Fermer.
- Chaque ligne représente une colonne du masque de données, avec les colonnes suivantes :
  - Nom : le nom de la colonne
  - Type : le type de colonne
  - Affichage : indicateur d'affichage ; une coche verte sera affichée si l'élément de masque correspondant est défini avec`skip = false`, sinon une croix rouge sera affichée
  - De et vers : la plage de base 0 pour la sous-partie de l'enregistrement
  - Valeur : valeur interprétée de la sous-partie de l'enregistrement, en utilisant le type et le codage

Modification du contenu de l'enregistrement :

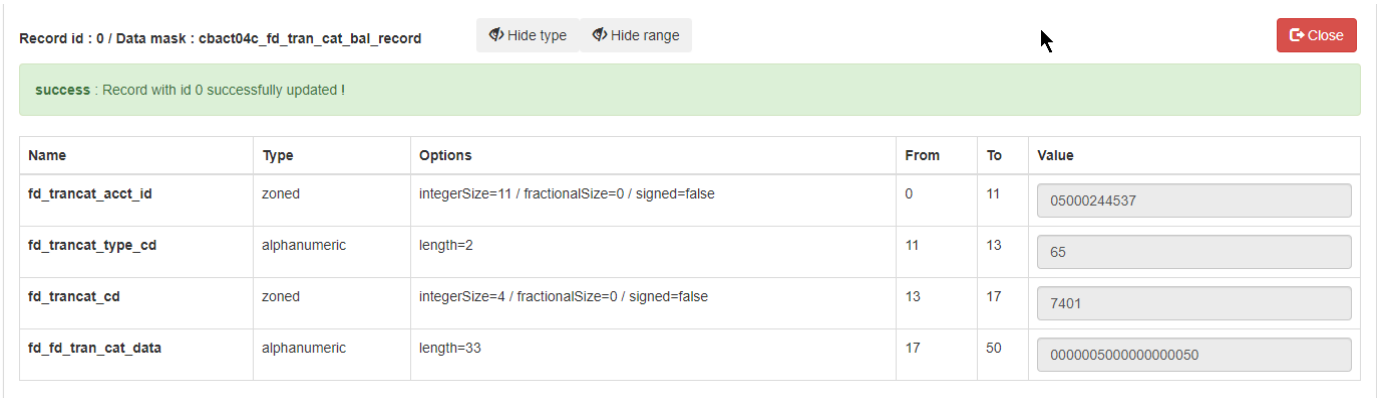
La page d'édition est similaire à la page d'affichage décrite ci-dessus, sauf que les valeurs des éléments du masque sont modifiables. Trois boutons contrôlent le processus de mise à jour :

- Réinitialiser : rétablit les valeurs modifiables par rapport aux valeurs d'enregistrement initiales (avant toute édition) ;

- Valider : valide l'entrée, en ce qui concerne le type d'élément du masque. Pour chaque élément du masque, le résultat de la validation sera imprimé à l'aide d'étiquettes visuelles (0Ket d'une case à cocher si la validation réussit, ERR0R d'une croix rouge en cas d'échec de la validation, ainsi que d'un message d'erreur indiquant l'échec de la validation). Si la validation est réussie, deux nouveaux boutons apparaîtront :
  - Enregistrer : essayez de mettre à jour l'enregistrement existant dans le stockage Blusam
  - Enregistrer une copie : essayer de créer un nouvel enregistrement dans le stockage Blusam



  - Si l'enregistrement dans le stockage est réussi, un message s'affiche et la page passe en mode lecture seule (les valeurs des éléments du masque ne peuvent plus être modifiées) :



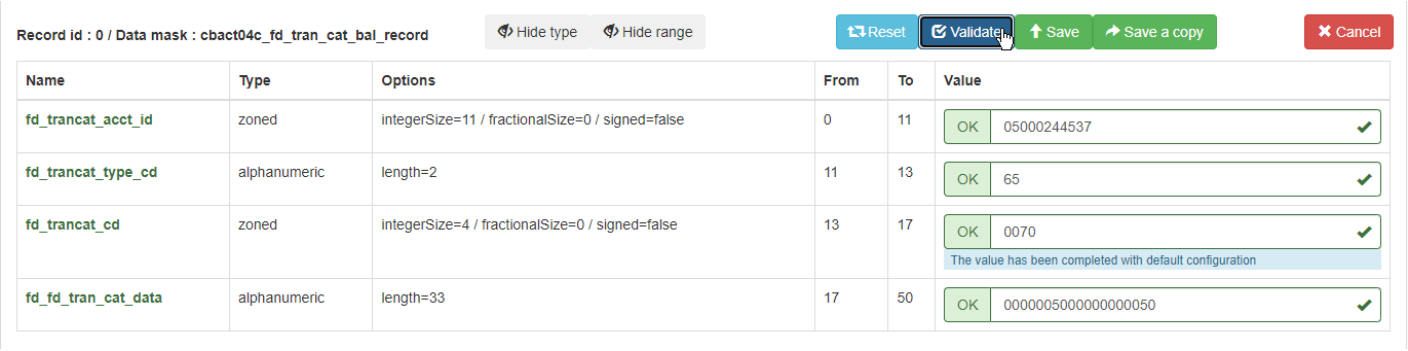
  - Si, pour une raison quelconque, la persistance des enregistrements dans le stockage échoue, un message d'erreur s'affiche en rouge, indiquant la raison de l'échec. Le cas d'échec le plus courant est que le stockage de l'enregistrement entraînerait une corruption de clé (clé non valide ou dupliquée). Pour une illustration, reportez-vous à la note suivante.
  - Pour quitter, cliquez sur le bouton Fermer.
- Annuler : met fin à la session de modification, ferme la page et vous ramène à la page de liste des enregistrements.

Remarque :

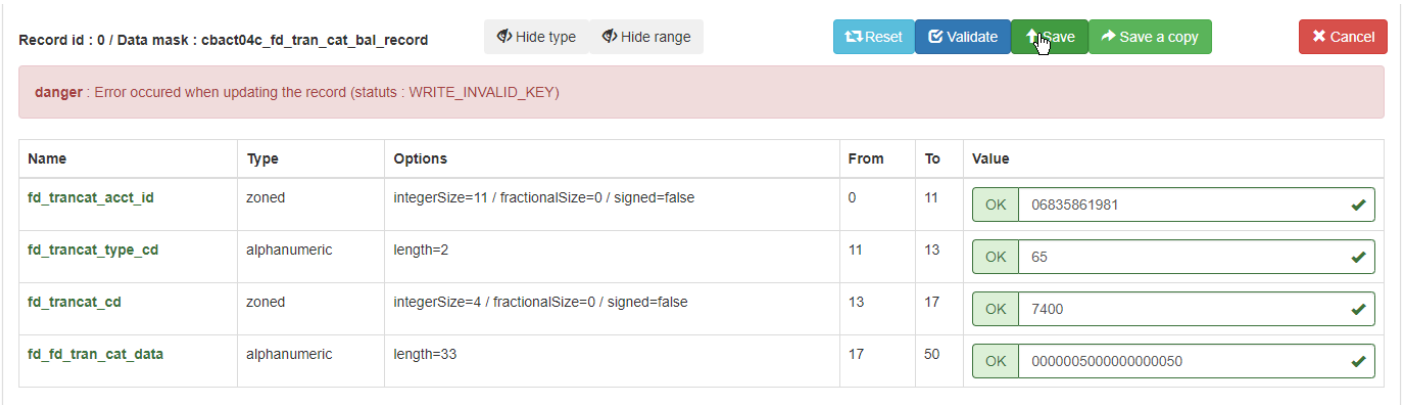
- Le mécanisme de validation vérifie uniquement que la valeur de l'élément de masque est formellement compatible avec le type d'élément de masque. Par exemple, consultez cet échec de validation sur un élément de masque numérique :

- Le mécanisme de validation peut essayer de corriger automatiquement une entrée non valide, en affichant un message d'information en bleu pour indiquer que la valeur a été automatiquement corrigée, en fonction de son type. Par exemple, en saisissant 7XX0 comme valeur numérique dans l'élément de masque numérique : `fd_trncat_cd`

La validation des appels aboutit aux résultats suivants :

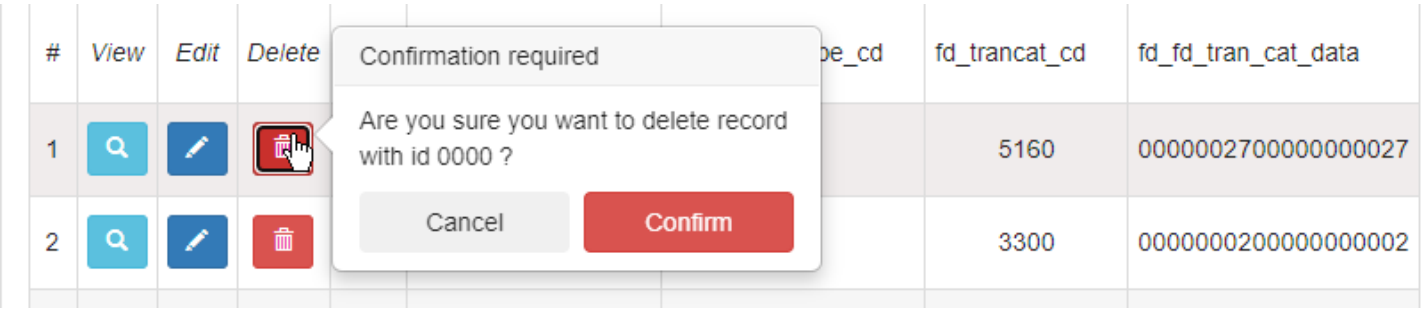


- Le mécanisme de validation ne vérifie pas si la valeur donnée est valide en termes d'intégrité de la clé (si une clé unique est impliquée pour l'ensemble de données donné). Par exemple, malgré le succès de la validation, si les valeurs fournies entraînent une situation de clé non valide ou dupliquée, la persistance échouera et un message d'erreur s'affichera :



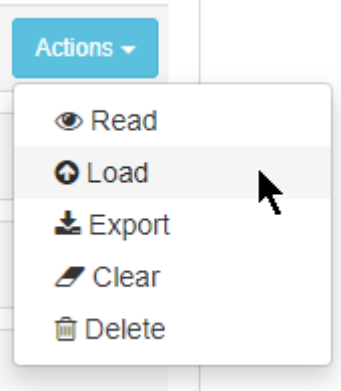
Supprimer un enregistrement :

Pour supprimer un enregistrement, cliquez sur le bouton de la corbeille :

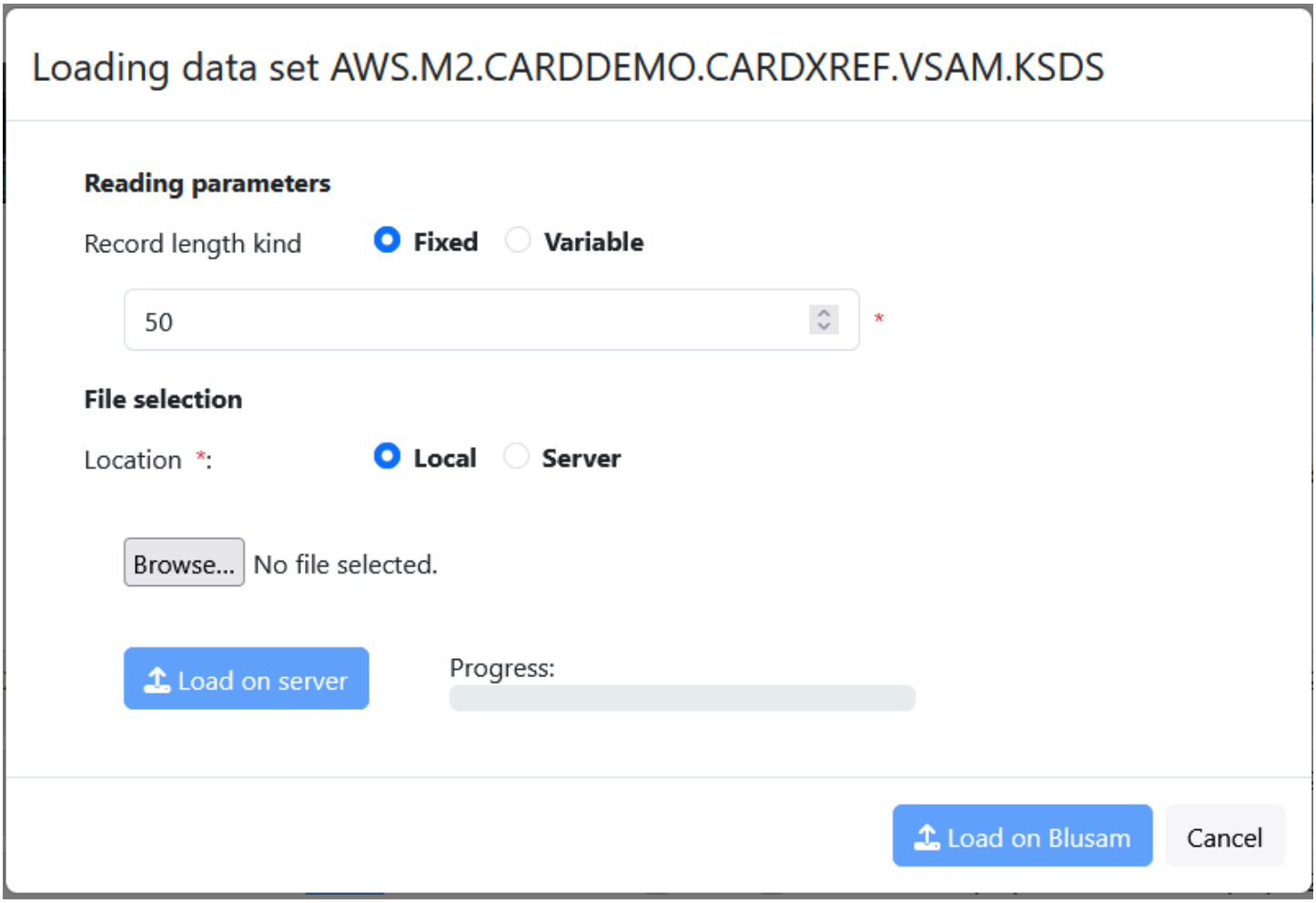


Charger des enregistrements dans un ensemble de données

Pour charger des enregistrements dans un ensemble de données, choisissez Actions, puis Charger.



Une fenêtre contenant les options de chargement apparaît.

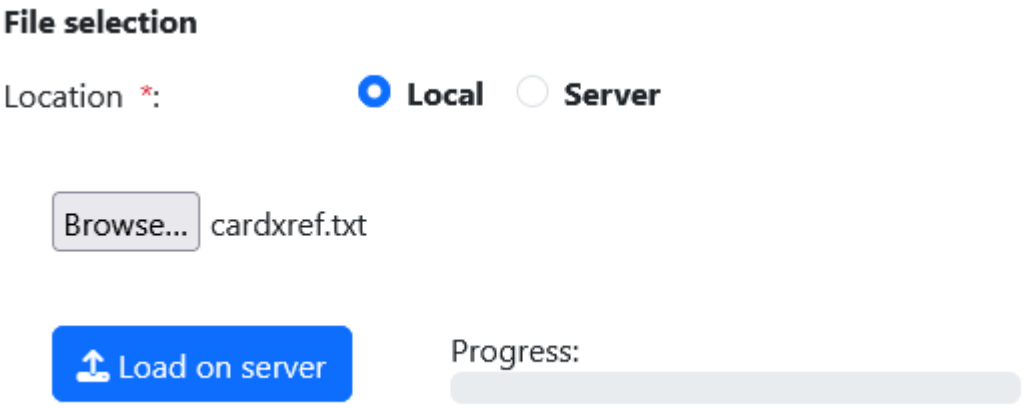


Dans un premier temps, les boutons Load on server et Load on Blusam sont désactivés.

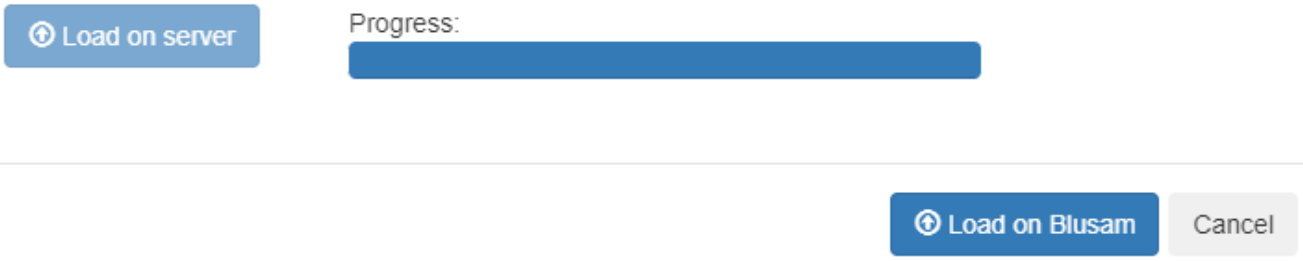
Paramètres de lecture :

- Type de longueur d'enregistrement :
  - Longueur d'enregistrement fixe ou variable : utilisez le bouton radio pour spécifier si l'ancienne exportation des ensembles de données utilise des enregistrements de longueur fixe ou des enregistrements de longueur variable (les enregistrements devraient commencer par des octets RDW). Si vous choisissez Fixe, la longueur de l'enregistrement doit être spécifiée (en octets) sous forme de valeur entière positive dans le champ de saisie. La valeur doit être préremplie à l'aide des informations provenant de l'ensemble de données. Si vous choisissez Variable, le champ de saisie indiqué disparaît.
  - Sélection du fichier :
    - Local : choisissez le fichier d'ensemble de données sur votre ordinateur local à l'aide du sélecteur de fichiers ci-dessous (Remarque : le sélecteur de fichiers utilise les paramètres

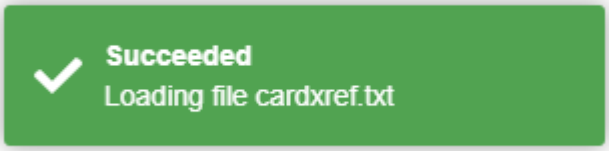
régionaux de votre navigateur pour imprimer ses messages, ici en français, mais l'apparence peut être différente de votre côté, ce qui est normal). Après avoir effectué la sélection, la fenêtre est mise à jour avec le nom du fichier de données et le bouton Charger sur le serveur est activé :



Choisissez Charger sur le serveur. Une fois la barre de progression terminée, le bouton Load on Blusam est activé :

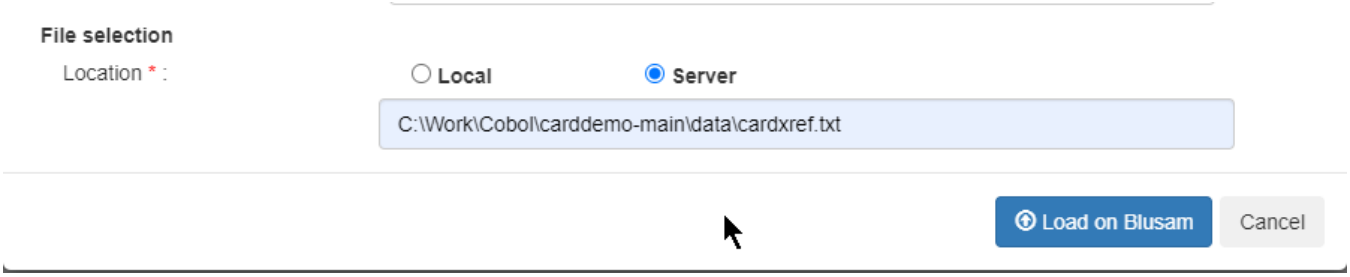


Pour terminer le processus de chargement vers le stockage Blusam, choisissez Load on Blusam. Sinon, sélectionnez Annuler. Si vous choisissez de poursuivre le processus de chargement, une notification apparaîtra dans le coin inférieur droit une fois le processus de chargement terminé :

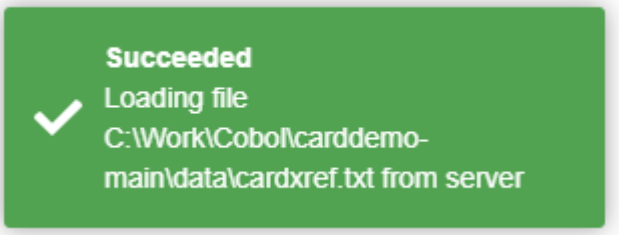


- Serveur : cette option fait apparaître un champ de saisie tandis que le bouton Charger sur le serveur disparaît. Le champ de saisie est l'endroit où vous devez spécifier le chemin vers le fichier d'ensemble de données sur le serveur Blusam (cela suppose que vous avez d'abord

transféré le fichier donné sur le serveur Blusam). Une fois que vous avez spécifié le chemin, Load on Blusam est activé :

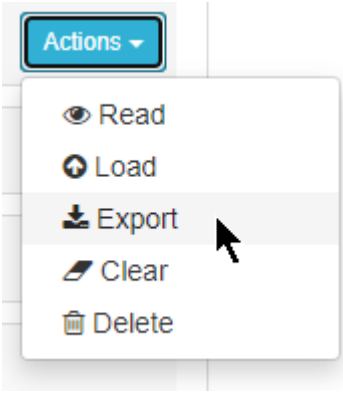


Pour terminer le processus de chargement, choisissez Load on Blusam. Sinon, sélectionnez Annuler. Si vous choisissez de poursuivre le chargement, une notification apparaît une fois le processus de chargement terminé. La notification est différente du chargement depuis le navigateur car elle affiche le chemin du serveur de fichiers de données suivi des mots « du serveur » :

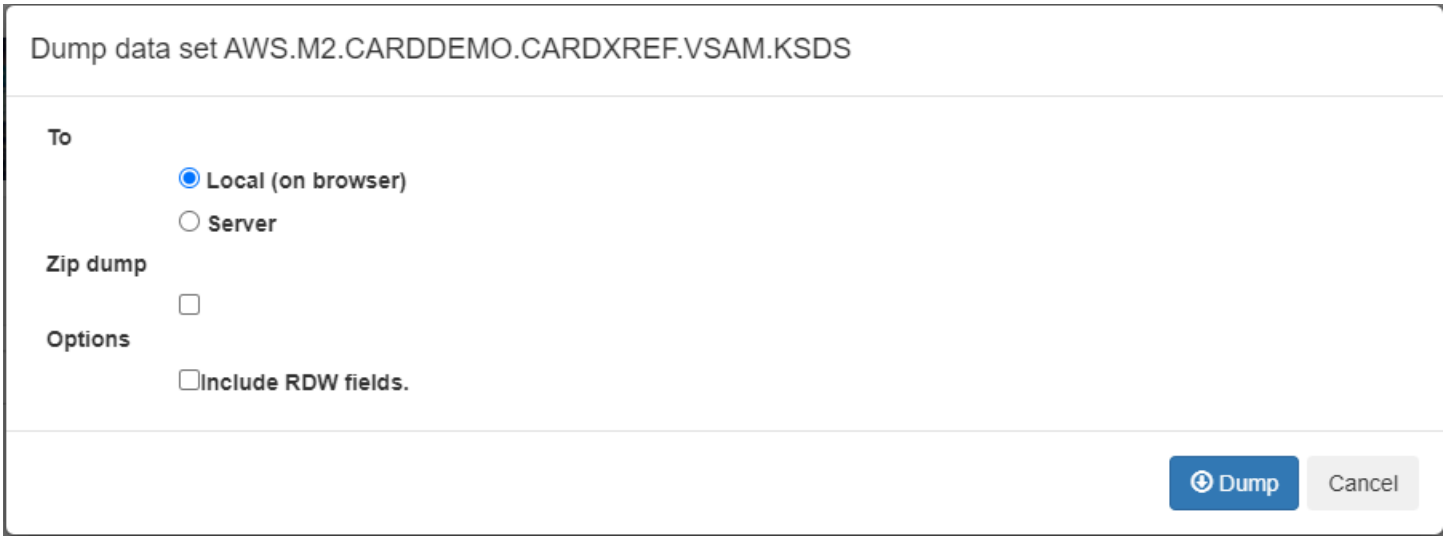


Exportation d'enregistrements à partir d'un ensemble de données

Pour exporter des enregistrements d'ensembles de données, choisissez Actions dans la ligne actuelle du jeu de données, puis sélectionnez Exporter :



La fenêtre contextuelle suivante apparaît.



Options :

À : un bouton radio permet de sélectionner la destination de l'exportation, soit sous forme de téléchargement dans le navigateur (local (sur le navigateur)), soit dans un dossier donné sur le serveur hébergeant l'application BAC. Si vous choisissez d'exporter en utilisant le choix Serveur, un nouveau champ de saisie sera affiché :

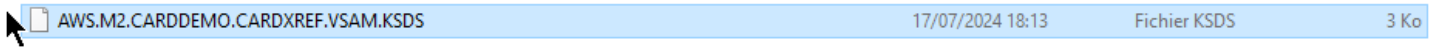
Comme l'indique l'astérisque rouge à droite du champ de saisie, il est obligatoire de fournir un emplacement de dossier valide sur le serveur (le bouton Dump sera inactif tant qu'aucun emplacement de dossier n'a été indiqué).

Pour exporter vers le serveur, vous devez disposer des droits d'accès suffisants au système de fichiers du serveur, si vous prévoyez de manipuler le fichier de jeu de données exporté après l'exportation.

Zip dump : case à cocher qui produit une archive compressée au lieu d'un fichier brut.

Options : Pour inclure un mot descripteur d'enregistrement (RDW) au début de chaque enregistrement de l'ensemble de données exporté dans le cas d'un ensemble de données d'enregistrement de longueur variable, choisissez Inclure les champs RDW.

Pour lancer le processus d'exportation des ensembles de données, choisissez Dump. Si vous choisissez d'exporter vers un navigateur, consultez le dossier de téléchargement pour le fichier d'ensemble de données d'exportation. Le fichier portera le même nom que le jeu de données :

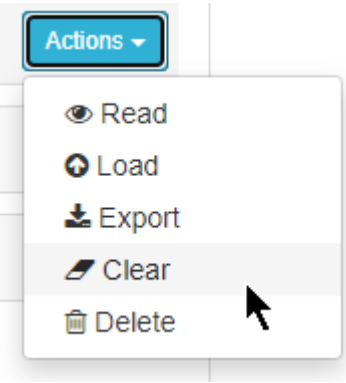


Remarque :

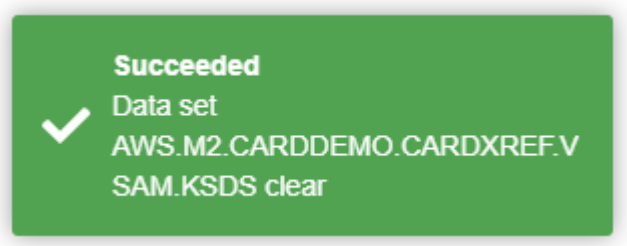
- Pour KSDS, les enregistrements seront exportés selon l'ordre des clés primaires.
- Pour ESDS et RRDS, les enregistrements seront exportés selon l'ordre RBA (Relative Byte Address).
- Pour tous les types d'ensembles de données, les enregistrements seront exportés sous forme de tableaux binaires bruts (aucune conversion n'aura lieu), garantissant ainsi une compatibilité directe avec les plateformes existantes.

Effacement des enregistrements d'un ensemble de données

Pour effacer tous les enregistrements d'un ensemble de données, choisissez Actions, puis Effacer :

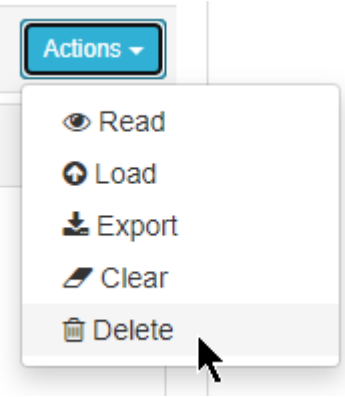


Une fois que tous les enregistrements ont été supprimés d'un ensemble de données, la notification suivante apparaît.

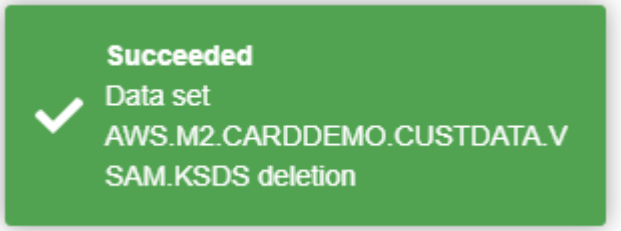


Supprimer un ensemble de données

Pour supprimer un ensemble de données, choisissez Actions, puis Supprimer :



Après avoir supprimé un ensemble de données, la notification suivante apparaît :

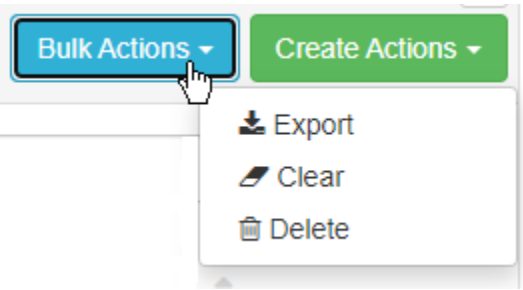


Opérations en vrac

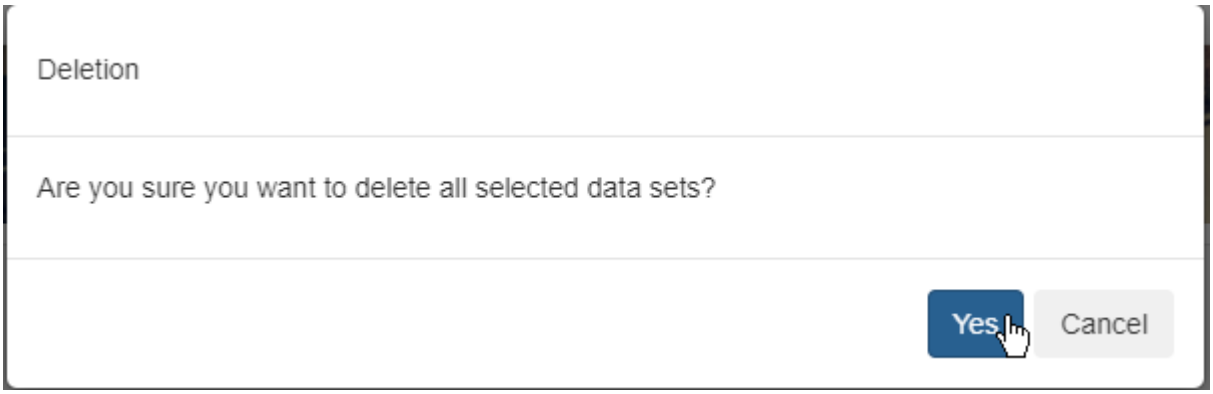
Trois opérations groupées sont disponibles sur les ensembles de données :

- Exportation
- Effacer
- Suppression

Les opérations groupées ne peuvent être appliquées qu'à une sélection d'ensembles de données (au moins un ensemble de données doit être sélectionné) ; la sélection des ensembles de données s'effectue en cochant les cases de sélection à gauche des lignes des ensembles de données, dans le tableau de liste des ensembles de données. La sélection d'au moins un ensemble de données activera la liste déroulante des actions groupées :



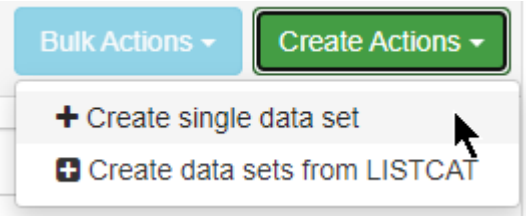
Outre le fait que les actions données s'appliquent à une sélection d'ensembles de données plutôt qu'à un seul, les actions sont similaires à celles décrites ci-dessus. Veuillez donc vous référer à la documentation dédiée aux actions pour plus de détails. Le contenu du texte des fenêtres contextuelles sera légèrement différent pour refléter la nature globale. Par exemple, lorsque vous essayez de supprimer plusieurs ensembles de données, la fenêtre contextuelle se présente comme suit :



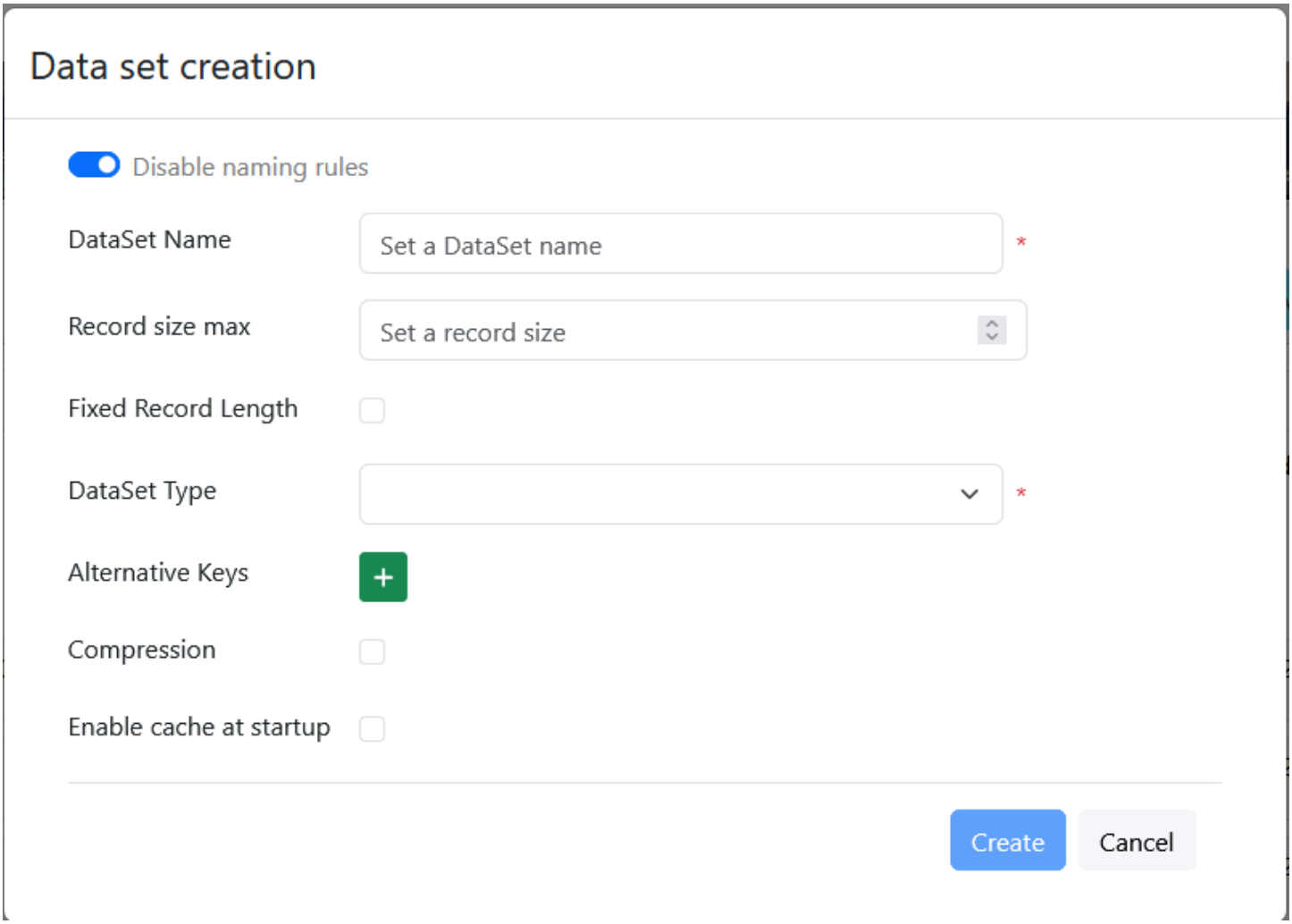
Création d'opérations

Création d'un ensemble de données unique

Choisissez Actions, puis sélectionnez Créer un ensemble de données unique :

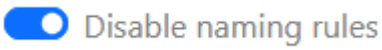


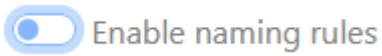
Le formulaire de création de l'ensemble de données sera ensuite affiché sous forme de fenêtre contextuelle :



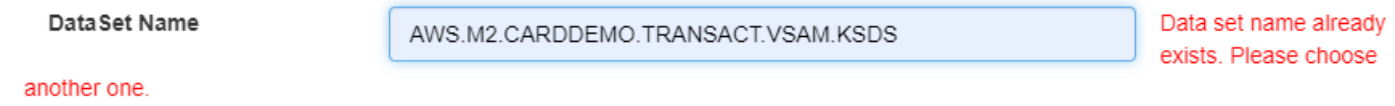
Vous pouvez spécifier les attributs suivants pour la définition de l'ensemble de données :

- Activation et désactivation des règles de dénomination : utilisez le widget à bascule « Désactiver les règles de nommage/Activer les règles de dénomination » pour désactiver et activer les conventions de dénomination des ensembles de données. Nous vous recommandons de laisser le bouton sur la valeur par défaut, avec les règles de dénomination des ensembles de données activées (le widget de bascule doit afficher « Désactiver les règles de dénomination ») :

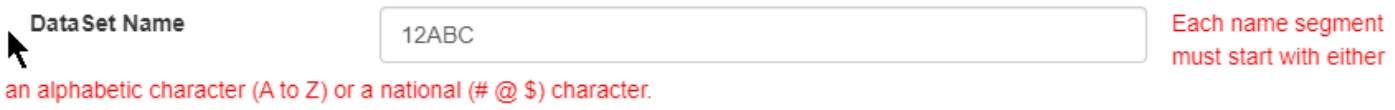
  

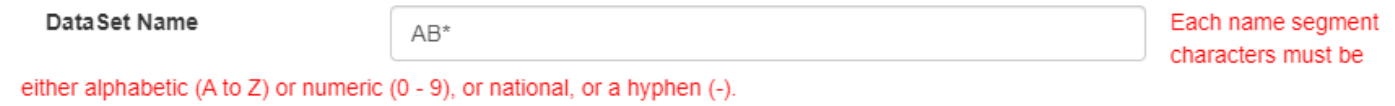
  

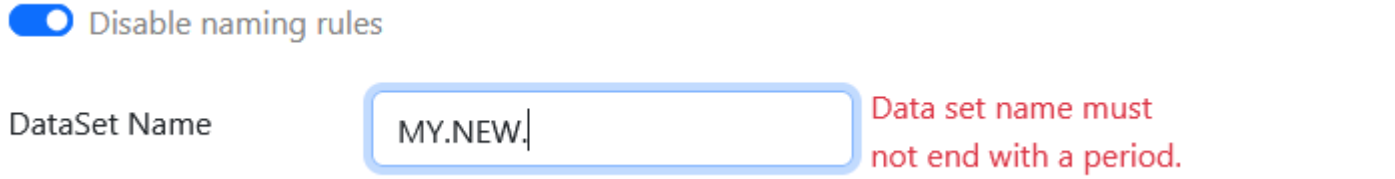
- Nom de l'ensemble de données : nom de l'ensemble de données. Si vous spécifiez un nom déjà utilisé, le message d'erreur suivant s'affiche.



Le nom doit également respecter la convention de dénomination s'il est activé :



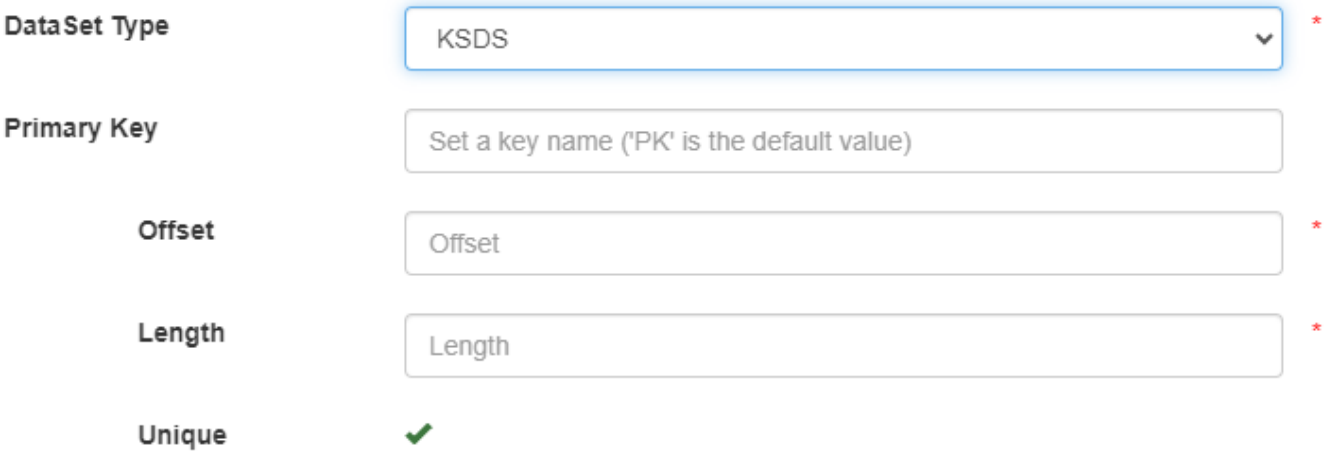




- Taille d'enregistrement maximale : il doit s'agir d'un entier positif représentant la taille d'enregistrement d'un ensemble de données contenant des enregistrements de longueur fixe. Vous pouvez laisser ce champ vide pour les ensembles de données contenant des enregistrements de longueur variable.
- Enregistrement de longueur fixe : case à cocher pour indiquer si la longueur de l'enregistrement est fixe ou variable. Si cette option est sélectionnée, l'ensemble de données contiendra des enregistrements de longueur fixe, sinon la longueur des enregistrements sera variable.

  Lorsque vous importez des données existantes dans un ensemble de données d'enregistrements de longueur variable, les anciens enregistrements fournis doivent contenir le mot RDW (Record Descriptor Word) qui indique la longueur de chaque enregistrement.
- Type de jeu de données : liste déroulante permettant de spécifier le type d'ensemble de données actuel. Les types suivants sont pris en charge.
  - ESDS
  - Gros ESD
  - KSDS

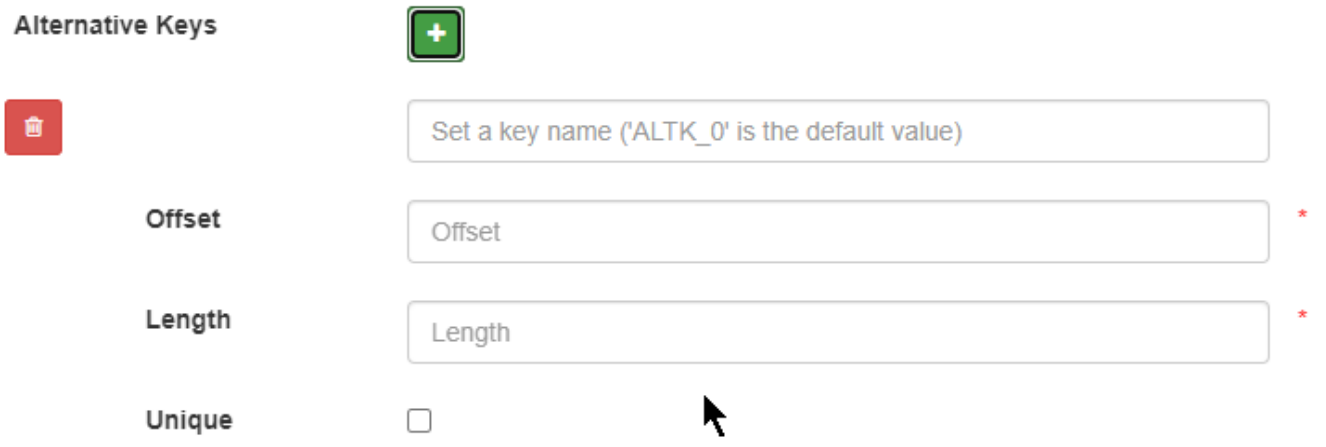
Pour KSDS, vous devez spécifier la clé primaire :



Pour la clé primaire, spécifiez les éléments suivants :

- Nom : Ce champ est facultatif. L'argument par défaut est **PK**.
- Décalage : décalage basé sur 0 de la clé primaire dans l'enregistrement. Le décalage doit être un entier positif. Ce champ est obligatoire.
- Longueur : longueur de la clé primaire. Cette longueur doit être un entier positif. Ce champ est obligatoire.

Pour KSDS et ESDS, vous pouvez éventuellement définir une collection de clés secondaires en cliquant sur le bouton Plus situé devant l'étiquette Alternate Keys. Chaque fois que vous cliquez sur ce bouton, une nouvelle section de définition de clé alternative apparaît dans le formulaire de création du jeu de données :



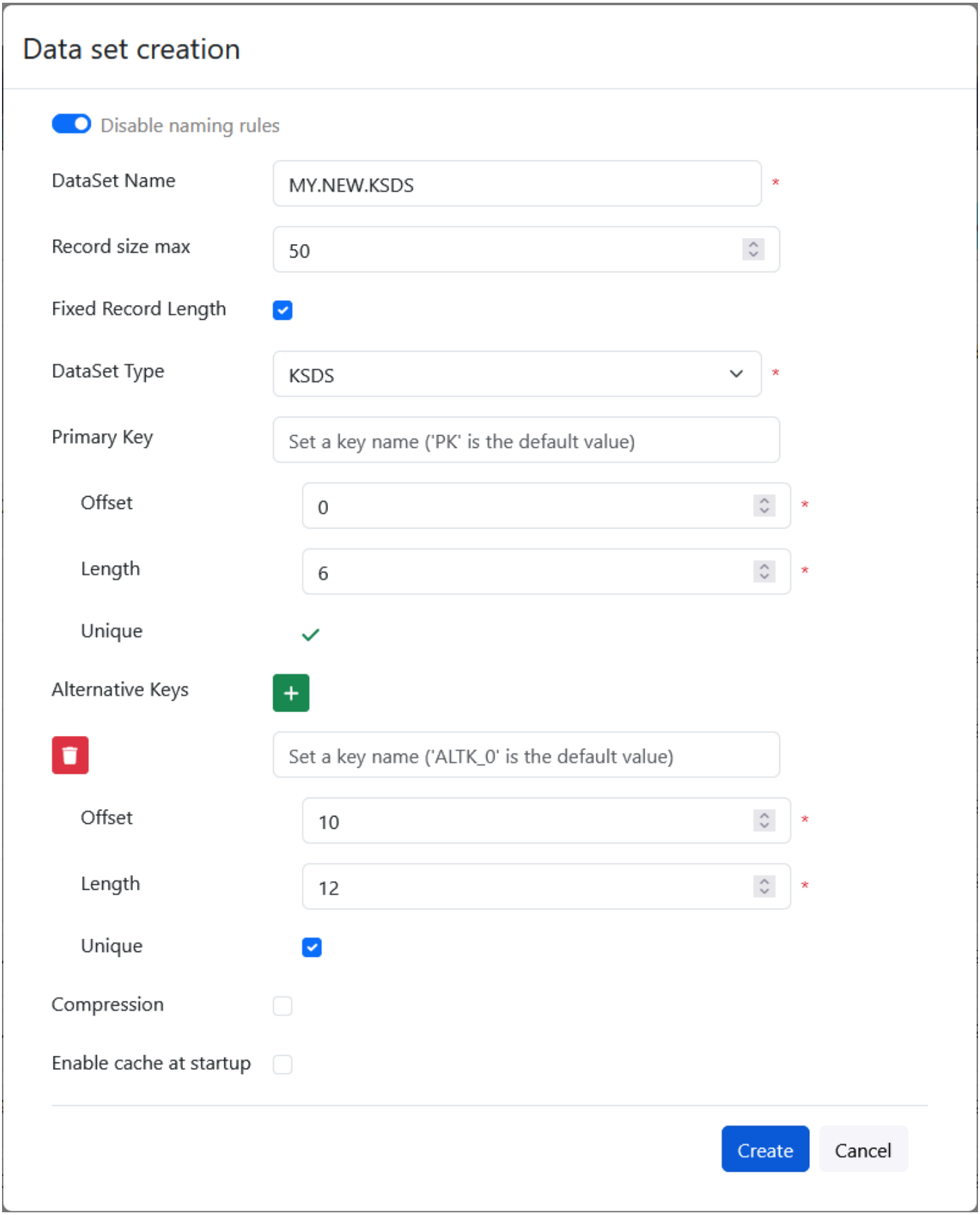
Pour chaque clé alternative, vous devez fournir :

- Nom : Ce champ est facultatif. La valeur par défaut est**`ALTK_#`**, où # représente un compteur auto-incrémenté qui commence à 0.
- Décalage : le décalage basé sur 0 de la clé alternative dans l'enregistrement. Il doit s'agir d'un entier positif. Ce champ est obligatoire.
- Longueur : longueur de la clé alternative. Cette longueur doit être un entier positif. Ce champ est obligatoire.
- Unique : case à cocher pour indiquer si la clé alternative accepte les entrées en double. Si cette option est sélectionnée, la clé alternative sera définie comme unique (aucune saisie de clé en double n'est acceptée). Ce champ est obligatoire.

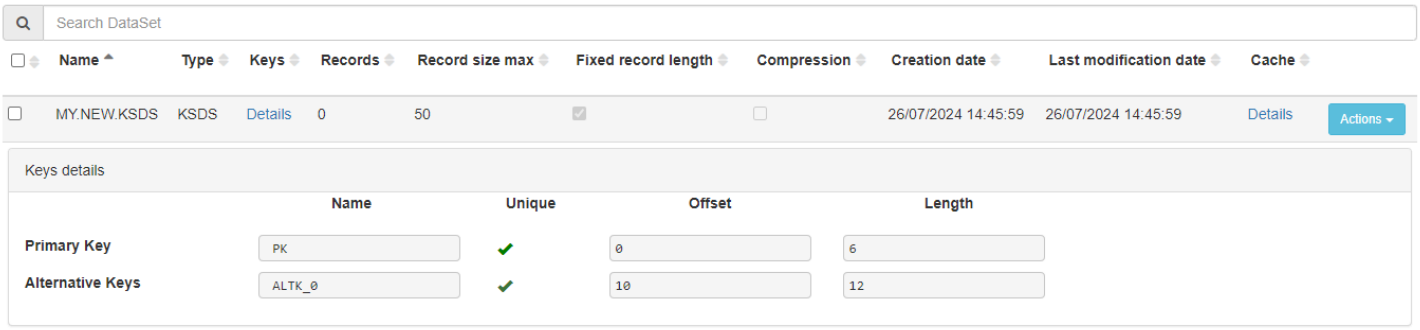
Pour supprimer la définition de clé alternative, utilisez le bouton de la corbeille sur la gauche.

- Compression : case à cocher pour indiquer si la compression sera utilisée pour stocker l'ensemble de données.
- Activer le cache au démarrage : case à cocher pour indiquer si le jeu de données doit être chargé dans le cache au démarrage de l'application.

Après avoir défini les définitions d'attributs, choisissez Create pour continuer :


Data set creation
Disable naming rules
DataSet Name
MY.NEW.KSDS
Record size max
50
Fixed Record Length
DataSet Type
KSDS
Primary Key
Set a key name ('PK' is the default value)
Offset
0
Length
6
Unique
Alternative Keys
Set a key name ('ALTK_0' is the default value)
Offset
10
Length
12
Unique
Compression
Enable cache at startup
Create
Cancel

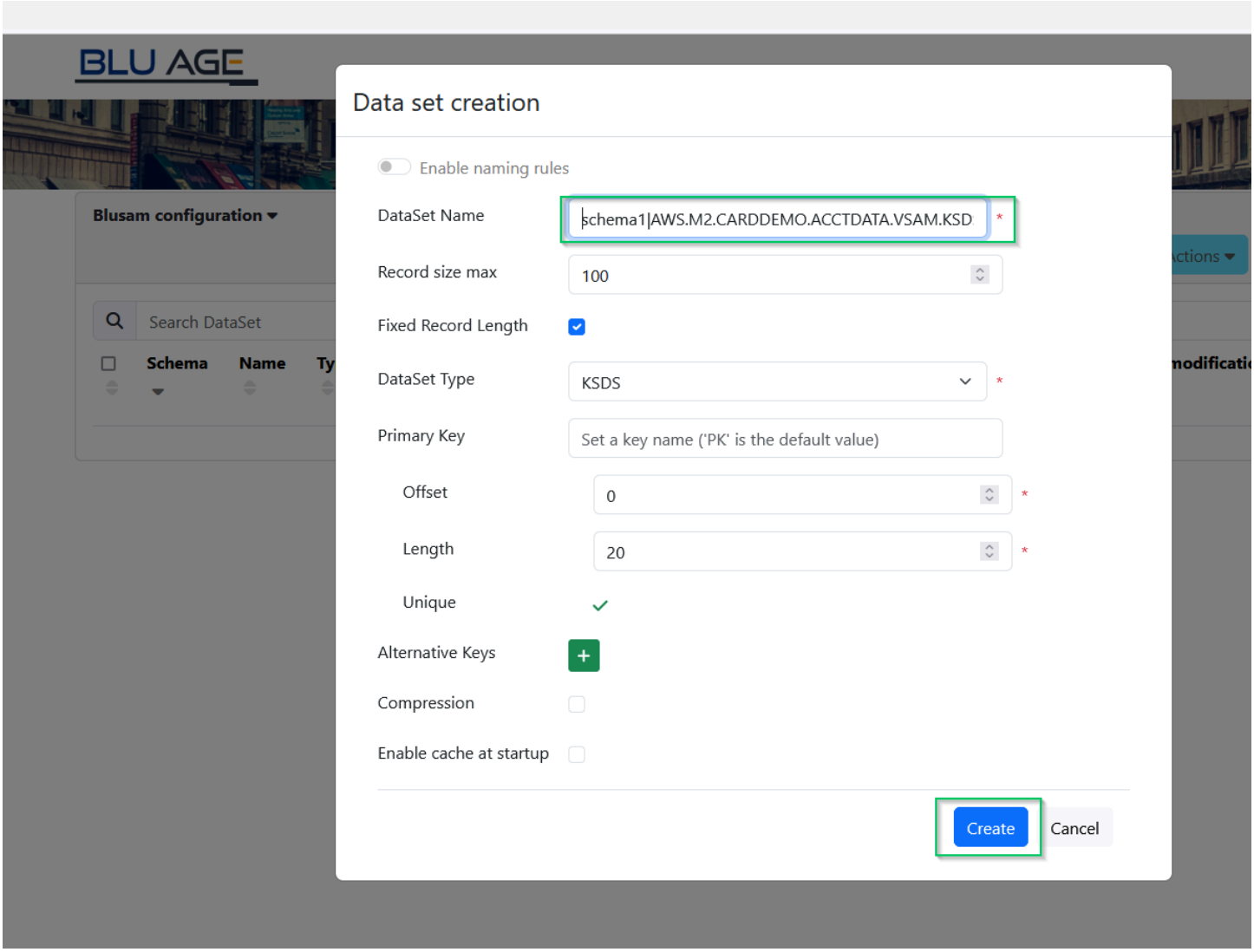
La fenêtre de création sera fermée et la page d'accueil contenant la liste des ensembles de données s'affichera. Vous pouvez consulter les détails de l'ensemble de données nouvellement créé.



Création d'un ensemble de données unique en mode multi-schémas

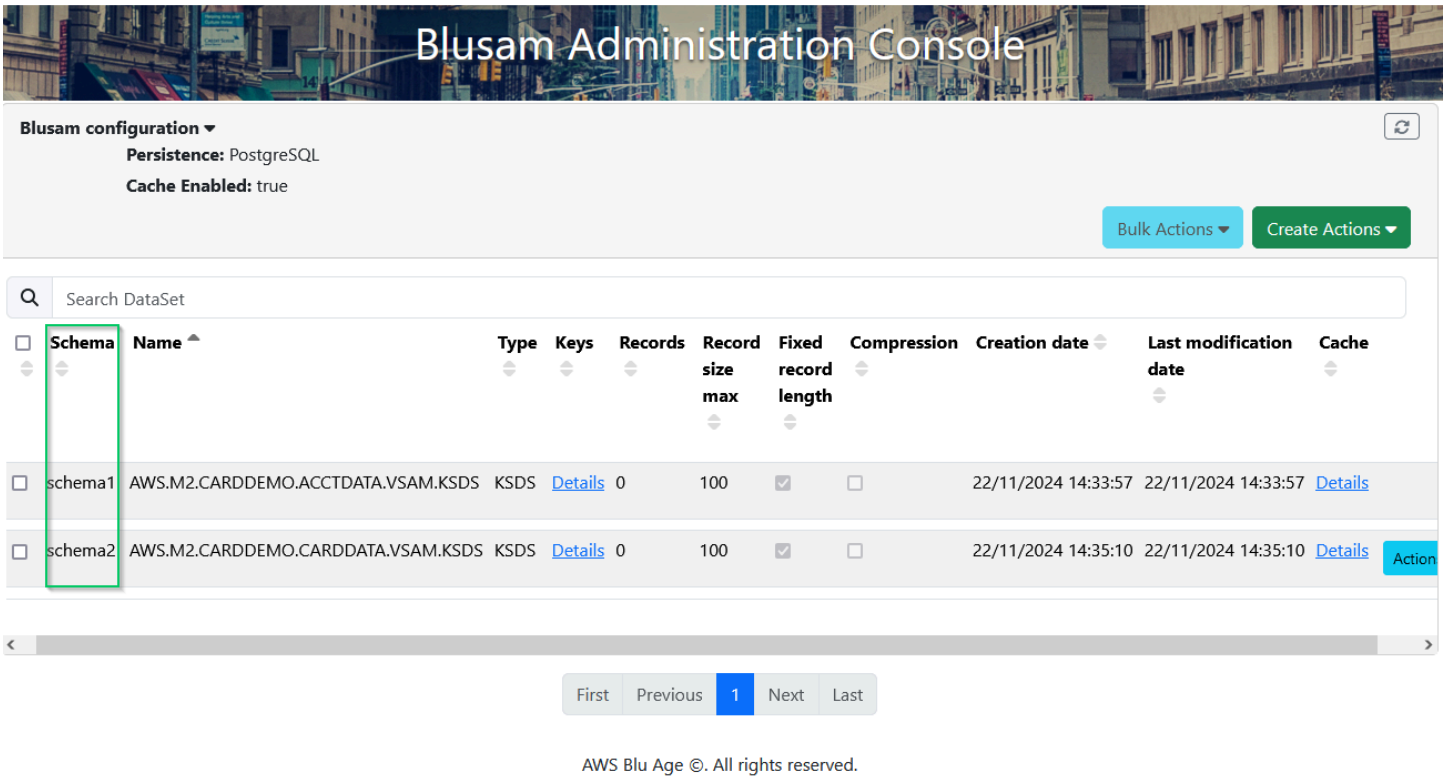
Un ensemble de données peut être créé en mode multi-schémas en préfixant le nom du jeu de données par le nom du schéma suivi d'un symbole en forme de tube (|) (par exemple,`schema1|AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS`).

> **Note**
>
> Le schéma utilisé pour créer l'ensemble de données doit être spécifié dans la `application-main.yml` configuration. Pour de plus amples informations, veuillez consulter the section called “Propriétés de configuration multi-schémas ”.



Si aucun préfixe de schéma n'est fourni, l'ensemble de données sera créé dans le schéma par défaut spécifié dans l'URL de la source de données Blusam dans la configuration de la source de données Blusam. Si aucun schéma n'est spécifié dans l'URL de la source de données Blusam, le schéma « public » est utilisé par défaut.

> **Note**
>
> En mode multi-schémas, la console BAC affiche les informations de schéma de l'ensemble de données dans la première colonne.



Création d'ensembles de données à partir de LISTCAT

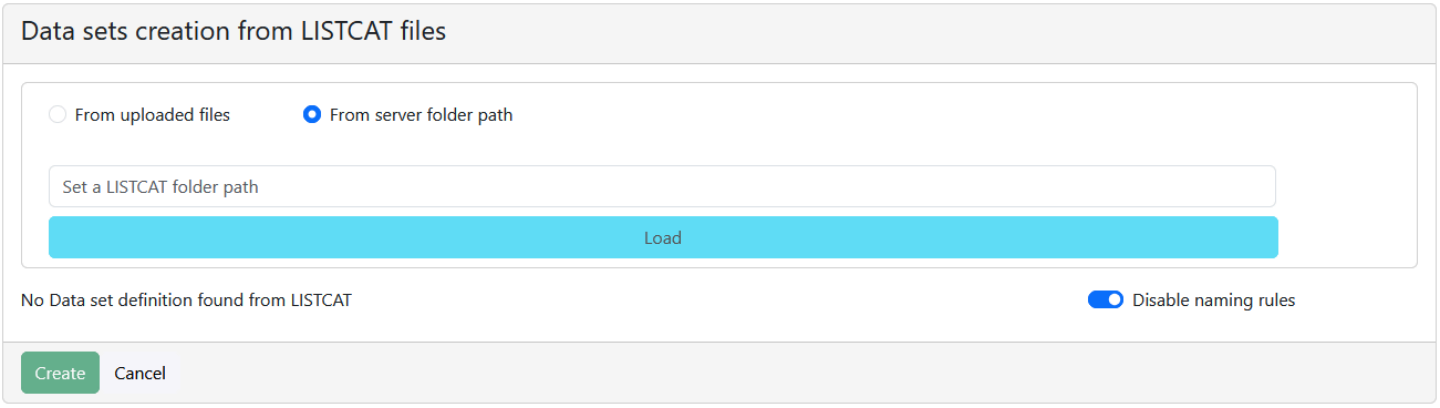
Cette fonctionnalité permet de tirer parti des fichiers LISTCAT JSON créés pendant le processus de BluAge transformation à l'aide BluInsights de Transformation Center à la suite de l'analyse de l'exportation LISTCAT à partir des anciennes plateformes : les exportations LISTCAT sont analysées et transformées en fichiers JSON contenant les définitions des ensembles de données (noms, type d'ensemble de données, définitions des clés et si la longueur de l'enregistrement est fixe ou variable).

Les fichiers JSON LISTCAT permettent de créer directement des ensembles de données sans avoir à saisir manuellement toutes les informations requises pour les ensembles de données. Vous pouvez également créer une collection d'ensembles de données directement au lieu de devoir les créer un par un.

Si aucun fichier LISTCAT JON n'est disponible pour votre projet (par exemple, parce qu'aucun fichier d'exportation LISTCAT n'était disponible au moment de la transformation), vous pouvez toujours en créer un manuellement, à condition de respecter le format LISTCAT JSON détaillé en annexe.

Dans la liste déroulante Créer des actions, choisissez Créer des ensembles de données à partir de LISTCAT.

La page dédiée suivante sera affichée :



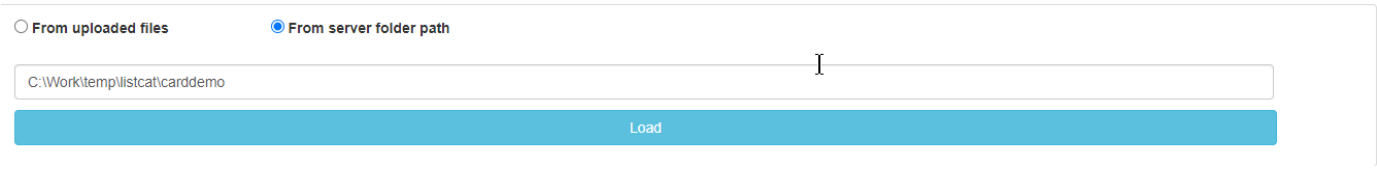
À ce stade, le bouton Charger est désactivé, ce qui est normal.

Utilisez les boutons radio pour spécifier la manière dont vous souhaitez fournir les fichiers JSON LISTCAT. Deux options s'offrent à vous :

- Vous pouvez utiliser votre navigateur pour télécharger les fichiers JSON.
- Vous pouvez sélectionner les fichiers JSON dans un dossier situé sur le serveur. Pour choisir cette option, vous devez d'abord copier les fichiers JSON dans le chemin de dossier indiqué sur le serveur avec les droits d'accès appropriés.

Pour utiliser des fichiers JSON sur le serveur

1. Définissez le chemin du dossier sur le serveur, en pointant sur le dossier contenant les fichiers LISTCAT JSON :

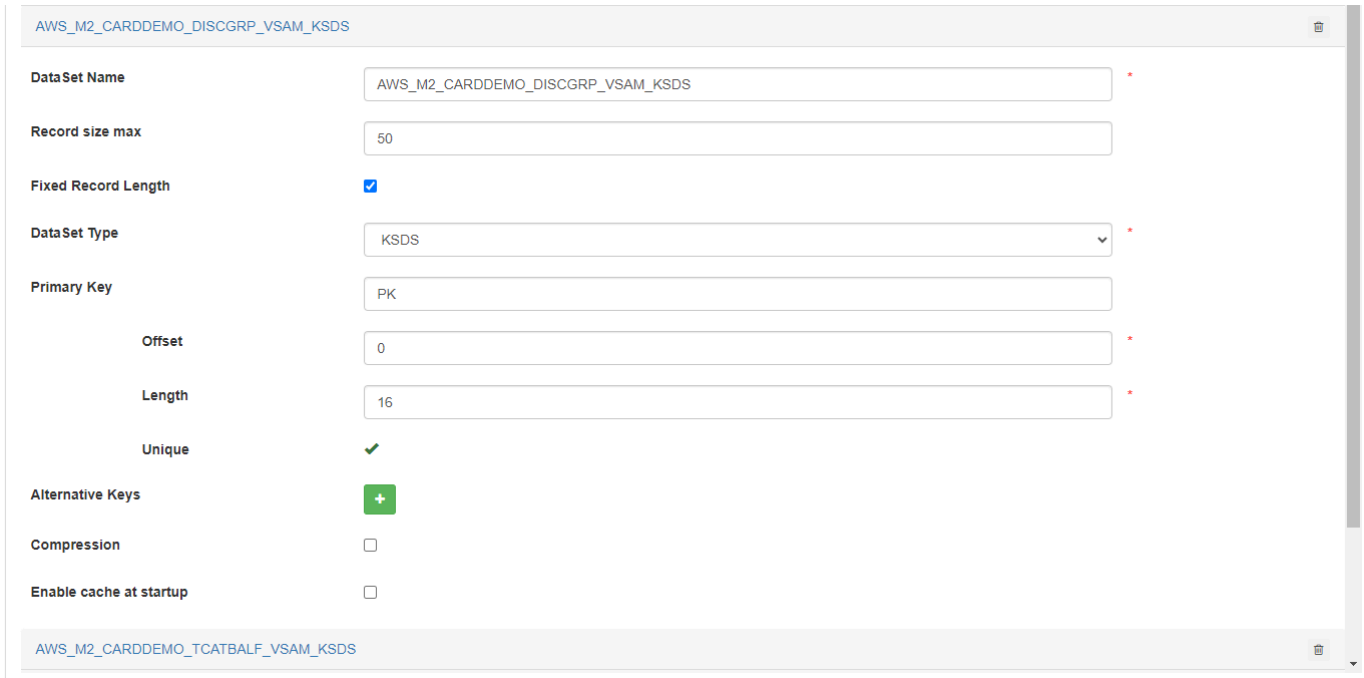


2. Cliquez sur le bouton Charger. Toutes les définitions d'ensembles de données reconnues seront répertoriées dans un tableau :

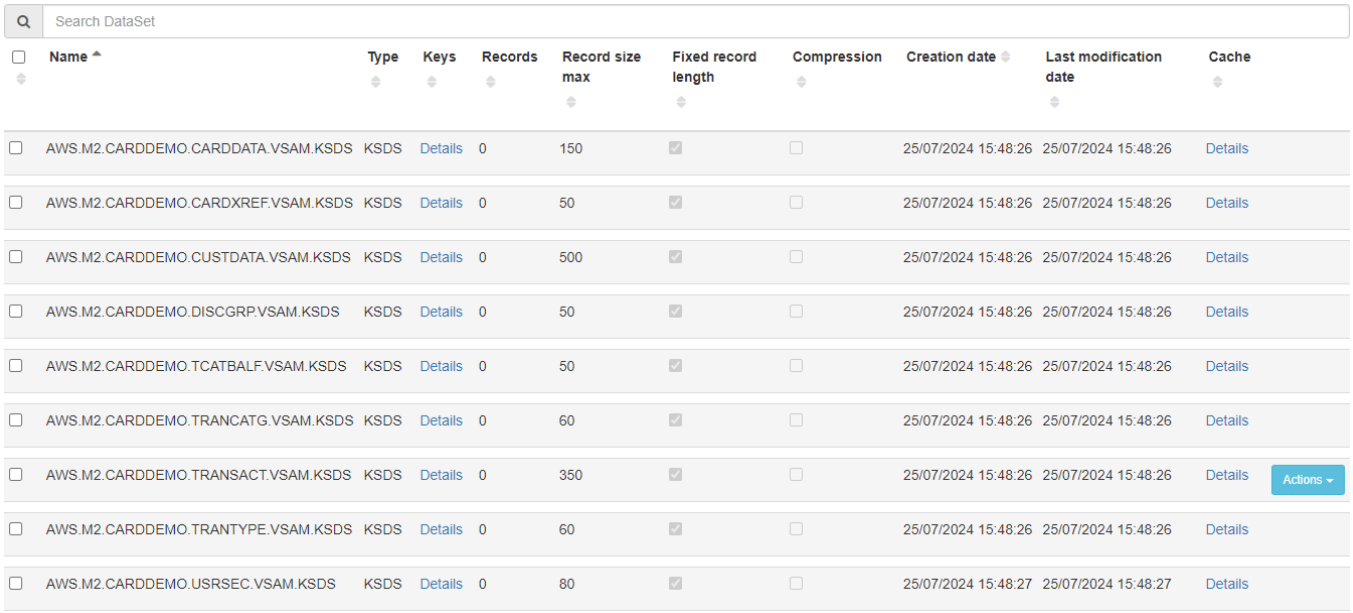
Chaque ligne représente une définition d'ensemble de données. Vous pouvez utiliser le bouton corbeille pour supprimer une définition d'ensemble de données de la liste.

> **Important**
>
> Le retrait de la liste est immédiat, sans message d'avertissement.

3. Le nom sur la gauche est un lien. Vous pouvez le choisir pour afficher ou masquer les détails de la définition de l'ensemble de données, qui est modifiable. Vous pouvez modifier librement la définition, en commençant par le fichier JSON analysé.

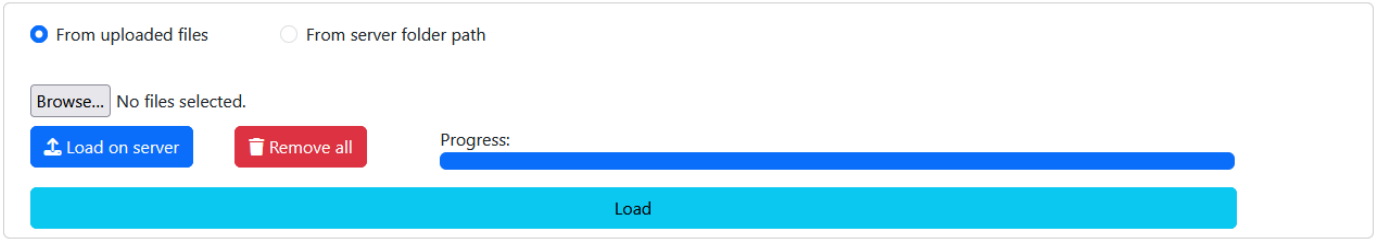


4. Pour créer tous les ensembles de données, choisissez Create. Tous les ensembles de données seront créés et affichés sur la page de résultats des ensembles de données. Les ensembles de données nouvellement créés comporteront tous 0 enregistrement.



Pour télécharger des fichiers sur le serveur

1. Cette option est similaire à l'utilisation des fichiers depuis le chemin du dossier du serveur, mais dans ce cas, vous devez d'abord télécharger les fichiers à l'aide du sélecteur de fichiers. Sélectionnez tous les fichiers à télécharger depuis votre ordinateur local, puis choisissez Charger sur le serveur.



2. Lorsque la barre de progression atteint la fin, tous les fichiers ont été correctement téléchargés sur le serveur et le bouton Charger est activé. Cliquez sur le bouton Charger et utilisez les définitions des ensembles de données découverts, comme expliqué précédemment.

Format JSON LISTCAT

Le format LISTCAT JSON est défini par les attributs suivants :

- « CatalogiD » facultatif : identifiant du catalogue existant sous forme de chaîne, ou « par défaut » pour le catalogue par défaut.
- « identifiant » : le nom de l'ensemble de données, sous forme de chaîne.
- « isIndexed » : indicateur booléen pour indiquer KSDS : true pour KSDS, false dans le cas contraire.
- « isLinear » : indicateur booléen pour indiquer l'ESDS : true pour ESDS, false dans le cas contraire.
- « isRelative » : un indicateur booléen pour indiquer le RRDS : vrai pour RRDS, faux dans le cas contraire
- Remarque : « IsIndexed », « IsLinear » et « IsRelative » s'excluent mutuellement.
- « isFixedLength Record » : indicateur booléen : défini sur true si des données d'enregistrements de longueur fixe sont définies, sur false dans le cas contraire.
- « avgRecordSize « : Taille d'enregistrement moyenne en octets, exprimée sous la forme d'un entier positif.
- « maxRecordSize « : Taille maximale de l'enregistrement en octets, exprimée sous forme d'entier. Doit être égal à avgRecordSize pour une taille d'enregistrement de longueur fixe.
- pour KSDS uniquement : définition de clé primaire obligatoire (en tant qu'objet imbriqué)
  - étiqueté « PrimaryKey »
  - « offset » : décalage de base de 0 octets pour la clé primaire de l'enregistrement.
  - « length » : longueur en octets de la clé primaire.
  - « unique » : doit être défini sur true pour la clé primaire.
- pour KSDS/ESDS, collection de clés secondaires (sous forme de collection d'objets imbriqués) :
  - étiqueté « AlternateKeys »
  - Pour chaque clé alternative :
    - « offset » : décalage de base de 0 octets pour la clé alternative de l'enregistrement.
    - « length » : longueur en octets de la clé alternative.
    - « unique » : doit être défini sur true pour la clé alternative, si la clé n'accepte pas les doublons, sur false dans le cas contraire.
- si aucune clé alternative n'est présente, fournissez une collection vide :

```
alternateKeys: []
```

Voici un exemple de fichier JSON KSDS LISTCAT.

```
{
  "catalogId": "default",
  "identifier": "AWS_M2_CARDDEMO_CARDXREF_VSAM_KSDS",
  "isIndexed": true,
  "isLinear": false,
  "isRelative": false,
  "isFixedLengthRecord": true,
  "avgRecordSize": 50,
  "maxRecordSize": 50,
  "primaryKey": {
    "offset": 0,
    "length": 16,
    "unique": true
  },
  "alternateKeys": [
    {
      "offset": 25,
      "length": 11,
      "unique": false
    }
  ]
}
```

Configurer la configuration pour AWS Blu Age Runtime

Le AWS Blu Age Runtime et le code client sont des applications Web utilisant le framework Spring Boot. Il tire parti des fonctionnalités de Spring pour fournir une configuration, avec plusieurs emplacements possibles et règles de priorité. Il existe également des règles de priorité similaires pour fournir de nombreux autres fichiers, tels que les scripts Groovy, SQL, etc.

Le AWS Blu Age Runtime contient également des applications Web facultatives supplémentaires, qui peuvent être activées si nécessaire.

Rubriques

- Principes de base de configuration des applications
- Priorité des applications
- JNDI pour bases de données
- AWS Les secrets de Blu Age Runtime
- Autres fichiers (groovy, sql, etc.)

- Application Web supplémentaire
- Activer les propriétés de AWS Blu Age Runtime
- Propriétés du cache Redis disponibles dans AWS Blu Age Runtime
- Configuration de la sécurité pour les applications Gapwalk

Principes de base de configuration des applications

La méthode par défaut pour gérer la configuration des applications consiste à utiliser des fichiers YAML dédiés à fournir dans le `config` dossier du serveur d'applications. Il existe deux principaux fichiers de configuration YAML :

- `application-main.yaml`
- `application-`*`profile`*`.yaml`(où *`profile`* la valeur est configurée lors de la génération de l'application).

Le premier fichier configure le framework, c'est-à-dire`Gapwalk-application.war`, tandis que le second est destiné à des options supplémentaires spécifiques à l'application cliente. Cela fonctionne avec l'utilisation de profils Spring : l'application Gapwalk utilise le `main` profil, tandis que l'application cliente utilise le *`profile`* profil.

L'exemple suivant montre un fichier YAML principal typique.

```
########################################
##### JICS datasource configuration #####
########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver
    url: jdbc:postgresql://localhost/jics
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource

#####################################################
##### Embedded Bluesam datasource configuration #####
#####################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver
    url : jdbc:postgresql://localhost/bluesam
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource

########################################
##### Embedded Bluesam configuration #####
########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql #pgsql, mssql, xodus...
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
```

L'exemple suivant montre un fichier YAML client typique.

```
# Logback context logger integration.
logging.config : classpath:logback-[redacted].xml
# Limits Spring logger output.
logging.level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN
logging.level.org.springframework.statemachine : WARN
# If the datasource support mode is not static-xa, spring JTA transactions autoconfiguration must me disabled
spring.jta.enabled : false

spring:
  aws:
    client:
      datasources:
        names: primary
        primary:
          secret: arn:aws:secretsmanager:[redacted]

spring.jta.atomikos.datasource.primary.unique-resource-name: primary
spring.jta.atomikos.datasource.primary.xa-data-source-class-name: org.postgresql.xa.PGXADataSource
spring.jta.atomikos.datasource.primary.maxPoolSize: 20
spring.jta.atomikos.datasource.primary.autoCommit: false
```

Pour plus d'informations sur le contenu des fichiers YAML, consultezActiver les propriétés de AWS Blu Age Runtime.

Priorité des applications

Pour ces fichiers de configuration, les règles de priorité Spring s'appliquent. Notamment :

- Le fichier `application-main` YAML apparaît dans le fichier war principal de Gapwalk avec les valeurs par défaut, et celui du `config` dossier le remplace.
- La même chose doit être faite pour la configuration de l'application cliente
- Des paramètres supplémentaires peuvent être transmis sur la ligne de commande au moment du lancement du serveur. Ils remplaceraient ceux de YAML.

Pour plus d'informations, consultez la documentation officielle de Spring Boot.

JNDI pour bases de données

La configuration de la base de données peut être fournie avec JNDI dans le fichier context.xml de Tomcat. Toute configuration de ce type remplacerait celle de YAML. Mais attention, son utilisation ne permettra pas d'encapsuler vos informations d'identification dans un gestionnaire secret (voir ci-dessous).

L'exemple suivant montre des exemples de configurations pour les JICS et les BluSam bases de données.

```
<Resource auth="Container" driverClassName="org.postgresql.Driver" initialSize="0"
 maxIdle="5"
   maxOpenPreparedStatements="-1" maxTotal="10" maxWaitMillis="-1" name="jdbc/jics"
   poolPreparedStatements="true" testOnBorrow="false" type="javax.sql.DataSource"
   url="jdbc:postgresql://XXXX.rds.amazonaws.com:5432/XXXX" username="XXXX"
 password="XXXX" />
```

jdbc/jics

Ce serait `jdbc/jics` pour la base de données JICS et `jdbc/bluesam` (attention au « e ») pour la base de données blusam.

url="jdbc:postgresql : //xxxx.rds.amazonaws.com:5432/XXXX » Nom d'utilisateur = « XXXX » Mot de passe = « XXXX »

URL, nom d'utilisateur et mot de passe de la base de données.

AWS Les secrets de Blu Age Runtime

Certaines configurations de ressources contenant des informations d'identification peuvent être davantage sécurisées à l'aide de AWS secrets. L'idée est de stocker les données critiques dans un AWS secret et de faire référence au secret dans la configuration YAML afin que le contenu secret soit récupéré à la volée au démarrage d'Apache Tomcat.

Secrets pour Aurora

La configuration de la base de données Aurora (pour JICS, Blusam, la base de données client, etc.) utilisera le secret de base de données intégré, qui remplira automatiquement tous les champs pertinents à partir de la base de données correspondante.

> **Note**
>
> La `dbname` clé est facultative, selon la configuration de votre base de données, elle entrera dans le secret ou non. Vous pouvez l'y ajouter manuellement ou en fournissant le nom au fichier YAML.

Autres secrets

Les autres secrets concernent les ressources dotées d'un mot de passe unique (notamment les caches Redis protégés par mot de passe). Dans ce cas, l'autre type de secret doit être utilisé.

Références YAML aux secrets

Ils `application-main.yml` peuvent référencer l'ARN secret pour différentes ressources :

Base de données JICS

Informations d'identification de base de données JICS avec `spring.aws.jics.db.secret`

```
spring:
   aws:
     jics:
       db:
         dbname: jics
         secret: arn:aws:secretsmanager:XXXX
```

Clés secrètes de base de données JICS prises en charge :

Clé secrète	Description de la clé secrète
hôte	Le nom de l'hôte
port	Le port
dbname	Le nom de la base de données
nom d'utilisateur	Le nom d'utilisateur
mot de passe	Le mot de passe
engine	Moteur de base de données : Postgres, Oracle, Db2, Microsoft SQL Server
Schéma actuel	Schéma spécifique à utiliser (support DB2 uniquement)
Connexion SSL	S'il faut utiliser une connexion SSL (support DB2 uniquement)
sslTrustStoreLocation	L'emplacement du truststore sur le client (support DB2 uniquement)
sslTrustStoreMot de passe	Le mot de passe du truststore sur le client (support DB2 uniquement)

> **Note**
>
> Le nom de la base de données est fourni soit dans le secret, soit dans la référence `spring.aws.jics.db.dbname` yaml.

Base de données Blusam

Informations d'identification de la base de données Blusam avec `spring.aws.client.bluesam.db.secret`

```
spring:
```

```
  aws:
    client:
      bluesam:
        db:
          dbname: bluesam
          secret: arn:aws:secretsmanager:XXXX
```

Clés secrètes de base de données Blusam prises en charge :

Clé secrète	Description de la clé secrète
hôte	Le nom de l'hôte
port	Le port
dbname	Le nom de la base de données
nom d'utilisateur	Le nom d'utilisateur
mot de passe	Le mot de passe
engine	Moteur de base de données : Postgres

> **Note**
>
> Le nom de la base de données est fourni soit dans le secret, soit dans la référence `spring.aws.client.bluesam.db.dbname` yaml.

Base de données clients

Le client `application-profile.yml` peut référencer l'ARN secret de la base de données client. Cela nécessite une propriété supplémentaire pour répertorier les noms `spring.aws.client.datasources.names` des sources de données. Pour chaque nom de source de données, `ds_name` spécifiez l'ARN secret dans la propriété suivante :`spring.aws.client.datasources.ds_name.secret`. Exemple :

```
spring:
  aws:
```

```
    client:
      datasources:
        names: primary,host
        primary:
          secret: arn:aws:secretsmanager:XXXX
        host:
          dbname: hostdb
          secret: arn:aws:secretsmanager:XXXX
```

noms : principal, hôte :

Exemple avec deux sources de données clientes nommées primary et host, chacune avec sa base de données et ses informations d'identification.

nom de base de données : hostdb :

Dans cet exemple, le nom de la base de données « hôte » ne figure pas dans le secret et est fourni ici, tandis que pour la base de données « principale », il figure dans le secret.

Clés secrètes de base de données client prises en charge :

Clé secrète	Description de la clé secrète
hôte	Le nom de l'hôte
port	Le port
dbname	Le nom de la base de données
nom d'utilisateur	Le nom d'utilisateur
mot de passe	Le mot de passe
engine	Moteur de base de données : Postgres, Oracle, Db2, Microsoft SQL Server
Schéma actuel	Schéma spécifique à utiliser (support DB2 uniquement)
Connexion SSL	S'il faut utiliser une connexion SSL (support DB2 uniquement)

Clé secrète	Description de la clé secrète
sslTrustStoreLocation	L'emplacement du truststore sur le client (support DB2 uniquement)
sslTrustStoreMot de passe	Le mot de passe du truststore sur le client (support DB2 uniquement)

Base de données des utilitaires PGM

Ils `application-utility-pgm.yml` peuvent référencer l'ARN secret pour diverses ressources.

- `spring.aws.client.datasources.primary`
  - `secret`

    ARN secret pour la base de données de l'application.

Type : chaîne

- `type`

Nom complet de l'implémentation du pool de connexions à utiliser.

Type : chaîne

Par défaut : `com.zaxxer.hikari.HikariDataSource`

- `spring.aws.client.utility.pgm.datasources`
  - `names`

Liste des noms de sources de données.

Type : chaîne

- `dsname`
  - `dbname`

Nom de l'hôte.

Type : chaîne

- `secret`

ARN secret de la base de données hôte.

Type : chaîne

- `type`

Nom complet de l'implémentation du pool de connexions à utiliser.

Type : chaîne

Par défaut : `com.zaxxer.hikari.HikariDataSource`

Pour un secret contenant plusieurs sources de données :

```
spring:
   aws:
     client:
       primary:
         secret: arn:aws:secretsmanager:XXXX
         type: dataSourceType
       utility:
         pgm:
           datasources:
             names: dsname1,dsname2,dsname3
               dsname1:
                 dbname: dbname1
                 secret: arn:aws:secretsmanager:XXXX
                 type: dataSourceType
               dsname2:
                 dbname: dbname2
                 secret: arn:aws:secretsmanager:XXXX
                 type: dataSourceType
               dsname3:
                 dbname: dbname3
                 secret: arn:aws:secretsmanager:XXXX
                 type: dataSourceType
```

Aucune clé secrète prise en charge par XA

- moteur (postgres/oracle/db2/mssql)
- port
- dbname
- Schéma actuel
- nom d'utilisateur
- mot de passe
- url
- Connexion SSL
- sslTrustStoreLocation
- sslTrustStoreMot de passe

Car `postgres` seules la valeur de la clé `sslMode` secrète (`disable/allow/prefer/require/verify-ca/verify-full`) et la propriété `spring.aws.rds.ssl.cert-path` YAML permettent de se connecter avec SSL.

Clés secrètes prises en charge par XA

Si la base de données client utilise XA, les propriétés subxa sont prises en charge par le biais de valeurs secrètes.

- hôte
- port
- dbname
- Schéma actuel
- nom d'utilisateur
- mot de passe
- url
- Connexion SSL (vrai/faux)
- sslTrustStoreLocation
- sslTrustStoreMot de passe

Cependant, pour les autres propriétés x (par exemple `maxPoolSize` ou`driverType`), la clé YAML normale `spring.jta.atomikos.datasource.XXXX.unique-resource-name` doit toujours être fournie.

La valeur secrète remplace les propriétés YAML.

BAC et JAC du Super Admin par défaut

Vous pouvez également configurer application-main.yml pour récupérer le nom d'utilisateur et le mot de passe de l'utilisateur super administrateur par défaut dans le secret d'AWS Secrets Manager en spécifiant l'ARN. L'exemple suivant montre comment déclarer ce secret dans un fichier YAML.

```
spring:
   aws:
     client:
       defaultSuperAdmin:
         secret: arn:aws:secretsmanager:XXXX
```

Clés secrètes de base de données super admin prises en charge par défaut :

Clé secrète	Description de la clé secrète
nom d'utilisateur	Le nom d'utilisateur.
mot de passe	Le mot de passe.

OAuth2

Vous pouvez également configurer `application-main.yml` pour récupérer le secret du OAuth2 client en spécifiant le fournisseur et l'ARN. AWS Secrets Manager La valeur par défaut de la propriété du fournisseur est Amazon Cognito. Voici un exemple de configuration pour le OAuth2 fournisseur Keycloak :

```
spring:
   aws:
     client:
       provider: keycloak
       keycloak:
         secret: arn:aws:secretsmanager:XXXX
```

Dans cet exemple, le secret client du OAuth2 fournisseur Keycloak est extrait de l'ARN spécifié dans AWS Secrets Manager. Cette configuration prend en charge plusieurs fournisseurs en résolvant dynamiquement le nom du fournisseur et l'ARN secret correspondant.

Clés OAuth2 secrètes prises en charge :

Clé secrète	Description de la clé secrète
client-secret	Le secret généré par le serveur d'autorisation lors du processus d'enregistrement de l'application.

Gestionnaire secret pour les caches Redis

Le `application-main.yml` fichier peut référencer l'ARN secret des caches Redis. Les modèles pris en charge sont les suivants :

- Informations d'identification Gapwalk Redis avec `spring.aws.client.gapwalk.redis.secret`
- Informations d'identification Bluesam Redis avec `spring.aws.client.bluesam.redis.secret`
- Bluesam verrouille les informations d'identification Redis avec `spring.aws.client.bluesam.locks.redis.secret`
- Informations d'identification Redis du catalogue de jeux de données avec `spring.aws.client.dataset.catalog.redis.secret`
- Informations d'identification JICS Redis avec `spring.aws.client.jics.redis.secret`
- Identifiants de session Redis avec `spring.aws.client.jics.redis.secret`
- Identifiants Redis de suivi de session avec `spring.aws.client.session.tracker.redis.secret`
- JICS TS met en file d'attente les informations d'identification Redis avec `spring.aws.client.jics.queues.ts.redis.secret`
- Informations d'identification JCL Checkpoint Redis avec `spring.aws.client.jcl.checkpoint.redis.secret`
- Les fichiers Gapwalk verrouillent les informations d'identification Redis avec `spring.aws.client.gapwalk.files.locks.redis.secret`

- Blu4iv verrouille les informations d'identification Redis avec `spring.aws.client.blu4iv.locks.redis.secret`

L'exemple suivant montre comment déclarer ces secrets dans un fichier YAML.

```
spring:
   aws:
     client:
       gapwalk:
         redis:
           secret: arn:aws:secretsmanager:XXXX
       bluesam:
         locks:
           redis:
             secret: arn:aws:secretsmanager:XXXX
         redis:
           secret: arn:aws:secretsmanager:XXXX
       dataset:
         catalog:
           redis:
             secret: arn:aws:secretsmanager:XXXX
       jics:
         redis:
           secret: arn:aws:secretsmanager:XXXX
       session:
         tracker:
           redis:
             secret: arn:aws:secretsmanager:XXXX
       jics:
         queues:
           ts:
             redis:
               secret: arn:aws:secretsmanager:XXXX
       jcl:
         checkpoint:
           redis:
             secret: arn:aws:secretsmanager:XXXX
       gapwalk:
         files:
           locks:
             redis:
               secret: arn:aws:secretsmanager:XXXX
       blu4iv:
```

```
        locks:
          redis:
            secret: arn:aws:secretsmanager:XXXX
```

Clés secrètes Redis prises en charge :

Clé secrète	Description de la clé secrète
hostname	Le nom d'hôte du serveur Redis.
port	Le port du serveur Redis.
nom d'utilisateur	Le nom d'utilisateur.
mot de passe	Le mot de passe.

Gestionnaire de secrets pour les paramètres de mot de passe SSL

Le `application-main.yml` fichier peut faire référence à l'ARN secret pour les paramètres de mot de passe SSL. Les éléments suivants sont pris en charge.

- Informations d'identification SSL Gapwalk avec `spring.aws.client.ssl.secret`

L'exemple suivant montre comment déclarer ces secrets dans un fichier YAML.

```
spring:
   aws:
     client:
       ssl:
         secret: arn:aws:secretsmanager:XXXX
```

Clé secrète	Description de la clé secrète
trustStorePassword	Le mot de passe du Truststore.
keyStorePassword	Le mot de passe du keystore.

Gestionnaire de secrets pour les paramètres de mot de passe IBM MQ

Le `application-main.yml` fichier peut faire référence à l'ARN secret pour les paramètres IBM MQ. Les éléments suivants sont pris en charge.

- Les connexions IBM MQ sont définies sous forme de liste, tout comme les informations d'identification :

  `mq.queues.jmsMQQueueManagers[N].secret:`

  N commence à 0 pour la première connexion.

L'exemple suivant montre comment déclarer ces secrets dans un fichier YAML.

```
mq.queues.jmsMQQueueManagers[0].secret: Secret-0-ARN
mq.queues.jmsMQQueueManagers[1].secret: Secret-1-ARN
```

Pour plus d'informations sur ARNs le secret, voir Que contient un secret de Secrets Manager ?

Les propriétés définies dans le secret remplaceront leurs valeurs correspondantes dans la configuration `jmsMQ` YAML.

S'il `queueManager` est défini dans le secret, il remplacera la `mq.queues.jmsMQQueueManagers[N].jmsMQQueueManager` valeur du fichier YAML.

Clé secrète	Description de la clé secrète
Gestionnaire de files d'attente	Nom du gestionnaire de files d'attente IBM MQ.
Nom de l'application	Nom de l'application IBM MQ.
channel	Nom du canal IBM MQ.
hôte	Le nom d'hôte IBM MQ.
port	Le port IBM MQ.
userId	Nom d'utilisateur IBM MQ.
mot de passe	Le mot de passe utilisateur IBM MQ.

Clé secrète	Description de la clé secrète
maxPoolSize	Taille maximale du pool IBM MQ.
sslCipherKey	La suite de chiffrement SSL IBM MQ.

Autres fichiers (groovy, sql, etc.)

Les autres fichiers utilisés par le projet client utilisent des règles de priorité similaires à celles de la configuration Spring. Exemples :

- Les scripts Groovy sont des `.groovy` fichiers contenus dans le `scripts` dossier ou les sous-dossiers.
- Les scripts SQL sont des `.sql` fichiers contenus dans le `sql` dossier ou les sous-dossiers.
- Les scripts Daemon sont des `.groovy` fichiers contenus dans le `daemons` dossier ou les sous-dossiers.
- Les fichiers de mappage de base de données de requêtes sont `queries-database.mapping` des fichiers nommés fichiers dans les sous-dossiers du `sql` dossier.
- Les modèles Jasper sont des `.jrxml` fichiers contenus dans le `templates` dossier ou les sous-dossiers.
- Les catalogues de jeux de données sont `.json` des fichiers contenus dans le `catalog` dossier.
- Les fichiers Lnk sont `.json` des fichiers du `lnk` dossier.

Tous ces emplacements peuvent être remplacés par le biais d'une propriété système ou d'une propriété YAML du client.

- Pour les scripts Groovy : `configuration.scripts`
- Pour les scripts SQL : `configuration.sql`
- Pour les scripts Daemon : `configuration.daemons`
- Pour le fichier de mappage de base de données de requêtes : `configuration.databaseMapping`
- Pour les modèles Jasper : `configuration.templates`
- Pour les catalogues de jeux de données : `configuration.catalog`
- Pour les fichiers Lnk : `configuration.lnk`

Si la propriété n'est pas trouvée, les fichiers seront extraits de l'emplacement par défaut mentionné ci-dessus. La recherche sera d'abord effectuée avec le répertoire de travail Tomcat en tant que racine, puis dans le fichier war de l'application.

Application Web supplémentaire

Le AWS Blu Age Runtime contient des applications Web supplémentaires dans son `webapps-extra` dossier. Ces applications ne sont pas desservies par défaut par le serveur Tomcat.

L'activation de ces applications Web dépend du projet de modernisation et s'effectue en déplaçant le fichier war souhaité du `webapps-extra` dossier vers le `webapps` dossier. Après cela, la guerre sera servie par le serveur Tomcat au prochain démarrage.

Certaines configurations supplémentaires spécifiques au projet peuvent également être ajoutées dans un fichier de configuration YAML pour chaque guerre supplémentaire, comme cela est fait dans le `application-main.yml` fichier et expliqué ci-dessus. Les guerres supplémentaires sont les suivantes :

- `gapwalk-utility-pgm.war`: contient le support pour les programmes utilitaires ZOS et l'utilise `application-utility-pgm.yaml` comme configuration.
- `gapwalk-cl-command.war`: contient le support pour les programmes utilitaires AS/400 et l'utilise `application-cl-command.yaml` comme configuration.
- `gapwalk-hierarchical-support.war`: contient le support des transactions IMS/MFS et l'utilise comme configuration `application-jhdb.yaml`

Activer les propriétés de AWS Blu Age Runtime

Dans les applications Spring Boot `application-main.yml` se trouve le fichier de configuration dans lequel nous définissons différents types de propriétés telles que le port d'écoute, la connectivité à la base de données, etc. Vous pouvez utiliser cette page pour en savoir plus sur les propriétés disponibles pour AWS Blu Age Runtime et pour savoir comment les activer.

Rubriques

- Notation YML
- Démarrage rapide/Cas d'utilisation
- Propriétés disponibles pour l'application principale
- Propriétés disponibles pour les applications Web facultatives

- Propriétés disponibles pour l'application cliente

Notation YML

Dans la documentation suivante, une propriété telle que celle-ci `parent.child1.child2=true` est écrite comme suit au format YAML.

```
parent:
 child1:
   child2: true
```

Démarrage rapide/Cas d'utilisation

Les cas d'utilisation suivants présentent des exemples de clés et de valeurs applicables.

- Application-main.yml par défaut

```
----
#### DEFAULT APPLICATION-MAIN.YML FILE      #####
#### SHOWING USEFUL CONFIGURATION ELEMENTS #####
#### SHOULD BE OVERRIDDEN AND EXTERNALIZED   #####

###################################
##### Logging configuration #####
###################################
logging:
  config: classpath:logback-main.xml
  level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN

###################################
##### Spring configuration #####
###################################
spring:
  quartz:
    auto-startup: false
    scheduler-name: Default
    properties:
      org.quartz.threadPool.threadCount: 1
  jta:
    enabled: false
    atomikos.properties.maxTimeout : 600000
    atomikos.properties.default-jta-timeout : 100000
```

```
  jpa:
# DISABLE OpenEntityManagerInViewInterceptor
    open-in-view: false
    # Fix Postgres JPA Error:
    # Method org.postgresql.jdbc.PgConnection.createClob() is not yet implemented.
    properties.hibernate.temp.use_jdbc_metadata_defaults : false
#####################################
##### Jics tables configuration #####
#####################################

    # The dialect should match the jics datasource choice
    database-platform : org.hibernate.dialect.PostgreSQLDialect #
 org.hibernate.dialect.PostgreSQLDialect, org.hibernate.dialect.SQLServerDialect

    # those properties can be used to create and initialize jics tables
 automatically.
#    properties:
#      hibernate:
#      globally_quoted_identifiers: true
#        hbm2ddl:
#           import_files_sql_extractor :
 org.hibernate.tool.hbm2ddl.MultipleLinesSqlCommandExtractor
#           import_files : file:./setup/initJics.sql
#           auto : create

#########################
###### Level 2 cache #####
#########################
#        cache:
#          use_second_level_cache: true
#          use_query_cache: true
#          region:
#            factory_class: org.hibernate.cache.ehcache.EhCacheRegionFactory
#      javax:
#        persistence:
#          sharedCache:
#            mode: ENABLE_SELECTIVE
#########################
###### Redis settings #####
#########################
  session:
    store-type: none #redis

# Secret manager configuration for global Redis cache
```

```
    aws:
      client:
        gapwalk:
          redis:
            secret: arn:aws:secretsmanager:XXXX

########################################
##### JICS datasource configuration #####
########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver # org.postgresql.Driver,
 com.microsoft.sqlserver.jdbc.SQLServerDriver
    url: jdbc:postgresql://localhost/jics # jdbc:postgresql://localhost:5433/jics,
 jdbc:sqlserver://localhost\SQLEXPRESS:1434;databasename=jics;
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource #
 org.postgresql.ds.PGSimpleDataSource,
 com.microsoft.sqlserver.jdbc.SQLServerDataSource

#######################################################
##### Embedded Bluesam datasource configuration #####
#######################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver
    url : jdbc:postgresql://localhost/bluesam
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource

########################################
##### Embedded Bluesam configuration #####
########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
    enabled: true
  pgsql :
```

```
    dataSource : bluesamDs

#########################
##### Jics settings #####
#########################
rabbitmq.host: localhost
jics:
  cache: false #redis
  resource-definitions.store-type: jpa # default value: jpa, other possible value:
 redis

jics.disableSyncpoint : false
#jics.initList:
#jics.parameters.datform: DDMMYY
#jics.parameters.applid: VELOCITY
#jics.parameters.sysid: CICS
#jics.parameters.eibtrmid: TERM
#jics.parameters.userid: MYUSERID
#jics.parameters.username: MYUSERNAME
#jics.parameters.opid: XXX
#jics.parameters.cwa.length: 0
#jics.parameters.netname: MYNETNAME
#jics.parameters.jobname: MJOBNAME
#jics.parameters.sysname: SYSNAME

###############################################
##### Jics RunUnitLauncher pool settings #####
###############################################
#jics.runUnitLauncherPool.enable: false
#jics.runUnitLauncherPool.size: 20
#jics.runUnitLauncherPool.validationInterval: 1000

#########################
##### Jhdb settings #####
#########################
#jhdb.lterm: LTERMVAL
#jhdb.identificationCardData: SomeIDData

###################################
##### DateHelper configuration ####
###################################
#forcedDate: "2013-08-26T12:59:58+01:57"

#############################
```

```
##### Sort configuration ####
#############################
#externalSort.threshold: 256MB

###################################
##### Server timeout (10 min)  ####
###################################
spring.mvc.async.request-timeout: 600000

###############################
##### DATABASE STATISTICS  ####
###############################
databaseStatistics : false

######################
##### CALLS GRAPH ####
######################
callGraph : false

###################################
#####     SSL configuration    #####
###################################
gapwalk.ssl.enabled : true
gapwalk.ssl.trustStore : "./config/clientkey.jks"
gapwalk.ssl.trustStorePassword : mysslcertifpassword

#################################
##### MQ settings #####
#################################
mq.queues: jmsmq
mq.queues.jmsMQQueueManagers[0].jmsMQQueueManager: QM1
mq.queues.jmsMQQueueManagers[0].jmsMQAppName: Gapwalk
mq.queues.jmsMQQueueManagers[0].jmsMQChannel: DEV.APP.SVRCONN
mq.queues.jmsMQQueueManagers[0].jmsMQHost: localhost
mq.queues.jmsMQQueueManagers[0].jmsMQPort: 1415
mq.queues.jmsMQQueueManagers[0].jmsMQUserid: app
mq.queues.jmsMQQueueManagers[0].jmsMQSSLCipher: "*TLS12ORHIGHER"
mq.queues.jmsMQQueueManagers[1].jmsMQQueueManager: QM2
mq.queues.jmsMQQueueManagers[1].jmsMQAppName: Gapwalk
mq.queues.jmsMQQueueManagers[1].jmsMQChannel: DEV.APP.SVRCONN
mq.queues.jmsMQQueueManagers[1].jmsMQHost: localhost
mq.queues.jmsMQQueueManagers[1].jmsMQPort: 1415
mq.queues.jmsMQQueueManagers[1].jmsMQUserid: app
```

```
################################
##### SQL SHIFT CODE POINT ####
################################
# Code point 384 match unicode character \u0180
sqlCodePointShift : 384

################################
##### LOCK TIMEOUT RECORD  ####
################################
# Blu4IV record lock timeout
lockTimeout : 100

################################
##### REPORTS OUTPUT PATH ####
################################
reportOutputPath: reports

################################
##### TASK EXECUTOR       ####
################################
taskExecutor:
  corePoolSize: 5
  maxPoolSize: 10
  queueCapacity: 50
  allowCoreThreadTimeOut: false

################################
##### PROGRAM NOT FOUND   ####
################################
stopExecutionWhenProgNotFound: false

#########################################################
#####  DISP DEFAULT VALUE (to be removed one day) ####
#########################################################
defaultKeepExistingFiles: true

#########################################################
#####  BLOCKSIZE DEFAULT VALUE  ####
#########################################################
#blockSizeDefault: 32760

#####################################
#####  JOBQUEUE CONFIGURATION    ####
#####################################
```

```
jobqueue:
  api.enabled: false
  impl: none # possible values: quartz, none
  schedulers:  # list of schedulers
    -
      name: queue1
      threadCount: 5
    -
      name: queue2
      threadCount: 5

################################################################################
##### QUERY BUILDING                                                        ##
# useConcatCondition : false by default
# if true, in the query, the where condition is build with key concatenation  ##
################################################################################
# query.useConcatCondition: true

########################################
#####  JCL Batch Restart Mechanism  ####
########################################
jcl:
checkpoint:
enabled: false
#expireTimeout: -1
#expireTimeoutUnit: SECONDS # Supported values: java.util.concurrent.TimeUnit
#provider: redis

----
```

- Utiliser des fichiers de longueur variable avec les commandes LISTCAT

```
[**/*.*]
encoding=IBM930
reencoding=false

[global]
listcat.variablelengthpreprocessor.enabled=true
listcat.variablelengthpreprocessor.type=rdw
# use "rdw" if your .listcat file contains a set of records (RDW)
# use "bdw" if your .listcat file contains a set of blocks (bdw)
```

- Fournir une valeur d'indicateur d'octet nul dans l'utilitaire LOAD/UNLOAD

```
# Unload properties
# For date/time: if use database configuration is enabled, formats are ignored
# For nbi; use hexadecimal syntax to specify the byte value
# - When the value is null in database : the value dumped to the file is filled by
 low value characters and the NBI is
# equal to the byte 6F (the ? character)
# - When the value is not null in database and the column is nullable: the NBI is
 equal to the byte 00 (low value) and NOT
# equal to the byte 40 (space)
unload:
  sqlCodePointShift: 0
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
```

Propriétés disponibles pour l'application principale

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs.

Clé	Type	Valeur par défaut	Description	Version de sortie
`logging.config`	Chemin	chemin de classe : logback-main.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalisation standard sont également disponibles.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`spring.jta.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa , la configuration automatique des transactions Spring JTA doit être désactivée.	
`datasource.jicsDs` + `-driver-class-name` + `-url` + `-username` + `-password` + `-type`	Source de données Spring standard avec sous-clés		Contient les informations de connexion à la base de données Jics. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dansthe section called "Base de données JICS".	

Clé	Type	Valeur par défaut	Description	Version de sortie
`datasource.bluesamDs` + `-driver-class-name` + `-url` + `-username` + `-password` + `-type`	Source de données Spring standard avec sous-clés		Contient les informations de connexion à la base de données Blusam. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dansthe section called "Base de données Blusam".	
`bluesam.disabled`	boolean	false	S'il faut désactiver complètement Blusam.	
`bluesam.cache`	chaîne		S'il n'est pas défini, le cache Blusam ne sera pas utilisé. Les valeurs possibles (implémentations du cache) sont cache et redis ()the section called "Propriétés du cache Redis".	

Clé	Type	Valeur par défaut	Description	Version de sortie
`bluesam.maxBluesamDisablingThreadpoolSize`	nombre	10	Spécifie la taille maximale du pool de threads utilisé pour désactiver les ensembles de données Bluesam pour le traitement par lots.	4.5.0
`bluesam.bluesamStatusPollingInterval`	nombre	1 000	Spécifie le temps (en millisecondes) à attendre entre chaque itération lorsque vous interrogez l'état de Bluesam pour vérifier les activités en ligne.	4.5.0
`bluesam.maxBluesamStatusPollingRetry`	nombre	3	Spécifie le nombre maximum de tentatives en cas d'échec du sondage sur le statut de Bluesam.	4.5.0
`bluesam.checkBluesamStatus`	boolean	false	Spécifie s'il faut vérifier l'état du jeu de données Bluesam avant d'y accéder.	4.5.0

Clé	Type	Valeur par défaut	Description	Version de sortie
`spring.aws.client.bluesam.redis.secret`	chaîne	null	Spécifie l'ARN secret d'identification pour le cache Bluesam Redis, voir. the section called "AWS Les secrets de Blu Age Runtime"	
`spring.aws.client.bluesam.locks.redis.secret`	chaîne	null	Spécifie l'ARN secret d'identification pour Bluesam verrouille le cache Redis, voir. the section called "AWS Les secrets de Blu Age Runtime"	
`forcedDate`	chaîne		Force la date à la date indiquée s'il y en a une.	
`frozenDate`	booléen	true	Spécifie s'il faut geler la date. S'appliqu e uniquement s'`forcedDat e` il est également défini.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`externalSort.threshold`	taille des données (exemple : 12 Mo)		Le seuil de tri : quand passer au tri externe (fusion).	
`blockSizeDefault`	nombre	32760	Taille de bloc par défaut à utiliser pour les octets BDW.	
`jics.parameters.datform`	chaîne	MMDDYY	Le formulaire de date.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.initList`	chaîne		La liste JICS d'initialisation, séparée par des virgules. Le cas échéant, il définit les noms de listes séparés par des virgules à activer au démarrage d'Apache Tomcat parmi les listes CICS. Exemple de valeur : `$UUU,DFH$IVPL,PEZ1` . Cela se répercutera sur les groupes contenus dans ces listes et leurs définitions de ressources sous-jacentes, qui seront ensuite visibles par le moteur d'exécution. Vide par défaut.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.parameters.applid`	chaîne	VÉLOCITÉ	Appliqué pour identifier l'application dans JICS (au moins 4 caractères, pas de longueur maximale).	
`jics.parameters.sysid`	chaîne	CICS	L'identification du système (SYSID).	
`jics.parameters.eibtrmid`	chaîne	TERME	L'identifiant du terminal (4 caractères maximum, 1 minimum).	
`jics.parameters.userid`	chaîne		Le nom d'utilisateur (8 caractères maximum, pas de minimum). Lorsqu'aucune valeur n'est fournie (vide par défaut), l'identifiant de session HTTP est utilisé comme identifiant utilisateur.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.parameters.username`	chaîne	MON NOM D'UTILISATEUR	Le nom d'utilisateur (10 caractères maximum, 1 minimum).	
`jics.parameters.netname`	chaîne	MYNETNAME	Le nom du réseau (8 caractères maximum, 1 au minimum).	
`jics.parameters.opid`	chaîne	XXX	L'identification de l'opérateur à 3 caractères.	
`jics.parameters.jobname`	chaîne	NOM DU POSTE	Le nom du poste.	
`jics.parameters.sysname`	chaîne	SYSNAME	Le nom du système AS4 00 (nom système).	
`jics.parameters.cwa.length`	nombre	0	La longueur de la zone de travail commune (CWA).	
`jics.parameters.charset`	chaîne	CP037	Jeu de caractères utilisé dans le monde entier par JICS.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.parameters.tsqimpl`	chaîne	bluesam	Implémentation de la file d'attente de stockage temporaire (TSQ) JICS (les valeurs autorisées sont`bluesam`/`mem`/`redis`	
`jics.queues.ts.redis.*`	Propriétés Redis prises en charge		Spécifie les propriétés de configuration pour le serveur Redis JICS TS Queues, voir. the section called "Propriétés Redis prises en charge"	
`spring.aws.client.jics.queues.ts.redis.secret`	chaîne	null	Spécifie l'ARN secret d'identification pour le serveur Redis JICS TS Queues, voir. the section called "AWS Les secrets de Blu Age Runtime"	

Clé	Type	Valeur par défaut	Description	Version de sortie
`lockTimeout`	nombre	500	Le délai d'expiration du verrouillage, en millisecondes.	
`sqlCodePointShift`	nombre		Facultatif. Le changement de point de code SQL. Déplace le point de code pour les caractères de contrôle que nous pouvons rencontrer lors de la migration de données SGBDR existantes vers un SGBDR moderne. Par exemple, vous pouvez spécifier de `384` faire correspondre le caractère Unicode`\u0180`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`sqlIntegerOverflowAllowed`	boolean	false	Spécifie s'il faut autoriser le dépassement des nombres entiers SQL, c'est-à-dire s'il est permis de placer des valeurs plus importantes dans la variable hôte.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`database.cursor.overflow.allowed`	booléen	true	Spécifie s'il faut autoriser le dépassement du curseur. Réglez sur `true` pour effectuer un appel suivant sur le curseur, quelle que soit sa position. Réglez `false` sur pour vérifier si le curseur est à la dernière position avant d'effectuer un prochain appel sur le curseur. Activez uniquement si le curseur est SCROLLABLE (SENSITIVE ou INSENSITIVE).	
`reportOutputPath`	chaîne	`/reports`	Le chemin de sortie du rapport.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`spring.session.store-type`	chaîne	none	Le cache de session pour les environnements à haute disponibilité. Les valeurs possibles sont `none` ou `redis`. La valeur par défaut est `none`.	
`stopExecutionWhenProgNotFound`	booléen	true	Spécifie s'il faut arrêter l'exécution si aucun programme n'est trouvé. S'il est défini sur `true`, interrompt l'exécution si aucun programme n'est trouvé.	
`forceHR`	boolean	false	Spécifie s'il faut utiliser SYSPRINT lisible par l'homme, soit en sortie de console, soit en sortie de fichier.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`rollbackOnRTE`	boolean	false	Spécifie s'il faut annuler la transaction d'unité d'exécution implicite sur les exceptions d'exécution.	
`sctThreadLimit`	long	5	La limite de threads pour le déclenchement de scripts.	
`dataSimplifier.onInvalidNumericData`	chaîne	rejeter	Comment réagir lors du décodage de données numériques non valides Les valeurs autorisées sont`reject/tolerate spaces /tolerate spaceslowvalues /tolerate ost` . La valeur par défaut est`reject`.	
`filesDirectory`	chaîne		Le répertoire des fichiers d'entrée/sortie par lots.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`ims.messages.extendedSize`	boolean	false	Spécifie s'il faut définir la taille étendue des messages IMS.	
`defaultKeepExistingFiles`	boolean	false	Spécifie s'il faut définir la valeur précédente par défaut de l'ensemble de données.	
`jics.db.ddlScriptLocation`	chaîne		L'emplacement du script DDL Jics. Permet de lancer le schéma de base de données Jics à l'aide d'un script .sql. Vide par défaut. Par exemple, `./jics/sql/jics.sql` .	
`jics.db.schemaTestQueryLocation`	chaîne		Emplacement du fichier SQL qui doit contenir une requête unique renvoyant le nombre d'objets du schéma jics (le cas échéant).	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.db.dataScriptLocation`	chaîne		Définit le chemin d'accès aux scripts SQL utilisés pour initialiser la base de données JICS. Accepte une liste de fichiers et de répertoires séparés par des virgules, ce qui permet de spécifier plusieurs scripts et dossiers.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.db.dataTestQueryLocation`	chaîne		Emplacement d'un script SQL contenant une seule requête SQL censée renvoyer un nombre d'objets (par exemple : compter le nombre d'enregistrements dans la table du programme jics). Si le nombre est égal à 0, la base de données sera chargée à l'aide du `jics.db.dataScriptLocation` script, sinon le chargement de la base de données sera ignoré.	
`jics.data.dataJsonInitLocation`	chaîne			
`jics.xa.agent.timeout`	nombre			

Clé	Type	Valeur par défaut	Description	Version de sortie
`query.useConcatCondition`	boolean	false	Spécifie si la condition clé est créée par concaténation de clés ou non.	
`system.qdecfmt`	chaîne			
`disposition.checkexistence`	boolean	false	Spécifie s'il convient de vérifier l'existence du fichier pour Dataset avec DISP SHR ou OLD.	
`useControlMVariable`	boolean	false	Spécifie s'il faut utiliser la spécification Control-M pour le remplacement des variables.	
`card.encoding`	chaîne	CP1145	Encodage de la carte : à utiliser avec`useControlMVariable` .	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mapTransfo.prefixes`	chaîne	&,@,%%	Liste des préfixes à utiliser lors de la transformation des variables ControlM. Chacun d'eux est séparé par une virgule.	
`checkinputfilesize`	boolean	false	Spécifie s'il convient de vérifier si la taille du fichier est un multiple de la taille de l'enregistrement.	
`stepFailWhenAbend`	booléen	true	Spécifie s'il faut déclencher un abend en cas d'échec ou de fin d'exécution d'une étape.	
`bluesam.fileLoading.commitInterval`	nombre	100 000	L'intervalle de validation du bluesam.	
`uppercaseUserInput`	booléen	true	Spécifie si les données saisies par l'utilisateur doivent être en majuscules.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jhdb.lterm`	chaîne		Permet de forcer un identifiant de terminal logique commun dans le cas d'une émulation IMS. S'il n'est pas défini, SessionId est utilisé.	
`jhdb.identificationCardData`	chaîne		Utilisé pour coder en dur certaines « données de la carte d'identification de l'opérateur » dans le champ MID désigné par le paramètre CARD. Vide par défaut, aucune restriction de saisie.	
`encoding`	chaîne	ASCII	L'encodage utilisé dans les projets (pas dans les fichiers groovy). Exige un encodage valide `CP1047IBM930,,A`	

Clé	Type	Valeur par défaut	Description	Version de sortie
`cl.configuration.context.encoding`	chaîne	CP297	L'encodage des fichiers CL. Exige un encodage valide `CP1047IBM930,,A` La valeur par défaut est `CP297`	
`cl.zonedMode`	chaîne	EBCDIC_STRICT	Mode d'encodage ou de décodage des commandes du langage de contrôle (CL). Les valeurs autorisées sont`EBCDIC_STRICT /EBCDIC_M( DIFIED /AS400`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`ims.programs`	chaîne		Liste des programmes IMS à utiliser. Séparez chaque paramètre par un point-virgule (`;`) et chaque transaction par une virgule (). `,` Exemples :`PCP008T008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`	
`jhdb.configuration.context.encoding`	chaîne	CP297	Le codage JHDB (base de données hiérarchique Java). Exige une chaîne de codage valide`CP1047,IBM`	
`jhdb.metadata.extrapath`	chaîne	fichier :. /configuration/	Paramètre de configuration qui spécifie un dossier racine supplémentaire spécifique à l'exécution pour les dossiers psbs et dbds.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jhdb.checkpointPersistence`	chaîne	none	Le mode de persistance du point de contrôle. Les valeurs autorisées sontnone/add/end addÀ utiliser pour conserver les points de contrôle lorsqu'un nouveau point est créé et ajouté au registre. Utilisez un point de contrôle end trop persistant lors de l'arrêt du serveur. Toute autre valeur désactive la persistance. Notez que chaque fois qu'un nouveau point de contrôle est ajouté au registre, tous les points de contrôle existants sont sérialisés et le fichier est effacé. Il ne s'agit	

Clé	Type	Valeur par défaut	Description	Version de sortie
			pas d'un ajout aux données existantes du fichier. Ainsi, en fonction du nombre de points de contrôle, cela peut avoir des effets sur les performances.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jhdb.checkpointPath`	chaîne	fichier :. /configuration/	Si ce n'`jhdb.checkpointPersistence` est pas le `casnone`, ce paramètre vous permet de configurer le chemin de persistance du point de contrôle (emplacement de stockage du fichier checkpoint.dat). Toutes les données des points de contrôle contenues dans le registre sont sérialisées et sauvegardées dans un fichier (checkpoint.dat) situé dans le dossier fourni. Notez que seules les données du point de contrôle (ScriptID, StepID, position de la base de données et	

Clé	Type	Valeur par défaut	Description	Version de sortie
			zone du point de contrôle) sont concernées par cette sauvegarde.	
`jhdb.navigation.cachenexts`	nombre	5000	Durée du cache (en millisecondes) utilisée dans la navigation hiérarchique pour un SGBDR.	
`jhdb.use-db-prefix`	booléen	true	Spécifie s'il faut activer un préfixe de base de données dans la navigation hiérarchique pour un SGBDR.	
`jhdb.query.limitJoinUsage`	booléen	true	Spécifie s'il faut utiliser le paramètre d'utilisation limite des jointures sur les graphes RDBMS.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`taskExecutor.corePoolSize`	nombre	5	Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille du pool principal.	
`taskExecutor.maxPoolSize`	nombre	10	Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille maximale du pool (nombre maximal de threads parallèles).	

Clé	Type	Valeur par défaut	Description	Version de sortie
`taskExecutor.queueCapacity`	nombre	50	Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille de la file d'attente. (= nombre maximum de transactions en attente lorsqu'il `taskExecutor.maxPoolSize` est atteint)	

Clé	Type	Valeur par défaut	Description	Version de sortie
`taskExecutor.allowCoreThreadTimeOut`	boolean	false	Spécifie s'il faut autoriser les threads principaux à expirer dans JCIS. Cela permet une croissance et une réduction dynamiques, même en combinaison avec une file d'attente différente de zéro (étant donné que la taille maximale du pool n'augmentera que lorsque la file d'attente sera pleine).	
`jics.runUnitLauncherPool.enable`	boolean	false	Spécifie s'il faut activer le pool de lanceurs d'unités exécutées dans JICS.	
`jics.runUnitLauncherPool.size`	nombre	20	Taille du pool de lanceurs d'unités exécutées dans JICS.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.runUnitLauncherPool.validationInterval`	nombre	1 000	Intervalle entre chaque exécution de la tâche qui ajuste la taille du pool.	
`jics.runUnitLauncherPool.parallelism`	nombre	2	Le nombre de threads utilisés pour produire les instances manquantes dans la file d'attente lors de l'exécution de la tâche d'ajustement.	
`context.preconstruct.enable`	boolean	false	Spécifie s'il faut activer la préconstruction du contexte du programme.	
`context.preconstruct.frequencyInMillis`	nombre	100	Intervalle entre chaque exécution de la tâche qui ajuste la taille du pool.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`context.preconstruct.parallelism`	nombre	5	Le nombre de threads utilisés pour produire les instances manquantes dans la file d'attente lors de l'exécution de la tâche d'ajustement.	
`context.preconstruct.minInstances`	nombre	2	Le nombre d'instances qui seront créées la première fois qu'un contexte est nécessaire.	
`spring.aws.application.credentials`	chaîne	null	Chargez les AWS informations d'identification à partir du fichier de profils d'identification dans JICS.	
`jics.queues.sqs.region`	chaîne	eu-west-1	AWS Région pour Amazon Simple Queue Service, utilisée dans JICS.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.jcl.rt.encoding`	chaîne	CP037	Le codage des scripts JCL écrits dans la file d'attente JICS dédiée.	
`jics.jcl.rt.queue`	chaîne	JICS	Nom de la file d'attente dans laquelle les scripts JCL peuvent être écrits ligne par ligne lors de l'exécution.	
`mq.queues.sqs.region`	chaîne	eu-west-3	AWS Région du service AWS SQS MQ.	
`quartz.scheduler.stand-by-if-error`	boolean	false	Spécifie s'il faut déclencher l'exécution des tâches si le planificateur de tâches est en mode veille. Si vrai, lorsque cette option est activée, l'exécution de la tâche n'est pas déclenchée.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`databaseStatistics`	boolean	false	Spécifie s'il faut autoriser les générateurs SQL à collecter et à afficher des informations statistiques.	
`dbDateFormat`	chaîne	yyyy-MM-dd	Le format de date cible de la base de données.	
`dbTimeFormat`	chaîne	HH : mm : SS	Le format d'heure cible de la base de données.	
`dbTimestampFormat`	chaîne	yyyy-MM-dd HH : MM : SS.SSSSSS	Le format d'horodatage cible de la base de données.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`dateTimeFormat`	chaîne	ISO	dateTimeFormat décrit comment répartir la date, l'heure et le type d'horodatage de la base de données dans des entités simplificatrices de données. Les valeurs autorisées sont`ISO/EUR//EUR/`	
`localDateFormat`	chaîne		Liste des formats de date locaux. Séparez chaque format par. \	
`localTimeFormat`	chaîne		Liste des formats d'heure locale. Séparez chaque format par \	
`localTimestampFormat`	chaîne		Liste des formats d'horodatage locaux. Séparez chaque format par\.	
`pgmDateFormat`	chaîne	yyyy-MM-dd	Le format de la date et de l'heure.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`pgmTimeFormat`	chaîne	HH.mm.ss	Le format d'heure utilisé pour l'exécution de pgm (programmes).	
`pgmTimestampFormat`	chaîne	YYYY-MM-DD-HH.MM.SS.SSSSSSSS	Le format d'horodatage.	
`cacheMetadata`	booléen	true	Spécifie s'il faut mettre en cache les métadonnées de base de données.	
`forceDisableSQLTrimStringType`	boolean	false	Spécifie s'il faut désactiver le découpage de tous les paramètres de chaîne SQL.	
`fetchSize`	nombre		La valeur FetchSize pour les curseurs. À utiliser lors de la récupération de données à l'aide de fragments par des utilitaires de chargement/déchargement.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`check-groovy-file`	booléen	true	Spécifie s'il faut vérifier le contenu des fichiers groovy avant de les enregistrer.	
`qtemp.uuid.length`	nombre	9	La longueur de l'identifiant unique QTEMP.	
`qtemp.dblog`	boolean	false	S'il faut activer la journalisation de la base de données QTEMP.	
`qtemp.cleanup.threshold.hours`	nombre	0	Pour spécifier quand `qtemp.dblog` est activé. Durée de vie de la partition de base de données (en heures).	
`sort.function`	chaîne		Nom de la fonction de tri pour la base de données blu4iv.	
`invalidDataTolerence`	booléen	true	Spécifie si les données non valides sont tolérées pour le type compressé.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`program.timeout`	nombre	-1	Spécifie un délai d'expiration pour toute exécution de programme /transaction en secondes. Passé ce délai, le système essaiera d'interrompre le programme.	
`gapwalk.line.separator`	chaîne	null	Spécifie le type de séparateur de lignes dans Gapwalk. Les valeurs autorisées sont WIN (CRLF)/UNIX (LF)/LINUX (LF). Les autres valeurs sont ignorées et la propriété System line.sepatator est utilisée.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`enableActivePgmIdCache`	boolean	false	Spécifie s'il faut activer le cache local de l'ID de programme actif. Utilisez cette fonctionnalité avec prudence car les ressources JICS peuvent être partagées entre les programmes et les utilisateurs. Ces ressources peuvent être modifiées en externe par n'importe quel administrateur et le cache local mis en place peut être invalidé.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mq.queues.default.syncpoint`	boolean	false	Spécifie le comportement par défaut des commandes MQ PUT lorsque ni MQPMO_SYNCPOINT ni MQPMO_NO_SYNCPOINT ne sont définis. Lorsqu'il est défini sur true, il agit comme `MQPMO_SYNCPOINT` et les messages ne sont PAS directement validés pendant la commande PUT. Lorsqu'il est défini sur false, il agit comme si `MQPMO_NO_SYNCPOINT` les messages étaient directement validés lors de la commande PUT.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`dataSimplifier.byteRangeBoundsCheck`	boolean	false	Lorsqu'il est défini sur true, il garantit que non ByteRange est créé avec des valeurs incorrectes. La valeur par défaut est false.	
`file.stdoutIntoLogger`	boolean	false	Spécifie s'il faut activer l'écriture dans l'enregistreur au lieu du flux de sortie système par défaut dans les `SYSPUNCH` fichiers `SYSPRINT` et fichiers par défaut.	
`tempFilesDirectory`	chaîne	null	Spécifie le nom de l'emplacement du dossier dans lequel se trouvent les fichiers temporaires générés.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`cleanTempFilesDirectoryAtStartup`	booléen	true	Spécifie s'il faut purger le contenu du dossier des fichiers temporaires au démarrage de l'application.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`tempFolderPattern`	chaîne	null	Spécifie un modèle qui sera utilisé pour créer dynamiquement le nom du dossier temporaire en fonction des informations prédéfinies et personnalisables suivantes. HOST : le nom de l'hôte. JOBID : l'identifiant de la tâche. HASHCODE : le code de hachage du contexte du job. TIMESTAMP : le modèle à utiliser pour obtenir l'horodatage. Le nom cible du dossier temporaire est TMP_DIR_{}. tempFolderPattern Par exemple, dans le cas du modèle	

Clé	Type	Valeur par défaut	Description	Version de sortie
			suivant, le nom commencera par l'ID de la tâche et se terminera par le « timestamp » tempFolde rPattern : JOBID, HOST=XXXX X, HASHCODE, TIMESTAMP =YYYYMMDD HHMMSS. Si la propriété n'`tempFolde rPattern` est pas ajoutée au fichier YAML ou si elle est vide, le nom du dossier temporaire sera « TMP_DIR_ » + this.hashCode () (). DefaultJo bContext	
`database. cursor.ra ise.alrea dy.opened .error`	boolean	false	Spécifie s'il faut activer le déclenche ment de l'erreur SQLCODE 502 lorsqu'un curseur déjà ouvert s'ouvre.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.spool.smtp.hostname`	chaîne	null	Spécifie l'hôte du serveur SMTP. Exemple : `smtp.xxx.com`	
`jics.spool.smtp.port`	chaîne	null	Spécifie le port du serveur SMTP. Exemple : 25	
`jics.spool.smtp.password`	chaîne	null	Spécifie le mot de passe de connexion du serveur SMTP.	
`jics.spool.smtp.username`	chaîne	null	Spécifie le nom d'utilisateur du serveur SMTP.	
`jics.spool.smtp.debug`	boolean	false	Spécifie le mode de débogage pour le serveur SMTP.	
`gapwalk-application.security`	chaîne	`disabled`	Activez la configuration de sécurité globale (authentification XSS, CORS, CSRF, OAUTH...). Les valeurs autorisées sont `disabled` et `enabled`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk-application.identity`	chaîne	null	Méthode d'authentification globale. La valeur recommandée est `oauth`. Les valeurs autorisées sont `json` et `oauth`. Cette option est requise lorsque c'`gapwalk-application.security` est le cas `enabled`.	
`gapwalk-application.security.issuerUri`	chaîne	null	L'URI de l'émetteur du fournisseur d'identité (IdP). Cette option est requise lorsque c'`gapwalk-application.identity` est le cas `oauth`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk-application.security.allowedOrigins`	chaîne []	`null`	La liste des origines à autoriser. Cette option doit `gapwalk-application.identity` être définie sur `oauth`.	
`gdgDirectoryPath`	chaîne	`output/gdg`	Le chemin du répertoire GDG est le répertoire dans lequel les fichiers GDG sont stockés.	4.6.0
`gapwalk-application.security.claimGroupName`	chaîne	`cognito:groups`	L'attribut de réclamation qui contient la liste de tous les groupes auxquels appartient un utilisateur. `cognito:groups` À utiliser pour Amazon Cognito ou toute autre chaîne pour un IdP étranger.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk-application.security.userAttributeName`	chaîne	`username`	Le nom de l'attribut de réclamation utilisé pour identifier une demande utilisateur. `username`À utiliser pour Amazon Cognito, `preferred_username` pour Keycloak ou toute autre chaîne pour un IdP étranger.	
`gapwalk-application.security.localhostWhitelistingEnabled`	booléen	true	Spécifie s'il faut activer l'authentification à partir de n'importe quelle `localhost` demande.	
`gapwalk-application.defaultSuperAdminUserName`	chaîne	`sadmin`	Lorsque cette option `gapwalk-application.security` est désactivée, indique le nom du super utilisateur local par défaut.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk-application.defaultSuperAdminUserPwd`	chaîne	`sadmin`	Lorsque cette option `gapwalk-application.security` est désactivée, spécifie le mot de passe du super utilisateur local par défaut.	
`gapwalk-application.security.filterURIs`	chaîne	`disabled`	Activer/désactiver la URIs configuration du filtrage. Les valeurs autorisées sont `disabled` et `enabled`.	
`gapwalk-application.security.blockedURIs`	chaîne []	null	La liste des personnes URIs à bloquer. Cette option est requise lorsque c'`gapwalk-application.security.filterURIs` est le cas`enabled`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jics.redis.*`	Propriétés Redis prises en charge		Spécifie les propriétés de configuration pour la fabrique de connexions au serveur JICS Redis, voir. the section called "Propriétés Redis prises en charge"	
`spring.aws.client.jics.redis.secret`	chaîne	null	Spécifie l'ARN secret d'identification pour la fabrique de connexions au serveur JICS Redis, voir. the section called "AWS Les secrets de Blu Age Runtime"	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jcl.checkpoint.enabled`	boolean	false	Spécifie si le mécanisme de point de contrôle JCL est activé pour permettre le redémarrage du travail. Les points de contrôle JCL sont créés et enregistrés dans le registre en mémoire au début de chaque étape ou appel du programme principal. Tous les points de contrôle au niveau de l'étape sont conservés à la fin de la tâche, si le fournisseur de persistance est défini.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jcl.checkpoint.expireTimeout`	nombre	-1	Spécifie la durée pendant laquelle les points de contrôle JCL doivent être conservés dans le fournisseur de persistance ou dans le registre en mémoire.	
`jcl.checkpoint.expireTimeoutUnit`	chaîne	SECONDES	Spécifie l'unité de durée de la `jcl.checkpoint.expireTimeout` propriété . Valeurs constantes enum prises en charge : java.util.concurrent.TimeUnit.	
`jcl.checkpoint.provider`	chaîne	null	Spécifie le fournisseur de persistance du mécanisme de point de contrôle JCL. Les valeurs autorisées sont`redis`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`jcl.checkpoint.redis.*`	Propriétés Redis prises en charge		Spécifie les propriétés de configuration pour le fournisseur de persistance REDIS du mécanisme de point de contrôle JCL, voir. the section called "Propriétés Redis prises en charge"	
`spring.aws.client.jcl.checkpoint.redis.secret`	chaîne	null	Spécifie l'ARN secret d'identification pour le fournisseur de persistance Redis du mécanisme de point de contrôle JCL, voir. the section called "AWS Les secrets de Blu Age Runtime"	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk.ssl.enabled`	boolean	false	Indiqué pour définir les `gapwalk.ssl.*` propriétés suivantes sur les propriétés actuelles du système JVM si elles ne sont pas déjà définies au démarrage de l'application.	
`gapwalk.ssl.trustStore`	chaîne	null	Définissez la valeur sur la propriété système `javax.net.ssl.trustStore` si ce n'est pas déjà fait au démarrage de l'application.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk.ssl.trustStorePassword`	chaîne	null	Définissez la valeur sur la propriété système `javax.net.ssl.trustStorePassword` si ce n'est pas déjà fait au démarrage de l'application. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dansthe section called "Gestionnaire de secrets pour les paramètres de mot de passe SSL".	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk.ssl.trustStoreType`	chaîne	null	Définissez la valeur sur la propriété système `javax.net.ssl.trustStoreType` si ce n'est pas déjà fait au démarrage de l'application.	
`gapwalk.ssl.keyStore`	chaîne	null	Définissez la valeur sur la propriété système `javax.net.ssl.keyStore` si ce n'est pas déjà fait au démarrage de l'application.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk.ssl.keyStorePassword`	chaîne	null	Définissez la valeur sur la propriété système `javax.net.ssl.keyStorePassword` si ce n'est pas déjà fait au démarrage de l'application. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dans the section called "Gestionnaire de secrets pour les paramètres de mot de passe SSL".	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mq.queues`	chaîne	`sqs`	Spécifie le courtier de files d'attente compatible à `sqs` utiliser entre Amazon SQS`rabbitmq`, Rabbit MQ sur site ou IBMMQ `jms` sur site.	
`mq.queues.jmsMQQueueManagers[N]`			Lorsque `mq.queues` c'est `jms` le cas, permet de spécifier une liste de connexions IBM MQ. `mq.queues.jmsMQQueueManagers[0]` pour la première connexion, `mq.queues.jmsMQQueueManagers[1]` pour la seconde et ainsi de suite.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mq.queues.jmsMQQueueManagers[N].jmsMQQueueManager`	chaîne	null	Nom du gestionnaire de files d'attente IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQAppName`	chaîne	null	Nom de l'application IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQChannel`	chaîne	null	Le nom du canal IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQHost`	chaîne	null	Le nom d'hôte IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQPort`	nombre	null	Le port IBMMQ.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mq.queues.jmsMQQueueManagers[N].jmsMQUserid`	chaîne	null	Le nom d'utilisateur IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQPassword`	chaîne	null	Le mot de passe de l'utilisateur IBMMQ. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dansthe section called "Gestionnaire de secrets pour les paramètres de mot de passe IBM MQ".	
`mq.queues.jmsMQQueueManagers[N].jmsMQMaxPoolSize`	nombre	0	Taille maximale du pool IBMMQ. Avec 0, un nombre infini de connexions physiques sont activées.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mq.queues.jmsMQQueueManagers[N].jmsMQSSLCipher`	chaîne	null	La suite de chiffrement SSL IBMMQ. Un exemple pourrait être`"*TLS12OR HIGHER"` . Reportez-vous à la documentation officielle TLS CipherSpecs et CipherSuites aux classes IBM MQ pour JMS pour plus de détails.	
`mq.queues.non.jms.client`	boolean	false	Indiquez si le client cible auquel envoyer des messages n'est pas JMS. Le format MQ natif sera utilisé pour les clients non-JMS tandis que le RFH2 format sera utilisé pour JMS.	4.5.0
			Quand `mq.queues` c'est le `casrabbitmq`, le nom d'hôte IBMMQ.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`mq.queues.rabbitMQHost`			Le nom d'hôte du Rabbit MQ.	
`mq.queues.rabbitMQVirtualHost`			Le nom d'hôte virtuel du Rabbit MQ.	
`mq.queues.rabbitMQPort`			Le port Rabbit MQ.	
`mq.queues.rabbitMQUsername`			L'utilisateur de Rabbit MQ.	
`mq.queues.rabbitMQPassword`			Le mot de passe Rabbit MQ.	
`mf.runtime.switch.N`	booléen	true	Permet l'insertion nulle pour les fichiers séquentiels de lignes MF Nature.	4.4.0
`mf.runtime.switch.T`	boolean	false	Permet d'insérer des caractères de tabulation dans les fichiers séquentiels de lignes MF Nature.	4.4.0

Clé	Type	Valeur par défaut	Description	Version de sortie
`gapwalk.database.support.useSavePointToRestoreFail`	boolean	false	Permet de récupérer les transactions en cas d'échec en utilisant des points de sauvegarde lors des requêtes d'insertion. L'activation de cette propriété peut affecter les performances de la base de données. Vous pouvez remplacer ce paramètre pour des requêtes spécifiques à l'aide de la configuration de query-to-database mappage.	4.6.0

Propriétés disponibles pour les applications Web facultatives

En fonction de votre application modernisée, vous devrez peut-être configurer une ou plusieurs applications Web facultatives qui représentent la prise en charge de dépendances telles que z/OS, AS/400, ou des IMS/MFS. The following tables contain lists of the available key/value paramètres de configuration de chaque application Web facultative.

gapwalk-utility-pgm.guerre

Cette application Web facultative prend en charge les programmes utilitaires Z/OS.

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs pour cette application.

Clé	Type	Valeur par défaut	Description	Version de sortie
`logging.config`	Chemin	chemin de classe : logback-utility.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalisation standard sont également disponibles.	
`spring.jta.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa , la configuration automatique des transactions Spring JTA doit être désactivée.	
`spring.datasource.primary.jndi-name`	chaîne	jdbc/primaire	Le nom JNDI (Java Naming And Directory Interface) de la source de données principale, si	

Clé	Type	Valeur par défaut	Description	Version de sortie
			vous utilisez JNDI.	
`primary.datasource -driver-class-name -url -username -password`	Source de données Spring standard avec sous-clés		Contient les informations de connexion pour la base de données de l'application, si vous n'utilisez pas JNDI. Doit avoir la même configuration que dans le fichier YAML de l'application modernisée. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dansthe section called "Base de données clients".	
`encoding`	chaîne	ASCII	Codage utilisé dans les programmes utilitaires. Exige un encodage valide `CP1047IBM930,,A`	

Clé	Type	Valeur par défaut	Description	Version de sortie
`sysPunchEncoding`	chaîne	ASCII	Le jeu de caractères de codage Syspunch. Exige un encodage valide `CP1047IBM930,,A`	
`systin.encoding`	chaîne	ASCII	Le jeu de caractères de codage de l'ensemble de données de fichiers SYSTIN. Exige un encodage valide `CP1047IBM930,,A`	4.5.0
`zonedMode`	chaîne	EBCDIC_STRICT	Mode de codage ou de décodage des types de données zonés. Les valeurs autorisées sont`EBCDIC_STRICT /EBCDIC_MODIFIED /AS400`.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`idcams.encoding.forced`	chaîne		Codage utilisé dans le programme utilitaire IDCAMS. Exige un encodage valide `CP1047IBM930,,A`	4.4.0
`unload.chunkSize`	nombre	0	Taille du morceau utilisée pour l'utilitaire de déchargement.	
`unload.computeRecordSizeIfNull`	boolean	false	Détermine s'il faut calculer la taille de l'enregistrement si elle n'est pas spécifiée. Si elle est spécifiée, la valeur reste inchangée.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`unload.sqlCodePointShift`	nombre	0	L'utilitaire SQL Pointshift for Unload. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible DB2 provient de Postgresql.	
`unload.columnFiller`	chaîne	espace	Le remplisseur de colonnes utilitaire de déchargement.	
`unload.varCharIsNull`	boolean	false	Utilisez ce paramètre dans le programme INFTILB. S'il est défini sur, tous les champs non nullables contenant des valeurs vides (espaces) renvoient une chaîne vide. `true`	

Clé	Type	Valeur par défaut	Description	Version de sortie
`unload.useDatabaseConfiguration`	boolean	false	Spécifie s'il faut utiliser la configuration de date ou d'heure de application-main.yml dans l'utilitaire de déchargement.	
`unload.format.date`	chaîne	MM/dd/yyyy	Si cette option `unload.useDatabaseConfiguration` est activée, le format de date à utiliser dans l'utilitaire de déchargement.	
`unload.format.time`	chaîne	HH.mm.ss	Si cette option `unload.useDatabaseConfiguration` est activée, le format d'heure à utiliser dans l'utilitaire de déchargement.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`unload.format.timestamp`	chaîne	yyyy-MM-dd-HH.mm.ss.ssssss	Si cette option `unload.useDatabaseConfiguration` est activée, le format d'horodatage à utiliser dans l'utilitaire de déchargement.	
`unload.nbi.whenNull`	hexadécimal	6F	La valeur NBI (Null Byte Indicator) à ajouter lorsque la valeur de la base de données est nulle.	
`unload.nbi.whenNotNull`	hexadécimal	00	La valeur NBI (Null Byte Indicator) à ajouter lorsque la valeur de la base de données n'est pas nulle.	
`unload.nbi.writeNullIndicator`	boolean	false	Spécifie s'il faut écrire l'indicateur nul dans le fichier de sortie de déchargement.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`unload.bmc.useInto`	boolean	false	Spécifie s'il faut gérer le mot clé de contrôle INTO bmc pour l'utilitaire de déchargement.	
`unload.fetchSize`	nombre	0	Vous permet de régler la taille de lecture lorsque vous manipulez des curseurs dans l'utilitaire de déchargement.	
`unload.noPad`	booléen	true	Indique que les champs de caractères de longueur variable (VARCHAR) doivent être déchargés sans que la longueur maximale soit atteinte.	4.5.0
`treatLargeNumberAsInteger`	boolean	false	Spécifie s'il faut traiter les grands nombres comme`Integer`. Ils sont traités comme `BigDecimal` par défaut.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`load.batchSize`	nombre	0	Taille du lot de l'utilitaire de chargement.	
`load.format.localDate`	chaîne	dd.mm.yyyy \ \ yyyy-mm-dd dd/MM/yyyy	Le format de date local de l'utilitaire de chargement à utiliser.	
`load.format.localTime`	chaîne	HH : mm : SS \ HH.mm.ss	Format d'heure locale de l'utilitaire de chargement à utiliser.	
`load.format.dbDate`	chaîne	yyyy-MM-dd	Format de base de données de l'utilitaire de chargement à utiliser.	
`load.format.dbTime`	chaîne	HH : mm : SS	Durée d'utilisation de la base de données de l'utilitaire de chargement.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`load.sqlCodePointShift`	nombre	0s	L'utilitaire SQL Pointshift for Load. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible DB2 provient de Postgresql.	
`load.applyRollback`	boolean	false	Définissez ce paramètre sur `true` pour indiquer que vous souhaitez que le service annule les modifications de table s'il rencontre une erreur lors du chargement des données dans la base de données.	
`forcedDate`	chaîne		Force la date à la date indiquée s'il y en a une.	

Clé	Type	Valeur par défaut	Description	Version de sortie
`frozenDate`	booléen	true	Spécifie s'il faut geler la date. S'applique uniquement s'`forcedDate` il est également défini.	
`jcl.type`	chaîne	mvs	Type de fichier .jcl. Les valeurs autorisées sont `jcl`/`vse`. Les commandes PRINT/REPRO de l'utilitaire IDCAMS renvoient 4 si le fichier est vide pour un jcl non vse.	
`hasGraphic`	boolean	false	Si l'utilitaire INFUTILB doit gérer les colonnes GRAPHIC DB2 .	
`convertGraphicDataToFullWidth`	booléen	true	Spécifie s'il faut convertir les données graphiques au format pleine largeur.	

gapwalk-cl-command.guerre

Cette application Web optionnelle prend en charge les programmes utilitaires AS/400.

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs pour cette application.

Clé	Type	Valeur par défaut	Description
`logging.config`	Chemin	chemin de classe : logback-utility.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalisation standard sont également disponibles.
`spring.jta.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa, la configuration automatique des transactions Spring JTA doit être désactivée.
`spring.datasource.primary.jndi-name`	chaîne	jdbc/primaire	Le nom JNDI (Java Naming And Directory Interface) de la source de données principale, si vous utilisez JNDI.
`primary.datasource + -driver-class-name + -url`	Source de données Spring standard avec sous-clés		Contient les informations de connexion pour la base de données de l'applica

Clé	Type	Valeur par défaut	Description
+ `-username` + `-password`			tion, si vous n'utilisez pas JNDI. Doit avoir la même configuration que dans le fichier YAML de l'application modernisée. Par ailleurs, l'utilisation de AWS secrets est fortement encouragée, comme expliqué dansthe section called "Base de données clients".
`encoding`	chaîne	ASCII	Codage utilisé dans les programmes utilitaires. Exige un encodage valide `CP1047IBM930,,ASCII,UTF-8`
`zonedMode`	chaîne	EBCDIC_STRICT	Mode de codage ou de décodage des types de données zonés. Les valeurs autorisées sont`EBCDIC_STRICT /EBCDIC_MODIFIED /AS400`.

Clé	Type	Valeur par défaut	Description
`commands-off`	chaîne		Liste des commandes à désactiver, séparées par des virgules. Les valeurs autorisées sont `PGM_BASIC RCVMSG,SNDRCVF,CHGVAR,QC` Utile lorsque vous souhaitez désactiver ou remplacer un programme existant. `PGM_BASIC` est un programme AWS Blu Age Runtime spécifique conçu à des fins de débogage.
`forcedDate`	chaîne		Force la date à la date indiquée s'il y en a une.

gapwalk-hierarchical-support.guerre

Cette application Web optionnelle prend en charge les transactions IMS/MFS.

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs pour cette application.

Clé	Type	Valeur par défaut	Description
`logging.config`	Chemin	chemin de classe : logback-utility.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalisation standard sont

Clé	Type	Valeur par défaut	Description
			également disponibles.
`spring.jta.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa, la configuration automatique des transactions Spring JTA doit être désactivée.
`jhdb.configuration.context.encoding`	chaîne		Le codage JHDB (base de données hiérarchique Java). Exige une chaîne de codage valide`CP1047`,`IBM930`,`ASCII`,

Clé	Type	Valeur par défaut	Description
`jhdb.checkpointPersistence`	chaîne	none	Le mode de persistance du point de contrôle. Les valeurs autorisées sont`none`/`add`/`end`. `add`À utiliser pour conserver les points de contrôle lorsqu'un nouveau point est créé et ajouté au registre. Utilisez un point de contrôle `end` trop persistant lors de l'arrêt du serveur. Toute autre valeur désactive la persistance. Notez que chaque fois qu'un nouveau point de contrôle est ajouté au registre, tous les points de contrôle existants sont sérialisés et le fichier est effacé. Il ne s'agit pas d'un ajout aux données existantes du fichier. Ainsi, en fonction du nombre de points de contrôle, cela peut avoir des effets sur les performances.

Propriétés disponibles pour l'application cliente

Votre application modernisée peut nécessiter des configurations de propriétés spécifiques pour l'application Spring cliente. Ces propriétés initialisent les beans à partir de classes contenues dans des fichiers JAR d'exécution. Le `application-profile.yaml` fichier, dans lequel la valeur du profil est définie lors de la génération de l'application, vous permet de configurer ces propriétés. Le tableau suivant répertorie les paramètres clé/valeur disponibles pour configurer l'application Web cliente qui utilise des beans issus de classes empaquetées dans le runtime Gapwalk

Clé	Type	Valeur par défaut	Description	Version de sortie
`blu4iv.dtaara.library.disable`	boolean	false	Contrôle l'utilisation de la bibliothèque dans le contexte des opérations de zone de données. Si ce paramètre est défini sur true, l'utilisation de la bibliothèque est désactivée pour les opérations de zone de données, mais cela n'affecte pas l'utilisation de QTemp. Si ce paramètre est défini sur false, la bibliothèque est prise en compte lors de l'exécution des opérations CRUD pour	4.5.0

Clé	Type	Valeur par défaut	Description	Version de sortie
			la zone de données.	

Propriétés du cache Redis disponibles dans AWS Blu Age Runtime

Vous pouvez utiliser ce document pour en savoir plus sur les caches Redis dans AWS Blu Age Runtime, ainsi que sur la configuration de Gapwalk, les propriétés Redis prises en charge et sur la façon dont le fichier `application-main.yml` peut référencer l'ARN secret pour les caches Redis.

Caches Redis dans AWS Blue Age Runtime

Les serveurs Redis peuvent être utilisés comme caches pour diverses fonctionnalités de l'application AWS Blu Age Gapwalk, telles que :

AWS Fonctionnalités de Blu Age Runtime qui utilisent la mise en cache Redis	Description
Cache Blusam	Un cache Redis Blusam pour lire efficacement les enregistrements, à l'aide d'une stratégie d'écriture différée, afin d'optimiser les charges de travail intensives en écriture rencontrées sur les charges utiles par lots.
Serrures Blusam	Un cache pour les verrous distribués pour les ensembles de données et les enregistrements.
Catalogue de jeux de données	Le cache du jeu de données du catalogue.
Cache de session	Un cache Redis pour. HttpSession Le cache stocke le nom d'utilisateur, l'état du dialogue avec le frontend Angular et des informations spécifiques sur le « dialecte » (BMS, MFS, AS4 00).

AWS Fonctionnalités de Blu Age Runtime qui utilisent la mise en cache Redis	Description
Suivi de session	Un cache de sessions actives avec le nom d'utilisateur et les session-creation-time informations associés.
Cache JICS	Un cache pour les définitions de ressources JICS.
files d'attente TS	Stockage pour les files d'attente TS.
Point de contrôle JCL	Cache de point de contrôle JCL.
Verrouillages de fichiers Gapwalk	Un cache pour les verrouillages de fichiers distribués par tâche.
Serrures Blu4iv	Stockage pour les verrous d'enregistrement Blu4IV.

Configuration de Redis Gapwalk

La configuration Redis globale est utilisée si elle `redis` est spécifiée comme mécanisme de mise en cache et qu'aucune configuration Redis n'est fournie pour la fonctionnalité spécifique. Cette configuration vous permet d'utiliser la même configuration pour plusieurs caches Redis simultanément.

Dans l'exemple suivant, le cache des ensembles de données Blusam et le cache JICS utilisent la configuration `gapwalk.redis` (`redis.server1`) car leur type de cache est défini sur et aucune propriété Redis implicite n'est spécifiée sous et. `redis` the section called “Définitions des ressources JICS” the section called “Définitions des ressources JICS” Cependant, le cache des verrous Blusam utilisera une configuration Redis différente (`redis.server2`) car ses propriétés Redis sont définies de manière explicite.

```
...

 gapwalk:
   redis:
     hostName: redis.server1
```

```
 port: 6379
...

bluesam:
  # Redis bluesam cache
  cache: redis
  # Redis locks cache
  locks:
    cache: redis
 hostName: redis.server2
 port: 6379
...
# Redis jics cache
jics:
  resource-definitions:
    store-type: redis
...
```

Pour activer la configuration globale de Redis, ajoutez la configuration suivante dans`main-application.yml`.

```
gapwalk:
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
```

```
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Propriétés Redis prises en charge

Le tableau suivant indique les propriétés Redis prises en charge pour les caches Redis globaux et spécifiques sur AWS Blu Age Runtime.

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
`mode`	Non	Le mode de fonctionnement de Redis.	`standalone | cluster`	`standalone`
`hostname`	Oui	Le nom d'hôte ou l'adresse IP du serveur Redis.	chaîne	null
`port`	Oui	Le numéro de port sur lequel le serveur Redis écoute les connexions.	int	null
`username`	Non	Le nom d'utilisateur pour l'authentification.	chaîne	null
`password`	Non	Le mot de passe pour l'authentification.	chaîne	chaîne vide
`useSsl`	Non	Spécifie s'il faut activer le chiffrement SSL/	boolean	false

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
		TLS pour la connexion Redis.		
`database`	Non	Le numéro de base de données Redis à utiliser. Redis prend en charge plusieurs bases de données logiques, et cette propriété indique laquelle utiliser.	int	0
`maxTotal`	Non	Le nombre maximum de connexion s autorisées dans le pool de connexions Redis.	int	128
`maxIdle`	Non	Le nombre maximum de connexion s inactives autorisées dans le pool de connexions Redis.	int	128

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
`minIdle`	Non	Le nombre minimum de connexions inactives à maintenir dans le pool de connexions Redis.	int	16
`testOnBorrow`	Non	Valeur booléenne indiquant s'il faut valider les connexions avant de les emprunter au pool.	booléen	true
`testOnReturn`	Non	Valeur booléenne indiquant s'il faut valider les connexions avant de les renvoyer au pool.	booléen	true

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
`testWhileIdle`	Non	Une valeur booléenne indiquant s'il faut valider régulièrement les connexions inactives dans le pool.	booléen	true
`testOnCreate`	Non	Valeur booléenne indiquant si les connexions doivent être validées lors de leur création.	booléen	true
`minEvictableIdleTimeMillis`	Non	Durée minimale (en millisecondes) pendant laquelle une connexion inactive doit rester dans le pool avant de pouvoir être expulsée.	long	60000 L

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
`timeBetweenEvictionRunsMillis`	Non	Durée (en millisecondes) entre les exécutions successives du thread d'éviction de connexion inactif.	long	30 000 L
`numTestsPerEvictionRun`	Non	Le nombre maximum de connexions à tester lors de chaque exécution du thread d'éviction de connexion inactif.	int	-1
`blockWhenExhausted`	Non	Valeur booléenne indiquant s'il faut bloquer et attendre qu'une connexion soit disponible lorsque le pool est épuisé.	booléen	true

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
`nettyThreads`	Non	Le nombre de threads Netty à utiliser pour gérer les connexions Redis.	int	32
`subscriptionsPerConnection`	Non	Le nombre maximum d'abonnements autorisés par connexion Redis.	int	10
`subscriptionConnectionPoolSize`	Non	Le nombre maximum de connexions autorisées dans le pool de connexions d'abonnement Redis.	int	100
`pageSizeInBytes`	Non	Taille de page par défaut en octets pour les opérations Redis.	long	262144000
`readTimeout`	Non	Le délai de lecture en millisecondes pour les opérations Redis.	long	2000

Nom de la propriété	Obligatoire ?	Description	Valeurs	Par défaut
`timeToLiveMillis`	Non	Durée (en millisecondes) pendant laquelle une entrée de cache reste dans le cache avant d'être considérée comme expirée et supprimée. Si cette propriété n'est pas spécifiée, les entrées du cache n'expireront pas automatiquement par défaut.	long	-1

Propriétés du cache Redis

Cache Redis Blusam

```
bluesam:
   cache: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostName: localhost
     port: 6379
     mode: standalone                        # Optional
     username:                               # Optional
     password: ""                            # Optional
     useSsl: false                           # Optional
     database: 0                             # Optional
     maxTotal: 128                           # Optional
```

```
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
    timeToLiveMillis: 60000                 # Optional
```

Cache Redis Blusam

```
bluesam:
   locks:
     cache: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
       hostName: localhost
       port: 6379
       mode: standalone                       # Optional
       username:                              # Optional
       password: ""                           # Optional
       useSsl: false                          # Optional
       database: 0                            # Optional
       maxTotal: 128                          # Optional
       maxIdle: 128                           # Optional
       minIdle: 16                            # Optional
       testOnBorrow: true                     # Optional
       testOnReturn: true                     # Optional
       testWhileIdle: true                    # Optional
       testOnCreate: true                     # Optional
       minEvictableIdleTimeMillis: 60000      # Optional
       timeBetweenEvictionRunsMillis: 30000   # Optional
       numTestsPerEvictionRun: -1             # Optional
       blockWhenExhausted: true               # Optional
       nettyThreads: 32                       # Optional
```

```
      subscriptionsPerConnection: 10          # Optional
      subscriptionConnectionPoolSize: 100     # Optional
      pageSizeInBytes: 8192                   # Optional
      readTimeout: 2000                       # Optional
```

Cache de session

```
spring:
  session:
    store-type: redis
# If the following redis properties are not specified gapwalk.redis configuration will
be used for this cache
jics:
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Définitions des ressources JICS

```
jics:
   resource-definitions:
```

```
    store-type: redis
# If the following redis properties are not specified gapwalk.redis configuration will
be used for this cache
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

files d'attente JICS TS

```
jics:
   parameters:
     tsqimpl: redis
# If the following redis properties are not specified gapwalk.redis configuration will
be used for this cache
  queues:
    ts:
      redis:
        hostName: localhost
        port: 6379
        mode: standalone                        # Optional
        username:                               # Optional
        password: ""                            # Optional
```

```
        useSsl: false                          # Optional
        database: 0                            # Optional
        maxTotal: 128                          # Optional
        maxIdle: 128                           # Optional
        minIdle: 16                            # Optional
        testOnBorrow: true                     # Optional
        testOnReturn: true                     # Optional
        testWhileIdle: true                    # Optional
        testOnCreate: true                     # Optional
        minEvictableIdleTimeMillis: 60000      # Optional
        timeBetweenEvictionRunsMillis: 30000   # Optional
        numTestsPerEvictionRun: -1             # Optional
        blockWhenExhausted: true               # Optional
        nettyThreads: 32                       # Optional
        subscriptionsPerConnection: 10         # Optional
        subscriptionConnectionPoolSize: 100    # Optional
        pageSizeInBytes: 8192                  # Optional
        readTimeout: 2000                      # Optional
```

Suivi de session

```
session-tracker:
   store-type: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostName: localhost
     port: 6379
     mode: standalone                       # Optional
     username:                              # Optional
     password: ""                           # Optional
     useSsl: false                          # Optional
     database: 0                            # Optional
     maxTotal: 128                          # Optional
     maxIdle: 128                           # Optional
     minIdle: 16                            # Optional
     testOnBorrow: true                     # Optional
     testOnReturn: true                     # Optional
     testWhileIdle: true                    # Optional
     testOnCreate: true                     # Optional
     minEvictableIdleTimeMillis: 60000      # Optional
     timeBetweenEvictionRunsMillis: 30000   # Optional
     numTestsPerEvictionRun: -1             # Optional
```

```
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Point de contrôle JCL

```
jcl:
   checkpoint:
     provider: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostname: localhost
     port: 6379
     mode: standalone                       # Optional
     username:                              # Optional
     password: ""                           # Optional
     useSsl: false                          # Optional
     database: 0                            # Optional
     maxTotal: 128                          # Optional
     maxIdle: 128                           # Optional
     minIdle: 16                            # Optional
     testOnBorrow: true                     # Optional
     testOnReturn: true                     # Optional
     testWhileIdle: true                    # Optional
     testOnCreate: true                     # Optional
     minEvictableIdleTimeMillis: 60000      # Optional
     timeBetweenEvictionRunsMillis: 30000   # Optional
     numTestsPerEvictionRun: -1             # Optional
     blockWhenExhausted: true               # Optional
     nettyThreads: 32                       # Optional
     subscriptionsPerConnection: 10         # Optional
     subscriptionConnectionPoolSize: 100    # Optional
     pageSizeInBytes: 8192                  # Optional
     readTimeout: 2000                      # Optional
```

Verrouillages de fichiers Gapwalk

```
filesLocks:
   enabled: true
```

```
  retryTime: 1000
  MaxRetry: 5
  provider: redis
# If the following redis properties are not specified gapwalk.redis configuration will
be used for this cache
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                          # Optional
    username:                                 # Optional
    password: ""                              # Optional
    useSsl: false                             # Optional
    database: 0                               # Optional
    pool:
      maxTotal: 128                           # Optional
      maxIdle: 128                            # Optional
      minIdle: 16                             # Optional
      testOnBorrow: true                      # Optional
      testOnReturn: true                      # Optional
      testWhileIdle: true                     # Optional
      testOnCreate: true                      # Optional
      minEvictableIdleTimeMillis: 60000       # Optional
      timeBetweenEvictionRunsMillis: 30000    # Optional
      numTestsPerEvictionRun: -1              # Optional
      blockWhenExhausted: true                # Optional
      nettyThreads: 32                        # Optional
      subscriptionsPerConnection: 10          # Optional
      subscriptionConnectionPoolSize: 100     # Optional
      pageSizeInBytes: 8192                   # Optional
      readTimeout: 2000                       # Optional
```

Serrures Blu4iv

```
blu4iv.lock: redis
blu4iv.lock.timeout: 10 #(in millisecondes)
 # If the following redis properties are not specified gapwalk.redis configuration
will be used for this cache
blu4iv.lock.redis:
      hostName: localhost
      port: 6379
      mode: standalone                        # Optional
      username:                               # Optional
```

```
      password: ""                          # Optional
      useSsl: false                         # Optional
      database: 0                           # Optional
      maxTotal: 128                         # Optional
      maxIdle: 128                          # Optional
      minIdle: 16                           # Optional
      testOnBorrow: true                    # Optional
      testOnReturn: true                    # Optional
      testWhileIdle: true                   # Optional
      testOnCreate: true                    # Optional
      minEvictableIdleTimeMillis: 60000     # Optional
      timeBetweenEvictionRunsMillis: 30000  # Optional
      numTestsPerEvictionRun: -1            # Optional
      blockWhenExhausted: true              # Optional
      nettyThreads: 32                      # Optional
      subscriptionsPerConnection: 10        # Optional
      subscriptionConnectionPoolSize: 100   # Optional
      pageSizeInBytes: 8192                 # Optional
      readTimeout: 2000                     # Optional
```

Catalogue de jeux de données

```
datasimplifier:
   catalogImplementation: redis
   # If the following redis properties are not specified gapwalk.redis configuration
 will be used for this cache
   redis:
     hostName: localhost
     port: 6379
     mode: standalone                       # Optional
     username:                              # Optional
     password: ""                           # Optional
     useSsl: false                          # Optional
     database: 0                            # Optional
     maxTotal: 128                          # Optional
     maxIdle: 128                           # Optional
     minIdle: 16                            # Optional
     testOnBorrow: true                     # Optional
     testOnReturn: true                     # Optional
     testWhileIdle: true                    # Optional
     testOnCreate: true                     # Optional
     minEvictableIdleTimeMillis: 60000      # Optional
     timeBetweenEvictionRunsMillis: 30000   # Optional
```

```
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Gestionnaire secret pour les caches Redis

Le `application-main.yaml` fichier peut référencer l'ARN secret des caches Redis. Pour plus d'informations sur la manière d'intégrer AWS Secrets Manager afin de récupérer en toute sécurité les détails de connexion Redis lors de l'exécution, consultezthe section called "AWS Les secrets de Blu Age Runtime".

Configuration de la sécurité pour les applications Gapwalk

Les rubriques suivantes décrivent comment sécuriser les applications Gapwalk.

Il est de votre responsabilité de fournir la bonne configuration pour garantir que l'utilisation du framework AWS Blu Age est sécurisée.

Toutes les fonctionnalités liées à la sécurité sont désactivées par défaut. Pour activer l'authentification (et CSRF, XSS, CSP, etc.), définissez `gapwalk-application.security` sur et sur. `enabled` `gapwalk-application.security.identity` `oauth`

Rubriques

- Configurer l'accessibilité des URI pour les applications Gapwalk
- Configurer l'authentification pour les applications Gapwalk

Configurer l'accessibilité des URI pour les applications Gapwalk

Cette rubrique décrit comment configurer le filtrage URIs des applications Gapwalk. Cette fonctionnalité ne nécessite pas de fournisseur d'identité (IdP).

Pour bloquer une liste de URIs, ajoutez les deux lignes suivantes à celle `application-main.yml` de votre application modernisée*`URI-1`*, en remplaçant*`URI-2`*, etc., par celle URIs que vous souhaitez bloquer.

```
gapwalk-application.security.filterURIs: enabled
gapwalk-application.security.blockedURIs: URI-1, URI-2, URI-3
```

Configurer l'authentification pour les applications Gapwalk

Pour configurer OAuth2 l'authentification pour votre application Gapwalk, vous devez configurer un fournisseur d'identité (IdP) et l'intégrer à votre application. Ce guide décrit les étapes à suivre pour utiliser Amazon Cognito ou Keycloak comme IdP. Avec Amazon Cognito, vous pouvez mettre à jour le fichier de configuration de votre application avec les détails du groupe d'utilisateurs Cognito. Avec Keycloak, vous pouvez contrôler l'accès à votre application APIs et à vos ressources en fonction des rôles assignés à l'utilisateur.

Rubriques

- Configurer l' OAuth2 authentification Gapwalk avec Amazon Cognito
- Configurer l' OAuth2 authentification Gapwalk avec Keycloak

Configurer l' OAuth2 authentification Gapwalk avec Amazon Cognito

Cette rubrique explique comment configurer l' OAuth2 authentification pour les applications Gapwalk utilisant Amazon Cognito en tant que fournisseur d'identité (IdP).

Prérequis

Dans ce didacticiel, nous utiliserons Amazon Cognito comme IdP et PlanetDemo comme projet modernisé.

Vous pouvez utiliser n'importe quel autre fournisseur d'identité externe. Les ClientRegistration informations doivent être obtenues auprès de votre IdP et sont requises pour l'authentification Gapwalk. Pour plus d'informations, consultez le Guide du développeur Amazon Cognito.

Les ClientRegistration informations :

identificateur du client

ID du ClientRegistration. Dans notre exemple, ce sera le cas PlanetsDemo.

client-secret

Le secret de votre client.

point final d'autorisation

L'URI du point de terminaison d'autorisation pour le serveur d'autorisation.

point de terminaison symbolique

L'URI du point de terminaison du jeton pour le serveur d'autorisation.

point de terminaison jwks

L'URI utilisé pour obtenir la clé Web JSON (JWK) qui contient les clés permettant de valider la signature Web JSON émise par le serveur d'autorisation.

URI de redirection

L'URI vers lequel le serveur d'autorisation redirige l'utilisateur final si l'accès est accordé.

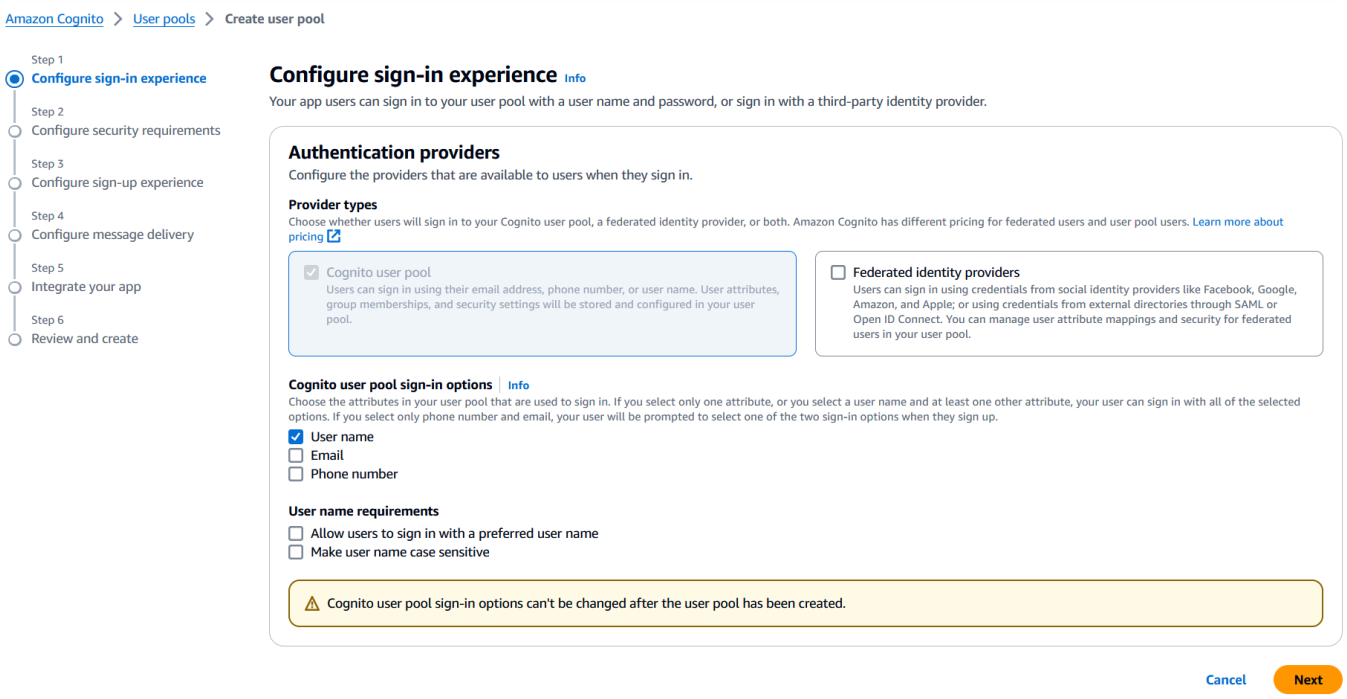
Configuration d'Amazon Cognito

Nous allons d'abord créer et configurer un groupe d'utilisateurs et un utilisateur Amazon Cognito que nous utiliserons avec notre application Gapwalk déployée à des fins de test.

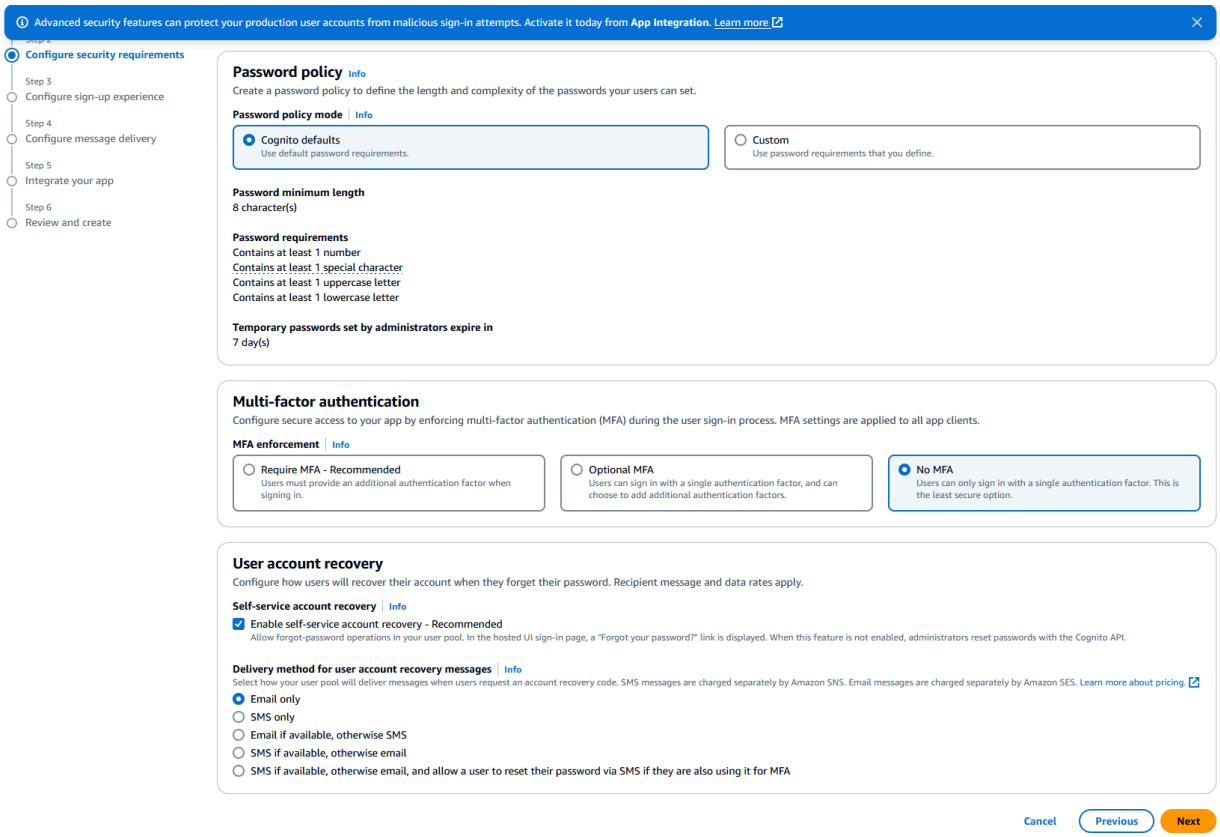
> **Note**
>
> Si vous utilisez un autre IdP, vous pouvez ignorer cette étape.

Création d'un groupe d'utilisateurs

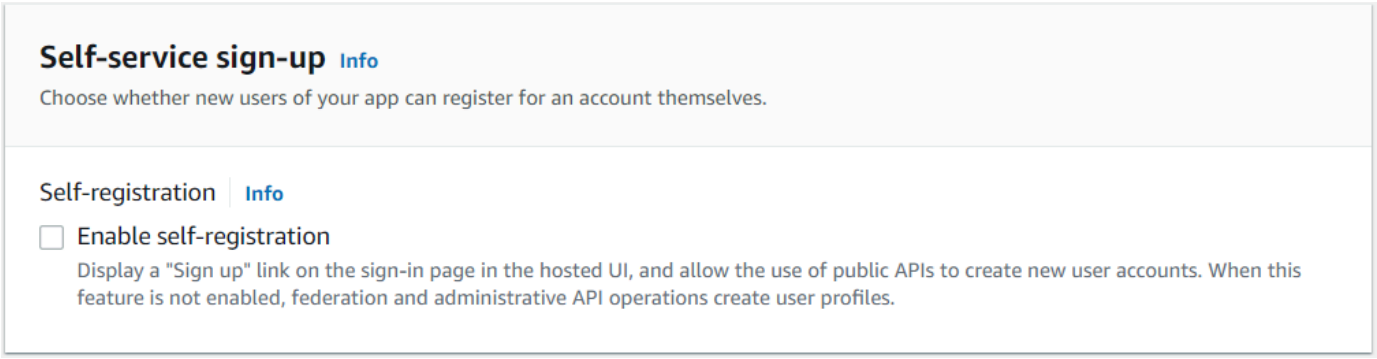
1. Accédez à Amazon Cognito dans le AWS Management Console et authentifiez-vous à l'aide de vos informations d'identification. AWS
2. Choisissez Groupes d’utilisateurs.
3. Sélectionnez Create a user pool.
4. Dans Configurer l'expérience de connexion, conservez le type de fournisseur par défaut du groupe d'utilisateurs Cognito. Vous pouvez choisir une ou plusieurs options de connexion au groupe d'utilisateurs de Cognito ; pour l'instant, choisissez Nom d'utilisateur, puis Suivant.



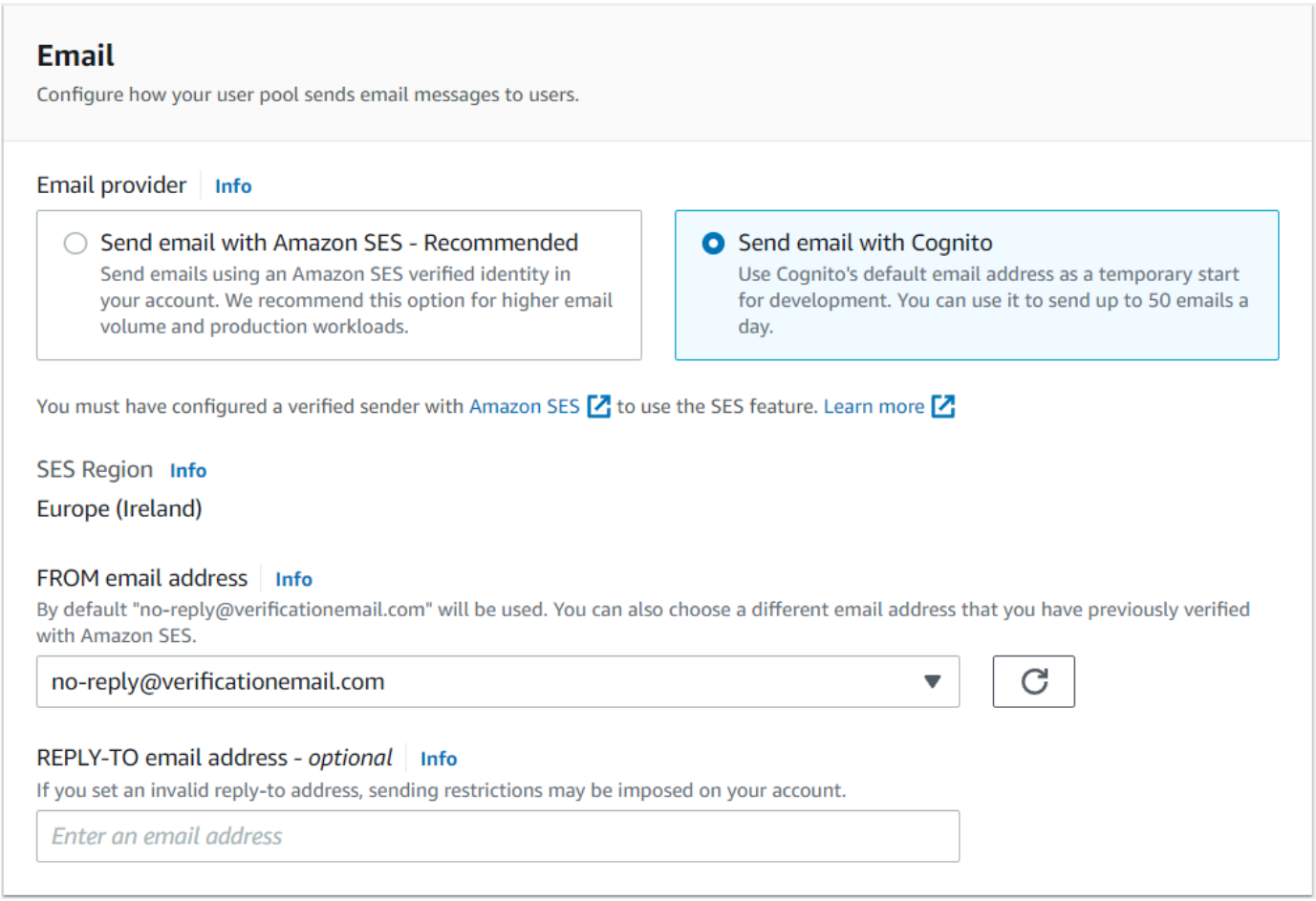
5. Dans Configurer les exigences de sécurité, conservez les valeurs par défaut et désactivez l'authentification multifactorielle en choisissant Pas de MFA, puis en choisissant Suivant.



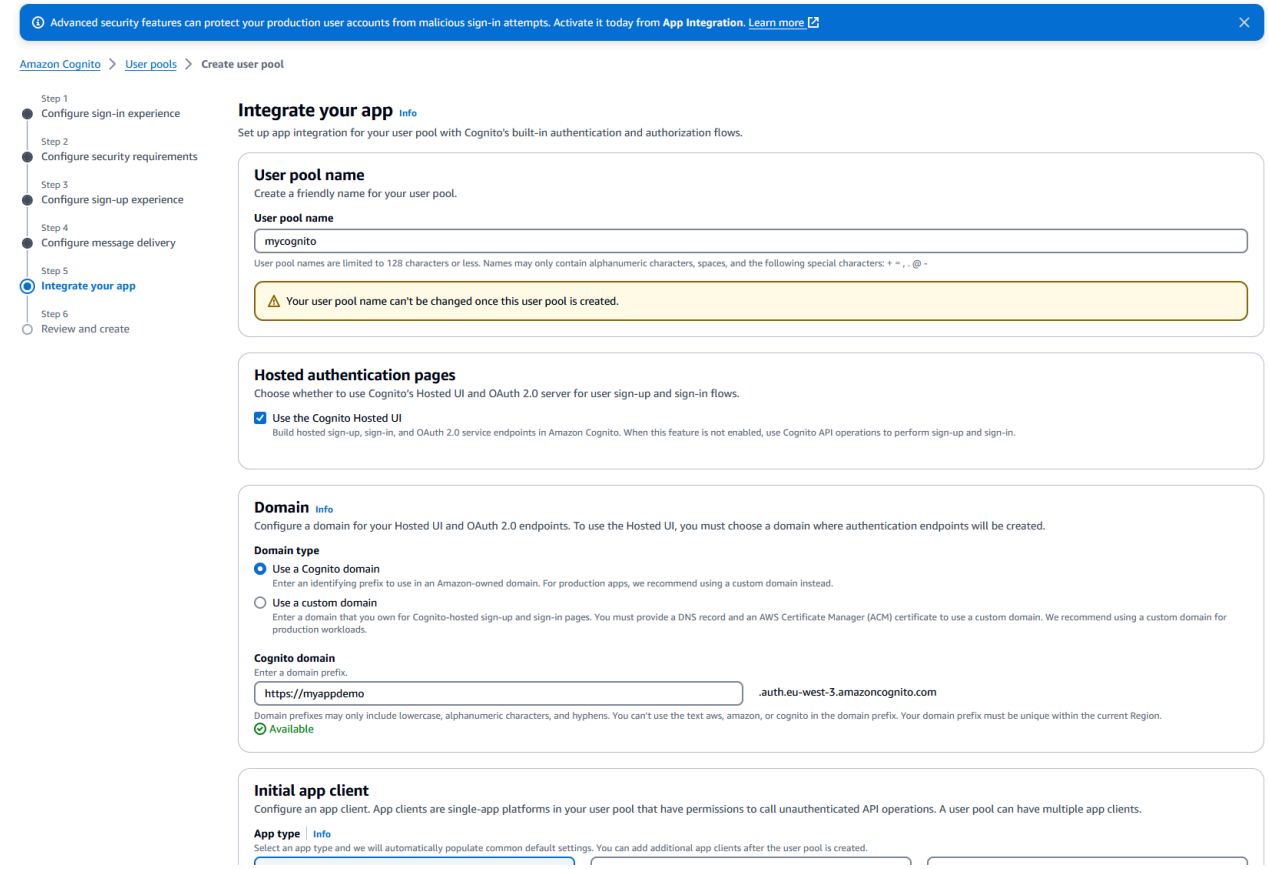
6. Par mesure de sécurité, désactivez Activer l'enregistrement automatique, puis choisissez Suivant.



7. Choisissez Envoyer un e-mail avec Cognito, puis cliquez sur Suivant.



8. Dans Intégrer votre application, spécifiez le nom de votre groupe d'utilisateurs. Dans les pages d'authentification hébergées, choisissez Utiliser l'interface utilisateur hébergée de Cognito.



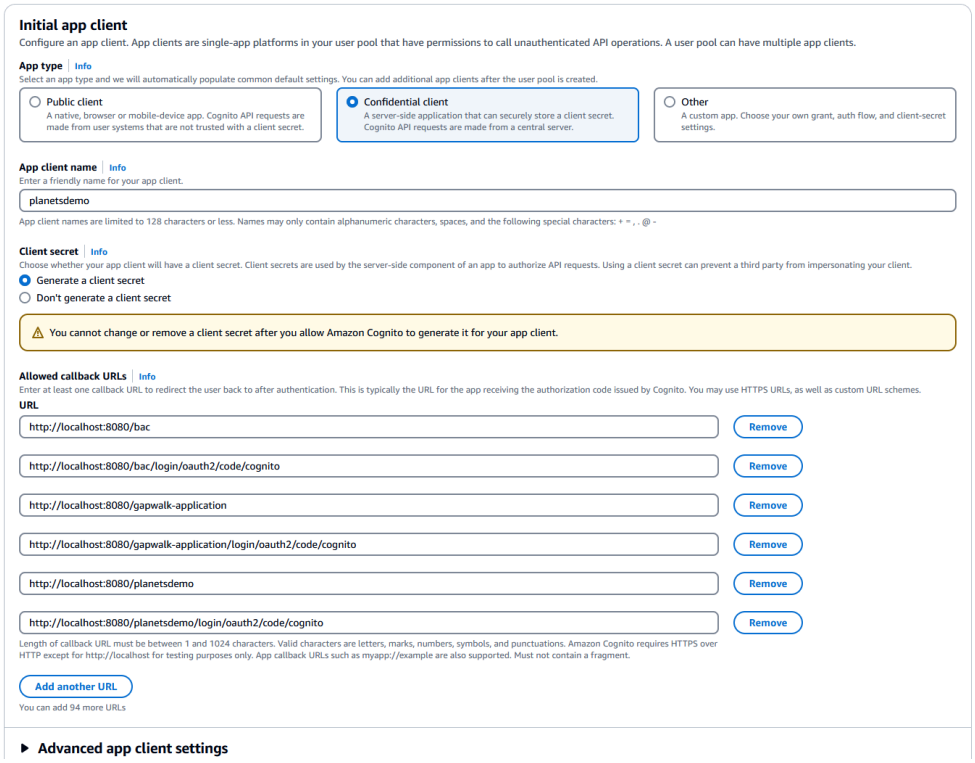
9. Pour plus de simplicité, dans Domaine, choisissez Utiliser un domaine Cognito et entrez un préfixe de domaine, par exemple,. `https://planetsdemo` L'application de démonstration doit être ajoutée en tant que client.

   a. Dans Client d'application initial, choisissez Client confidentiel. Entrez un nom de client d'application, tel que**`planetsdemo`**, puis choisissez Generate a client secret.

   b. Dans URL de rappel autorisée, entrez l'URL vers laquelle rediriger l'utilisateur après l'authentification. L'URL doit se terminer par`/login/oauth2/code/cognito`. Par exemple, pour nos applications d'application et de backend Gapwalk et BAC :

   ```
   http://localhost:8080/bac
         http://localhost:8080/bac/login/oauth2/code/cognito
         http://localhost:8080/gapwalk-application
         http://localhost:8080/gapwalk-application/login/oauth2/code/cognito
         http://localhost:8080/planetsdemo
         http://localhost:8080/planetsdemo/login/oauth2/code/cognito
   ```

   Vous pourrez modifier l'URL ultérieurement.



c. Dans Déconnexion autorisée, URLs entrez l'URL de la page de déconnexion vers laquelle vous souhaitez qu'Amazon Cognito redirige lorsque votre application déconnecte les utilisateurs. Par exemple, pour les applications principales Gapwalk et BAC :

```
http://localhost:8080/bac/logout
http://localhost:8080/gapwalk-application/logout
http://localhost:8080/planetsdemo/logout
```

Vous pourrez modifier l'URL ultérieurement.

d. Conservez les valeurs par défaut dans les sections Paramètres avancés du client de l'application et Autorisations de lecture et d'écriture des attributs.

e. Choisissez Next (Suivant).

10. Dans Révision et création, vérifiez vos choix, puis choisissez Créer un groupe d'utilisateurs.

Pour plus d'informations, voir Création d'un groupe d'utilisateurs.

Création d'utilisateurs

L'enregistrement automatique étant désactivé, créez un utilisateur Amazon Cognito. Accédez à Amazon Cognito dans le. AWS Management Console Choisissez le groupe d'utilisateurs que vous avez créé, puis dans Utilisateurs, choisissez Créer un utilisateur.

Dans Informations utilisateur, choisissez Envoyer une invitation par e-mail, entrez un nom d'utilisateur et une adresse e-mail, puis choisissez Générer un mot de passe. Choisissez Create user (Créer un utilisateur).

Création de rôles

Dans l'onglet Groupes, créez 3 groupes (SUPER_ADMIN, ADMIN et USER) et associez votre utilisateur à un ou plusieurs de ces groupes. Ces rôles sont ensuite mappés à ROLE_SUPER_ADMIN, ROLE_ADMIN et ROLE_USER par l'application Gapwalk pour permettre l'accès à certains appels REST d'API restreints.

Intégrer Amazon Cognito dans l'application Gapwalk

Maintenant que votre groupe d'utilisateurs et vos utilisateurs Amazon Cognito sont prêts, accédez au `application-main.yml` fichier de votre application modernisée et ajoutez le code suivant :

```
gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
gapwalk-application.security.issuerUri: https://cognito-idp.<region-id>.amazonaws.com/
<pool-id>
gapwalk-application.security.domainName: <your-cognito-domain>
gapwalk-application.security.localhostWhitelistingEnabled: false

spring:
  security:
    oauth2:
      client:
        registration:
          cognito:
            client-id: <client-id>
            client-name: <client-name>
            client-secret: <client-secret>
            provider: cognito
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "<redirect-uri>"
        provider:
          cognito:
            issuer-uri: ${gapwalk-application.security.issuerUri}
```

```
            authorization-uri: ${gapwalk-application.security.domainName}/oauth2/
authorize
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/
jwks.json
            token-uri: ${gapwalk-application.security.domainName}/oauth2/token
            user-name-attribute: username
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/jwks.json
```

Remplacez les espaces réservés suivants comme décrit :

1. Accédez à Amazon Cognito dans le AWS Management Console et authentifiez-vous à l'aide de vos informations d'identification. AWS
2. Choisissez Groupes d'utilisateurs et choisissez le groupe d'utilisateurs que vous avez créé. Vous pouvez trouver votre identifiant ***pool-id*** dans le groupe d'utilisateurs.
3. Choisissez Intégration d'applications où vous pouvez trouver votre*your-cognito-domain*, puis accédez à Clients d'applications et analyses et choisissez votre application.
4. Dans App client : YourApp, vous pouvez trouver le *client-nameclient-id*, et *client-secret* (Afficher le secret du client).
5. *region-id*correspond à l'ID de AWS région dans lequel vous avez créé votre utilisateur et votre groupe d'utilisateurs Amazon Cognito. Exemple: `eu-west-3`.
6. Pour *redirect-uri* saisir l'URI que vous avez spécifié pour l'URL de rappel autorisée. Dans notre exemple, c'est le cash`http://localhost:8080/planetsdemo/login/oauth2/code/cognito`.

Vous pouvez désormais déployer votre application Gapwalk et utiliser l'utilisateur créé précédemment pour vous connecter à votre application.

Configurer l' OAuth2 authentification Gapwalk avec Keycloak

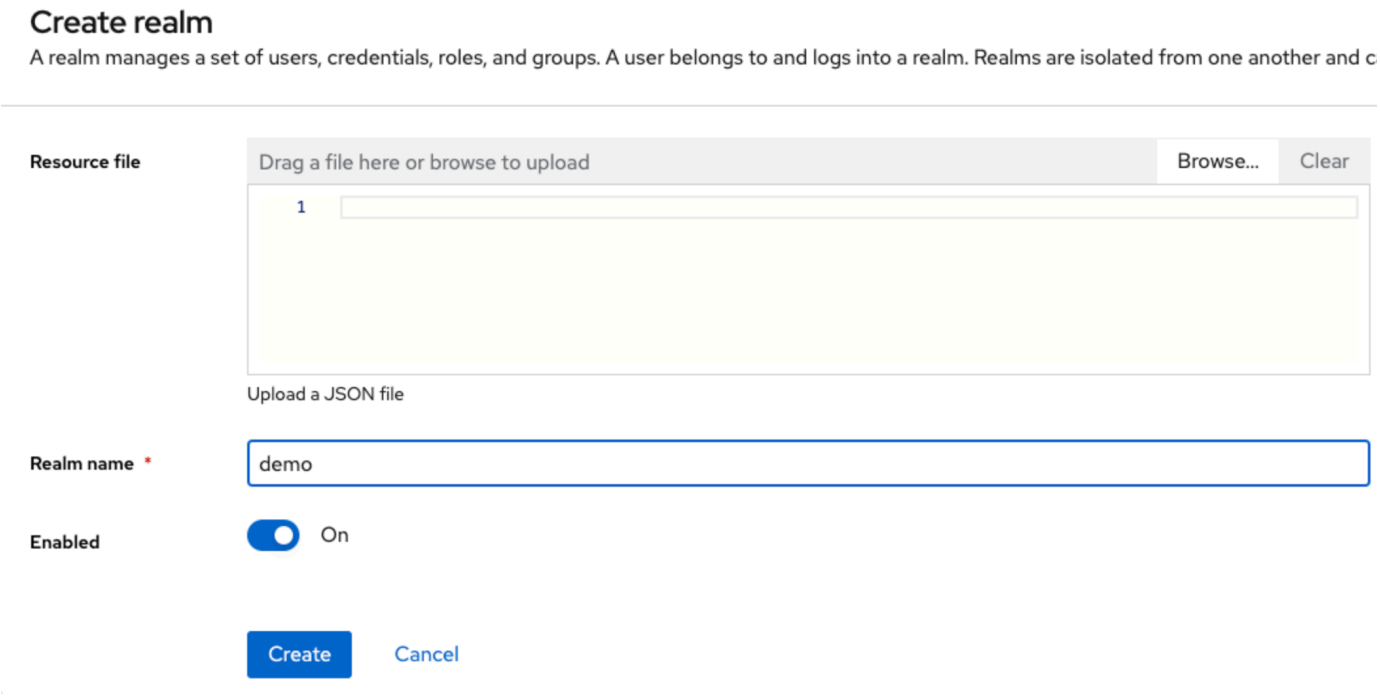
Cette rubrique décrit comment configurer l' OAuth2 authentification pour les applications Gapwalk utilisant Keycloak comme fournisseur d'identité (IdP). Dans ce tutoriel, nous utilisons Keycloak 24.0.0.

Prérequis

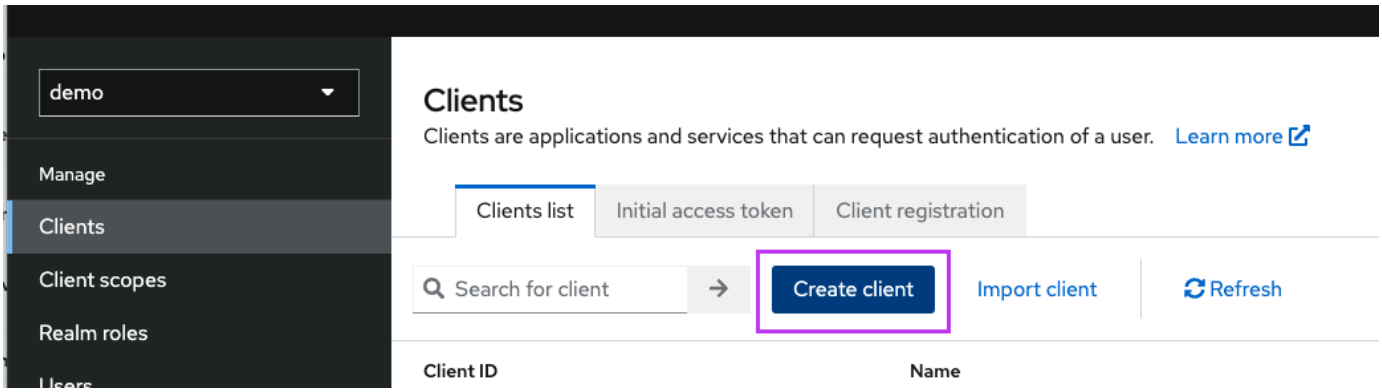
- Cape à clés
- Application Gapwalk

Configuration de Keycloak

1. Accédez à votre tableau de bord Keycloak dans votre navigateur Web. Les informations d'identification par défaut sont admin/admin. Accédez à la barre de navigation en haut à gauche et créez un domaine portant le nom**`demo`**, comme indiqué dans l'image suivante.



2. Créez un client portant ce nom**`app-demo`**.



Remplacez `localhost:8080` par l'adresse de votre application Gapwalk

General settings

Client ID * app-demo

Name

Description

Always display in UI Off

Access settings

Root URL http://localhost:8080

Home URL

Valid redirect URIs
- http://localhost:8080/*
- https://localhost:8080/*

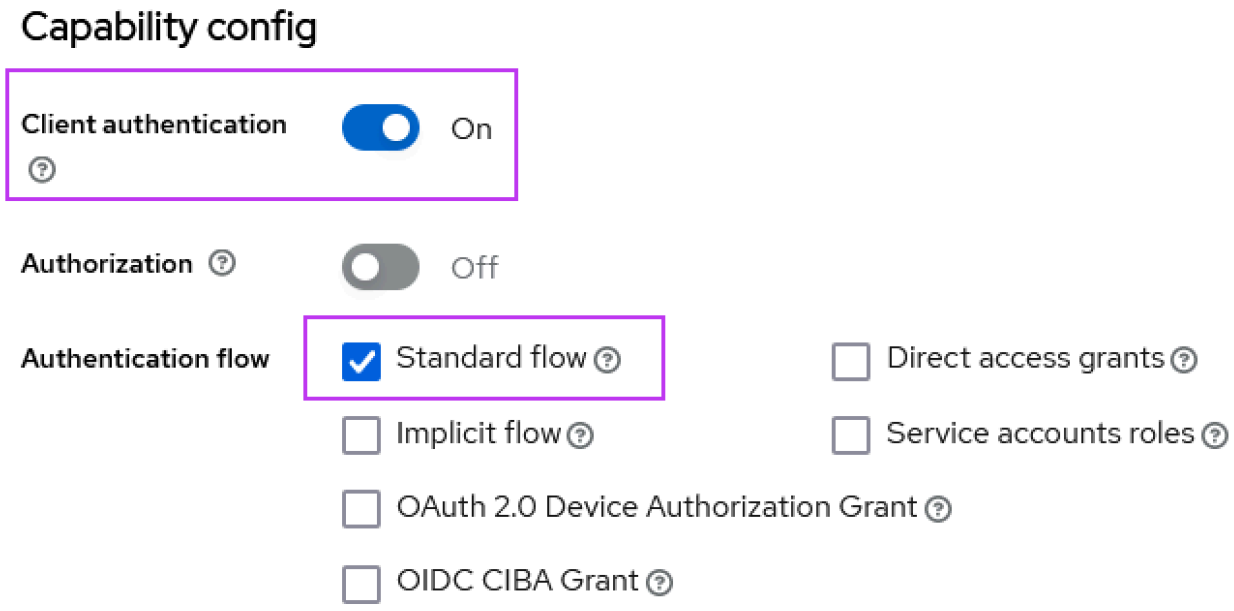
Add valid redirect URIs

Valid post logout redirect URIs
- http://localhost:8080/*
- https://localhost:8080/*

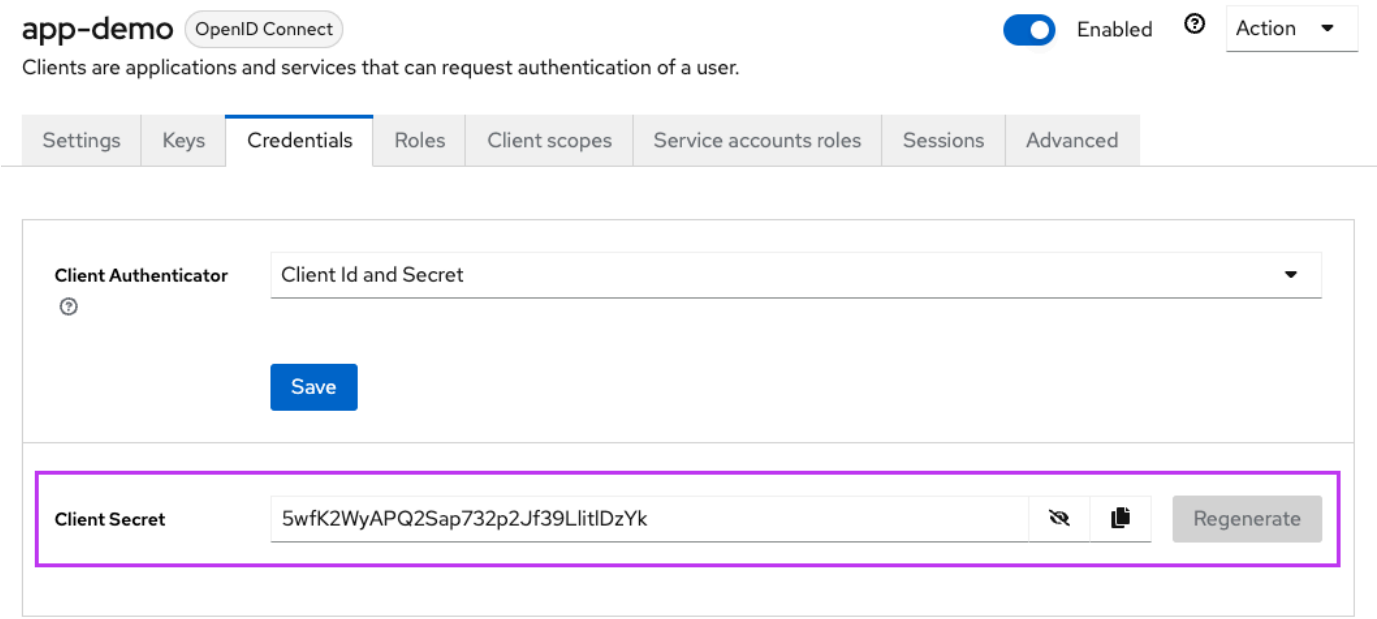
Add valid post logout redirect URIs

Web origins
- +

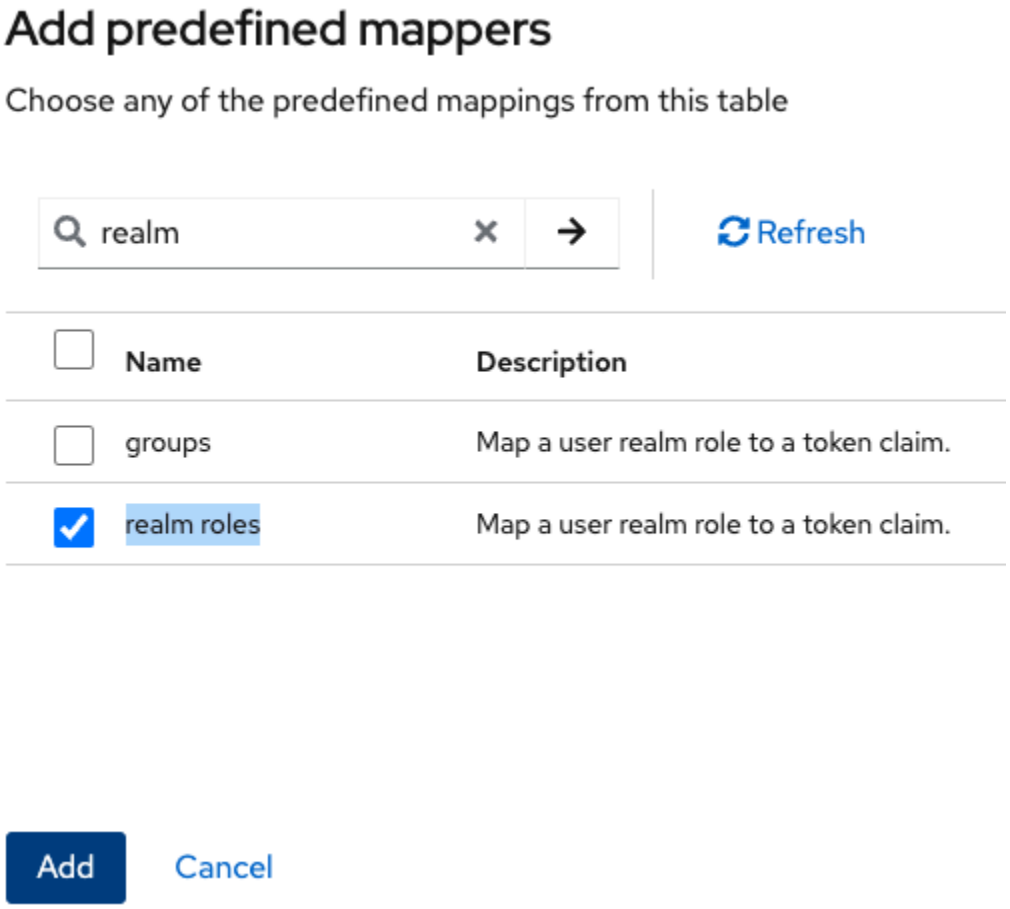
Add web origins



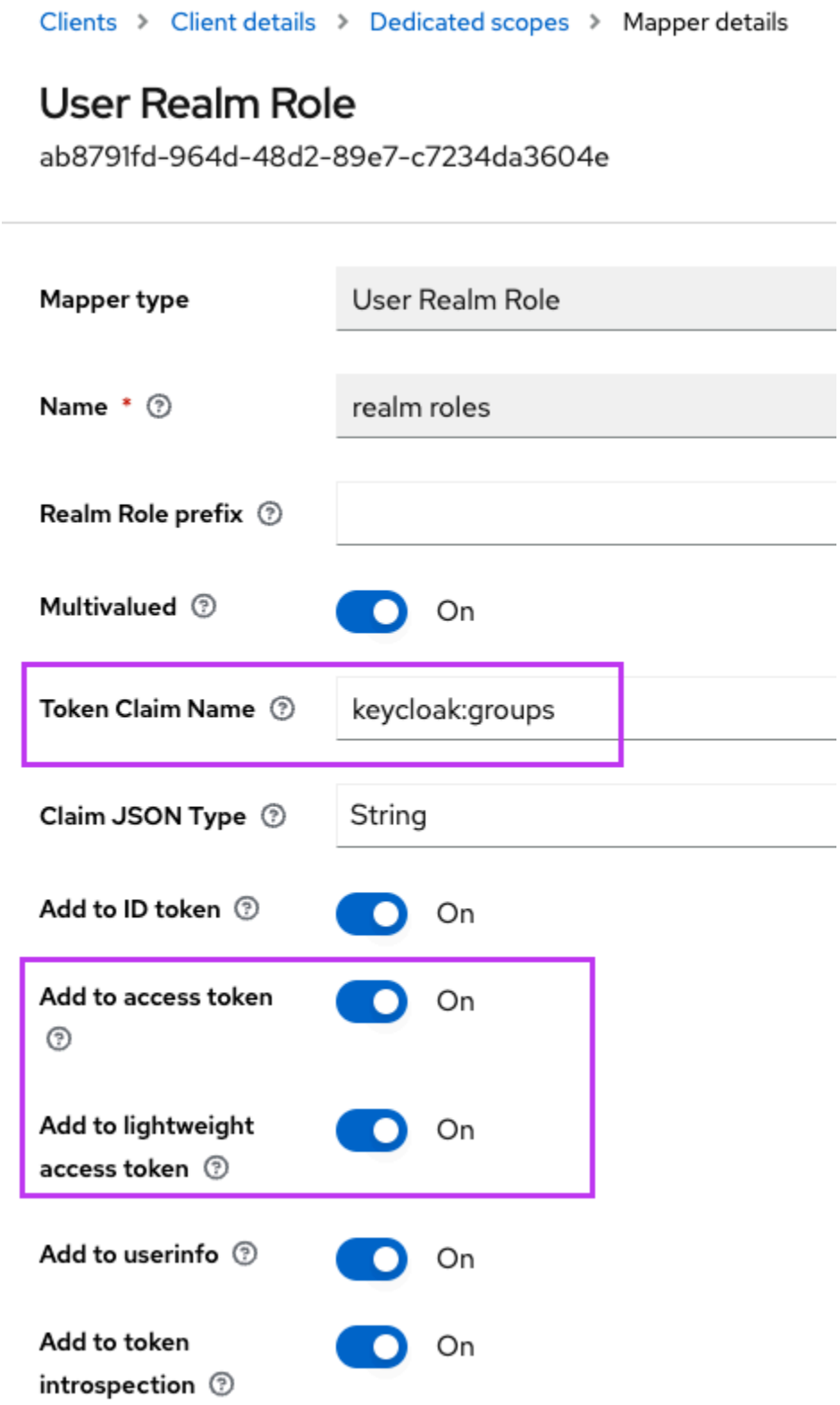
3. Pour obtenir le secret de votre client, choisissez Clients, app-demo, puis Credentials.



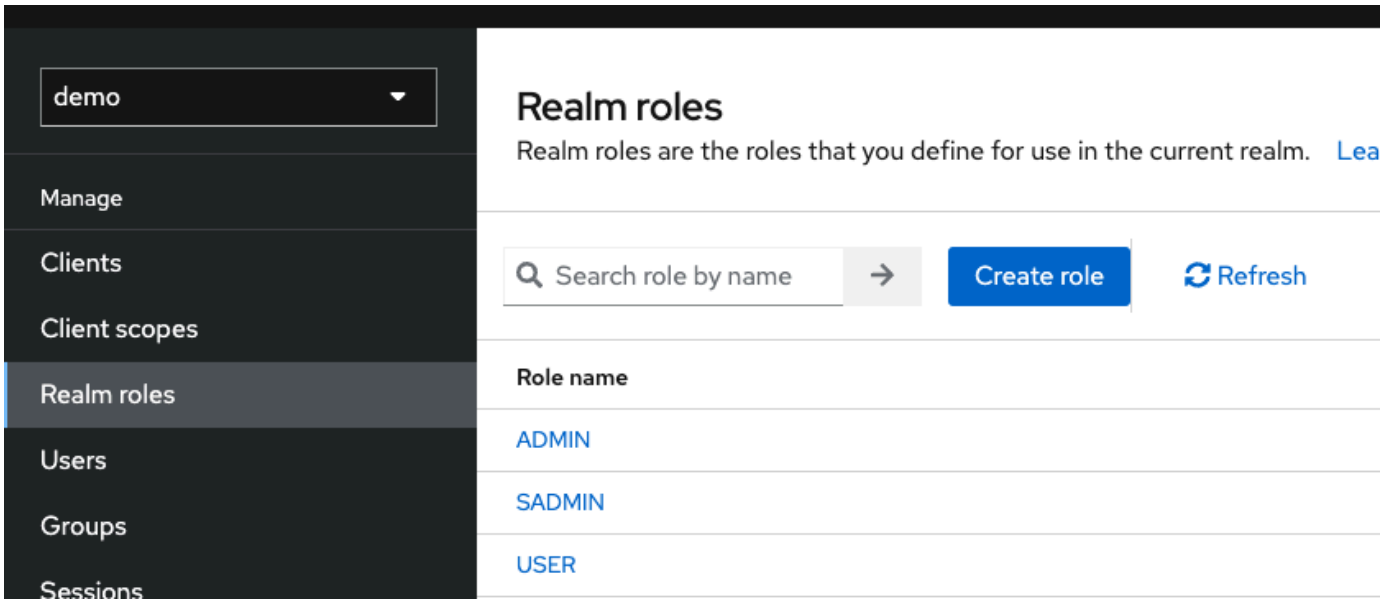
4. Choisissez Clients, puis Étendue du client, puis Ajouter un mappeur prédéfini. Choisissez les rôles du domaine.



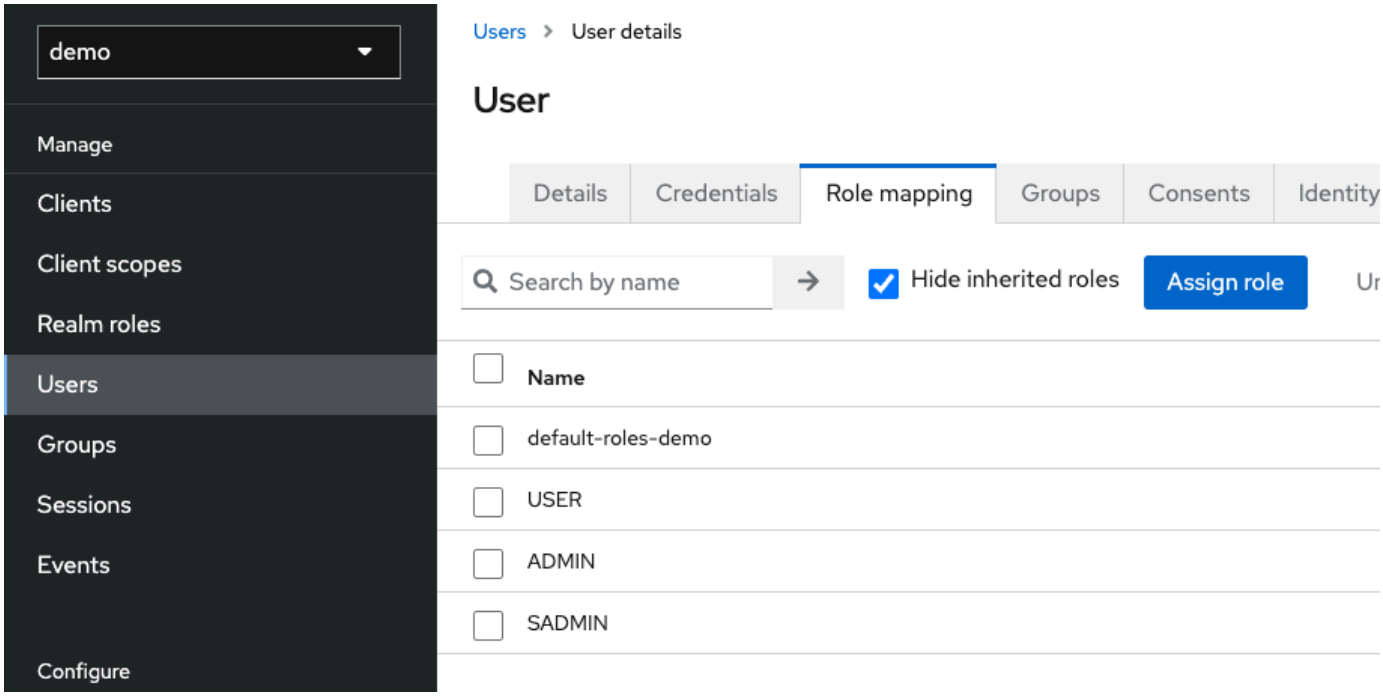
5. Modifiez votre rôle de domaine avec la configuration illustrée dans l'image suivante.


Clients > Client details > Dedicated scopes > Mapper details
User Realm Role
ab8791fd-964d-48d2-89e7-c7234da3604e
Mapper type
User Realm Role
Name *
realm roles
Realm Role prefix
Multivalued
On
Token Claim Name
keycloak:groups
Claim JSON Type
String
Add to ID token
On
Add to access token
On
Add to lightweight access token
On
Add to userinfo
On
Add to token introspection
On

6. N'oubliez pas le nom de demande de jeton défini. Vous aurez besoin de cette valeur dans la définition des paramètres Gapwalk de la `gapwalk-application.security.claimGroupName` propriété.



7. Choisissez les rôles Realms, puis créez 3 rôles : **`SUPER_ADMINADMIN`**, et**`USER`**. Ces rôles sont ensuite mappés à `ROLE_SUPER_ADMINROLE_ADMIN`, et `ROLE_USER` par l'application Gapwalk pour pouvoir accéder à certains appels REST d'API restreints.



Intégrez Keycloak dans l'application Gapwalk

Modifiez votre `application-main.yml` comme suit :

```
gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
gapwalk-application.security.issuerUri: http://<KEYCLOAK_SERVER_HOSTNAME>/realms/
<YOUR_REALM_NAME>
gapwalk-application.security.claimGroupName: "keycloak:groups"

gapwalk-application.security.userAttributeName: "preferred_username"
# Use "username" for cognito,
#     "preferred_username" for keycloak
#      or any other string
gapwalk-application.security.localhostWhitelistingEnabled: false

spring:
  security:
    oauth2:
      client:
        registration:
          demo:
            client-id: <YOUR_CLIENT_ID>
            client-name: Demo App
            client-secret: <YOUR_CLIENT_SECRET>
            provider: keycloak
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "{baseUrl}/login/oauth2/code/{registrationId}"
        provider:
          keycloak:
            issuer-uri: ${gapwalk-application.security.issuerUri}
            authorization-uri: ${gapwalk-application.security.issuerUri}/protocol/
openid-connect/auth
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/certs
            token-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/token
            user-name-attribute: ${gapwalk-application.security.userAttributeName}
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/certs
```

Remplacez*<KEYCLOAK_SERVER_HOSTNAME>*, *<YOUR_REALM_NAME><YOUR_CLIENT_ID>*, et par le nom *<YOUR_CLIENT_SECRET>* d'hôte de votre serveur Keycloak, votre nom de domaine, votre identifiant client et votre secret client.

AWS Blue Age Runtime APIs

Le AWS Blu Age Runtime utilise plusieurs applications Web pour exposer les points de terminaison REST, fournissant ainsi des moyens d'interagir avec les applications modernisées à l'aide de clients REST (par exemple en appelant des tâches à l'aide d'un planificateur).

Le but de ce document est de répertorier les points de terminaison REST disponibles, en fournissant des détails sur :

- Leur rôle
- La façon de les utiliser correctement

La liste des points de terminaison est organisée en catégories, en fonction de la nature du service fourni et de l'application Web exposant les points de terminaison.

Nous partons du principe que vous avez déjà une connaissance de base de l'utilisation des points de terminaison REST (à l'aide d'outils dédiés tels que POSTMAN, Thunder Client, CURL, navigateurs Web, etc.) ou en écrivant votre propre code pour effectuer un appel d'API.

Rubriques

- Points de terminaison disponibles pour l'utilisateur lors de la création URLs
- Points de terminaison pour l'application Gapwalk dans Blu Age AWS
- Points de terminaison REST de la console d'applications Blusam
- Gérer la console d'applications JICS dans AWS Blu Age
- Structures de données pour les utilisateurs de AWS Blu Age

Points de terminaison disponibles pour l'utilisateur lors de la création URLs

Cette rubrique répertorie les chemins URLs d'accès racine pour les points de terminaison. Chaque application Web ci-dessous définit un chemin racine, partagé par tous les points de terminaison. Chaque point de terminaison ajoute ensuite son propre chemin dédié. L'URL résultante à utiliser

est le résultat de la concaténation des chemins. Par exemple, en considérant le premier point de terminaison de l'application Gapwalk, nous avons :

- `/gapwalk-application`pour le chemin de l'application Web racine.
- `/scripts`pour le chemin du point de terminaison dédié.

L'URL résultante à utiliser sera `http://`*`server`*`:`*`port`*`/gapwalk-application/scripts`

serveur

pointe vers le nom du serveur (celui hébergeant l'application Web donnée).

port

le port exposé par le serveur.

Points de terminaison pour l'application Gapwalk dans Blu Age AWS

Dans cette rubrique, découvrez les points de terminaison de l'application Web Gapwalk. Ils utilisent le chemin racine`/gapwalk-application`.

Rubriques

- Points de terminaison liés aux tâches par lots (modernisés JCLs et similaires)
- Points de terminaison des métriques
- Autres points de terminaison
- Points de terminaison liés aux files d'attente de tâches

Points de terminaison liés aux tâches par lots (modernisés JCLs et similaires)

Les tâches Batch peuvent être exécutées de manière synchrone ou asynchrone (voir les détails ci-dessous). Les tâches par lots sont exécutées à l'aide de scripts groovy issus de la modernisation des anciens scripts (JCL).

Rubriques

- Répertorier les scripts déployés
- Lancer un script de manière synchrone

- Lancer un script de manière asynchrone
- Liste des scripts déclenchés
- Récupération des détails d'exécution des tâches
- Liste des scripts lancés de manière asynchrone qui peuvent être supprimés
- Liste des scripts lancés de manière synchrone qui peuvent être supprimés
- Annulation de l'exécution d'une tâche donnée
- Répertorier les points de contrôle existants pour la redémarrabilité
- Redémarrer une tâche (de manière synchrone)
- Redémarrer une tâche (de manière asynchrone)
- Définition de la limite de threads pour les exécutions de tâches asynchrones

Répertorier les scripts déployés

- Méthode prise en charge : GET
- Trajectoire : `/scripts`
- Arguments : aucun
- Ce point de terminaison renvoie la liste des scripts groovy déployés sur le serveur, sous forme de chaîne. Ce point de terminaison est principalement destiné à être utilisé à partir d'un navigateur Web, car la chaîne qui en résulte est une page HTML, avec des liens actifs (un lien par script pouvant être lancé, voir l'exemple ci-dessous).

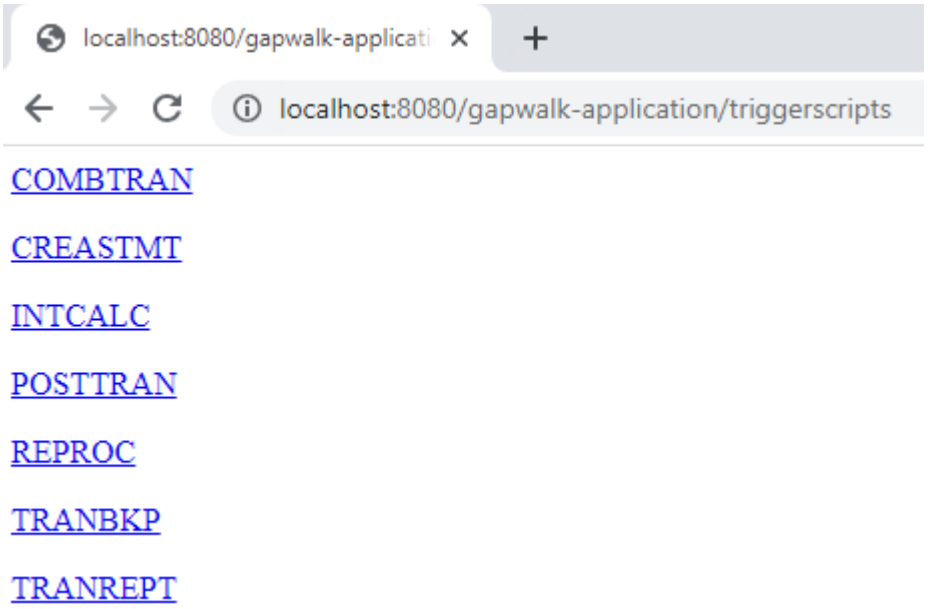
Exemple de réponse :

```
<p><a href=./script/COMBTRAN>COMBTRAN</a></p><p><a href=./script/CREASTMT>CREASTMT</
a></p><p><a href=./script/INTCALC>INTCALC</a></p><p><a href=./script/
POSTTRAN>POSTTRAN</a></p><p><a href=./script/REPROC>REPROC</a></p><p><a href=./script/
TRANBKP>TRANBKP</a></p><p><a href=./script/TRANREPT>TRANREPT</a></p><p><a href=./
script/functions>functions</a></p>
```

> **Note**
>
> Les liens représentent l'URL à utiliser pour lancer chaque script répertorié de manière synchrone.

- Méthode prise en charge : GET
- Trajectoire : `/triggerscripts`
- Arguments : aucun
- Ce point de terminaison renvoie la liste des scripts groovy déployés sur le serveur, sous forme de chaîne. Ce point de terminaison est principalement destiné à être utilisé à partir d'un navigateur Web, car la chaîne qui en résulte est une page HTML, avec des liens actifs (un lien par script pouvant être lancé, voir l'exemple ci-dessous).

  Contrairement à la réponse précédente du point de terminaison, les liens représentent l'URL à utiliser pour lancer chaque script répertorié de manière asynchrone.



Lancer un script de manière synchrone

Ce point de terminaison possède deux variantes avec des chemins dédiés pour l'utilisation de GET et POST (voir ci-dessous).

- Méthode prise en charge : GET
- Trajectoire : `/script/{scriptId:.+}`
- Méthode prise en charge : POST
- Trajectoire : `/post/script/{scriptId:.+}`
- Arguments :
  - identifiant du script à lancer

  - optionnellement : paramètres à transmettre au script, en utilisant les paramètres de requête (considérés comme un`Map<String,String>`). Les paramètres donnés seront automatiquement ajoutés aux liaisons du script groovy invoqué.
- L'appel lancera le script avec l'identifiant donné, en utilisant des paramètres supplémentaires s'ils sont fournis, et attendra la fin de l'exécution du script avant de renvoyer un message (`String`) qui sera soit :
  - « C'est fait. » (si l'exécution du travail s'est bien déroulée).
  - Un message d'erreur JSON contenant des détails sur ce qui s'est mal passé lors de l'exécution de la tâche. De plus amples informations peuvent être extraites des journaux du serveur afin de comprendre ce qui s'est mal passé lors de l'exécution de la tâche.

```
{
    "exitCode": -1,
    "stepName": "STEP15",
    "program": "CBACT04C",
    "status": "Error"
}
```

En consultant les journaux du serveur, nous pouvons découvrir qu'il s'agit d'un problème de déploiement (le programme attendu n'a pas été correctement déployé, il est donc introuvable, ce qui entraîne l'échec de l'exécution de la tâche) :

```
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - --> executing script INTCALC
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - Bound jobContext 419695287 - GDGEventsQueueHandler :907380469
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Added jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] to Sync Script Control Tower.
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor               - a65c2791-864f-43c9-972a-b5f2353389e6 - worker :Thread-26 [1547512424]
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor               - Triggered script: INTCALC - [a65c2791-864f-43c9-972a-b5f2353389e6] - jobContext [419695287]
2023-06-09_10-27-29-613 | [JOB] INTCALC - Started
2023-06-09_10-27-29-651 | [STEP] STEP15 - Started
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl    - Could not find program "CBACT04C" in the program registry.
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl    - Could not find program "CBACT04C" in the program registry.
2023-06-09_10-27-29-760 | Program not found => not executed !
2023-06-09_10-27-29-761 | [STEP] STEP15 - Ended
2023-06-09_10-27-29-772 | [JOB] INTCALC - Ended
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext          - Job [419695287] - starting final operation
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext          - End of job [419695287]
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Removed jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] from Script Control Tower.
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Remaining jobExecutors:0
```

> **Note**
>
> Les appels synchrones doivent être réservés aux tâches de courte durée. Les tâches exécutées pendant de longues périodes devraient plutôt être lancées de manière asynchrone (voir point de terminaison dédié ci-dessous).

Lancer un script de manière asynchrone

- Méthodes prises en charge : GET/POST
- Trajectoire : `/triggerscript/{scriptId:.+}`
- Arguments :
  - identifiant du script à lancer
  - optionnellement : paramètres à transmettre au script, en utilisant les paramètres de requête (considérés comme un`Map<String,String>`). Les paramètres donnés seront automatiquement ajoutés aux https://docs.groovy-lang.org/latest/html/api/groovy/lang/Binding.html [bindings] du script groovy invoqué.
- Contrairement au mode synchrone ci-dessus, le point de terminaison n'attend pas la fin de l'exécution de la tâche pour envoyer une réponse. L'exécution de la tâche est lancée immédiatement, si un thread disponible est trouvé pour le faire, et une réponse est immédiatement envoyée à l'appelant, avec l'identifiant d'exécution de la tâche, un identifiant unique représentant l'exécution de la tâche, qui peut être utilisé pour demander l'état d'exécution de la tâche ou forcer l'arrêt d'une exécution de tâche censée mal fonctionner. Le format de la réponse est le suivant :

  ```
  Triggered script <script identifier> [unique job execution id] @ <date and time>
  ```

- Étant donné que l'exécution asynchrone de la tâche repose sur un nombre limité de threads, l'exécution de la tâche risque de ne pas être lancée si aucun thread disponible n'est trouvé. Dans ce cas, le message renvoyé ressemblera plutôt à :

  ```
  Script [<script identifier>] NOT triggered - Thread limit reached (<actual thread
   limit>) - Please retry later or increase thread limit.
  ```

  Consultez le `settriggerthreadlimit` point de terminaison ci-dessous pour savoir comment augmenter la limite de threads.

Exemple de réponse :

```
Triggered script INTCALC [d43cbf46-4255-4ce2-aac2-79137573a8b4] @ 06-12-2023 16:26:15
```

L'identifiant unique d'exécution des tâches permet de récupérer rapidement les entrées de journal associées dans les journaux du serveur si nécessaire. Il est également utilisé par plusieurs autres points de terminaison détaillés ci-dessous.

Liste des scripts déclenchés

- Méthodes prises en charge : GET
- Chemins :`/triggeredscripts/{status:.+}`, `/triggeredscripts/{status:.+}/{namefilter}`
- Arguments :
  - État (obligatoire) : le statut des scripts déclenchés à récupérer. Les valeurs possibles sont les suivantes :
    - `all`: affiche tous les détails relatifs à l'exécution des tâches, qu'elles soient toujours en cours d'exécution ou non.
    - `running`: affiche uniquement les détails des tâches en cours d'exécution.
    - `done`: affiche les détails des jobs uniquement pour les jobs dont l'exécution est terminée.
    - `killed`: affiche uniquement les détails des tâches dont l'exécution a été interrompue de force à l'aide du point de terminaison dédié (voir ci-dessous).
    - `triggered`: affiche uniquement les détails des tâches qui ont été déclenchées mais qui n'ont pas encore été lancées.
    - `failed`: affiche uniquement les détails des tâches dont l'exécution a été marquée comme ayant échoué.
    - _namefilter (facultatif) _ : récupère uniquement les exécutions pour l'identifiant de script donné.
- Renvoie une collection de détails sur les exécutions de tâches au format JSON. Pour de plus amples informations, veuillez consulter Structure du message détaillée sur l'exécution des tâches.

Exemple de réponse :

```
[
    {
      "scriptId": "INTCALC",
      "caller": "127.0.0.1",
      "identifier": "d43cbf46-4255-4ce2-aac2-79137573a8b4",
      "startTime": "06-12-2023 16:26:15",
      "endTime": "06-12-2023 16:26:15",
      "status": "DONE",
      "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
 \"CBACT04C\", \"status\": \"Error\" }",
      "executionMode": "ASYNCHRONOUS"
    }
```

```
]
```

Récupération des détails d'exécution des tâches

- Méthode prise en charge : GET
- Trajectoire : `/getjobexecutioninfo/{jobexecutionid:.+}`
- Arguments :
  - jobexecutionid (obligatoire) : identifiant unique d'exécution de la tâche pour récupérer les détails d'exécution de la tâche correspondants.
- Renvoie une chaîne JSON représentant les détails d'exécution d'une seule tâche (voirStructure du message détaillée sur l'exécution des tâches) ou une réponse vide si aucun détail d'exécution de tâche n'a pu être trouvé pour l'identifiant donné.

Liste des scripts lancés de manière asynchrone qui peuvent être supprimés

- Méthode prise en charge : GET
- Trajectoire : `/killablescripts`
- Renvoie une collection d'identifiants d'exécution de tâches lancées de manière asynchrone qui sont toujours en cours d'exécution et peuvent être supprimées de force (voir le `/kill` point de terminaison ci-dessous).

Liste des scripts lancés de manière synchrone qui peuvent être supprimés

- Méthode prise en charge : GET
- Trajectoire : `/killablesyncscripts`
- Renvoie une collection d'identifiants d'exécution de tâches qui ont été lancées de manière synchrone, sont toujours en cours d'exécution et peuvent être supprimées de force (voir le `/kill` point de terminaison ci-dessous).

Annulation de l'exécution d'une tâche donnée

- Méthode prise en charge : GET
- Trajectoire : `/kill/{identifier:.+}`
- Argument : identifiant d'exécution de la tâche (obligatoire) : identifiant unique d'exécution de la tâche qui doit être supprimée de force.

- Renvoie un message textuel détaillant le résultat de la tentative d'arrêt de l'exécution de la tâche ; le message contiendra l'identifiant du script, l'identifiant unique de l'exécution de la tâche ainsi que la date et l'heure auxquelles l'arrêt de l'exécution s'est produit. Si aucune exécution de tâche en cours n'a pu être trouvée pour l'identifiant donné, un message d'erreur sera renvoyé à la place.

⚠ Warning

- Le moteur d'exécution fait de son mieux pour arrêter correctement l'exécution de la tâche cible. Ainsi, la réponse du point de terminaison /kill peut mettre un certain temps à atteindre l'appelant, car le moteur d'exécution de AWS Blu Age essaiera de minimiser l'impact commercial d'une suppression de la tâche.
- L'arrêt forcé d'une exécution de travail ne doit pas être fait à la légère, car cela peut avoir des conséquences commerciales directes, notamment une perte ou une corruption de données. Il doit être réservé aux cas où l'exécution d'une tâche donnée a échoué et où les moyens de correction des données sont clairement identifiés.
- La suppression d'un emploi devrait donner lieu à de nouvelles enquêtes (analyse post mortem) afin de déterminer ce qui n'a pas fonctionné et de prendre les mesures correctives appropriées.
- Dans tous les cas, toute tentative d'arrêt d'une tâche en cours d'exécution sera consignée dans les journaux du serveur avec des messages de niveau d'avertissement.

Répertorier les points de contrôle existants pour la redémarrabilité

La capacité de redémarrage des tâches repose sur la capacité des scripts à enregistrer des points de contrôle dans le `CheckpointRegistry` afin de suivre la progression de l'exécution des tâches. Si l'exécution d'une tâche ne se termine pas correctement et que des points de contrôle de redémarrage ont été enregistrés, il suffit de redémarrer l'exécution de la tâche à partir du dernier point de contrôle enregistré connu (sans avoir à exécuter les étapes précédentes au-dessus du point de contrôle).

- Méthode prise en charge : GET
- Trajectoire : `/restarts/{scriptId}/{jobId}`
- Arguments :
  - ScriptId (facultatif - chaîne) : le script en cours de redémarrage.
  - JoBid (facultatif - chaîne) : identifiant unique de l'exécution d'une tâche.

- Renvoie une liste au format JSON des points de redémarrage existants, qui peut être utilisée pour redémarrer une tâche dont l'exécution ne s'est pas terminée correctement ou pour déclencher un redémarrage différé en contournant les étapes précédemment exécutées. Si aucun point de contrôle n'a été enregistré par aucun script, le contenu de la page sera « Aucun point de contrôle enregistré ».

Redémarrer une tâche (de manière synchrone)

- Méthode prise en charge : GET
- Trajectoire : `/restart/{hashcode}/{scriptId}/{skipflag}`
- Arguments :
  - hashcode (entier - obligatoire) : redémarre l'exécution la plus récente d'une tâche en utilisant le hashcode fourni comme valeur de point de contrôle (voir le point de `/restarts` terminaison ci-dessus pour savoir comment récupérer une valeur de point de contrôle valide).
  - ScriptId (facultatif - chaîne) : le script en cours de redémarrage.
  - skipflag (facultatif - booléen) : ignore l'exécution de l'étape sélectionnée (point de contrôle) et relance l'étape suivante (le cas échéant).
- Retours : voir la description des `/script` retours ci-dessus.

Redémarrer une tâche (de manière asynchrone)

- Méthode prise en charge : GET
- Trajectoire : `/triggerrestart/{hashcode}/{scriptId}/{skipflag}`
- Arguments :
  - hashcode (entier - obligatoire) : redémarre l'exécution la plus récente d'une tâche en utilisant le hashcode fourni comme valeur de point de contrôle (voir le point de `/restarts` terminaison ci-dessus pour savoir comment récupérer une valeur de point de contrôle valide).
  - ScriptId (facultatif - chaîne) : le script en cours de redémarrage.
  - skipflag (facultatif - booléen) : ignore l'exécution de l'étape sélectionnée (point de contrôle) et relance l'étape suivante (le cas échéant).
- Retours : voir la description des `/triggerscript` retours ci-dessus.

Définition de la limite de threads pour les exécutions de tâches asynchrones

L'exécution asynchrone de la tâche repose sur un pool de threads dédié dans la JVM. Ce pool a une limite fixe en ce qui concerne le nombre de threads disponibles. L'utilisateur a la possibilité d'ajuster la limite en fonction des capacités de l'hôte (nombre de CPUs, mémoire disponible, etc...). Par défaut, la limite de threads est fixée à 5 threads.

- Méthode prise en charge : GET
- Trajectoire : `/settriggerthreadlimit/{threadlimit:.+}`
- Argument (entier) : nouvelle limite de thread à appliquer. Doit être un entier strictement positif.
- Renvoie un message (`String`) indiquant la nouvelle limite de thread et la précédente, ou un message d'erreur si la valeur de limite de thread fournie n'est pas valide (il ne s'agit pas d'un entier strictement positif).

Exemple de réponse :

```
Set thread limit for Script Tower Control to 10 (previous value was 5)
```

Le comptage des tâches en cours a déclenché des exécutions de tâches

- Méthode prise en charge : GET
- Trajectoire : `/countrunningtriggeredscripts`
- Renvoie un message indiquant le nombre de tâches en cours lancées de manière asynchrone et la limite de threads (c'est-à-dire le nombre maximum de tâches déclenchées pouvant être exécutées simultanément).

Exemple de réponse :

```
0 triggered script(s) running (limit =10)
```

> **Note**
>
> Cela peut être utilisé pour vérifier, avant de lancer une tâche, si la limite de threads n'a pas été atteinte (ce qui empêcherait le lancement de la tâche).

Purger les informations relatives aux exécutions de tâches

Les informations relatives à l'exécution des tâches restent dans la mémoire du serveur tant que celui-ci est actif. Il peut être pratique de purger les informations les plus anciennes de la mémoire, car elles ne sont plus pertinentes ; c'est le but de ce point de terminaison.

- Méthode prise en charge : GET
- Trajectoire : `/purgejobinformation/{age:.+}`
- Arguments : une valeur entière strictement positive représentant l'âge en heures des informations à purger.
- Renvoie un message contenant les informations suivantes :
  - Nom du fichier de purge dans lequel les informations d'exécution des tâches purgées sont stockées à des fins d'archivage.
  - Nombre d'informations d'exécution de tâches purgées.
  - Nombre d'informations d'exécution de tâches restantes dans le mémo

Points de terminaison des métriques

JVM

Ce point de terminaison renvoie les métriques disponibles relatives à la JVM.

- Méthode prise en charge : GET
- Trajectoire : `/metrics/jvm`
- Arguments : aucun
- Renvoie un message contenant les informations suivantes :
  - threadActiveCount: nombre de threads actifs.
  - jvmMemoryUsed: mémoire utilisée activement par la machine virtuelle Java.
  - jvmMemoryMax: Mémoire maximale autorisée pour la machine virtuelle Java.
  - jvmMemoryFree: la mémoire disponible n'est pas actuellement utilisée par la machine virtuelle Java.

Session

Ce point de terminaison renvoie des métriques relatives aux sessions HTTP actuellement ouvertes.

- Méthode prise en charge : GET
- Trajectoire : `/metrics/session`
- Arguments : aucun
- Renvoie un message contenant les informations suivantes :
  - SessionCount : nombre de sessions utilisateur actives actuellement gérées par le serveur.

Par lots

- Méthode prise en charge : GET
- Trajectoire : `/metrics/batch`
- Arguments :
  - StartTimestamp (facultatif, numéro) : horodatage de début pour le filtrage des données.
  - EndTimestamp (facultatif, numéro) : horodatage de fin pour le filtrage des données.
  - page (facultatif, numéro) : numéro de page pour la pagination.
  - PageSize (facultatif, nombre) : nombre d'éléments par page en pagination.
- Renvoie un message contenant les informations suivantes :
  - contenu : liste des mesures d'exécution par lots.
  - PageNumber : numéro de page actuel dans la pagination.
  - PageSize : nombre d'éléments affichés par page.
  - TotalPages : nombre total de pages disponibles.
  - numberOfElements: nombre d'éléments sur la page en cours.
  - last : drapeau booléen pour la dernière page.
  - premier : drapeau booléen pour la première page.

Transaction

- Méthode prise en charge : GET
- Trajectoire : `/metrics/transaction`
- Arguments :
  - StartTimestamp (facultatif, numéro) : horodatage de début pour le filtrage des données.
  - EndTimestamp (facultatif, numéro) : horodatage de fin pour le filtrage des données.
  - page (facultatif, numéro) : numéro de page pour la pagination.

  - PageSize (facultatif, nombre) : nombre d'éléments par page en pagination.
- Renvoie un message contenant les informations suivantes :
  - contenu : liste des mesures d'exécution des transactions.
  - PageNumber : numéro de page actuel dans la pagination.
  - PageSize : nombre d'éléments affichés par page.
  - TotalPages : nombre total de pages disponibles.
  - numberOfElements: nombre d'éléments sur la page en cours.
  - last : drapeau booléen pour la dernière page.
  - premier : drapeau booléen pour la première page.

Autres points de terminaison

Utilisez ces points de terminaison pour répertorier les programmes ou services enregistrés, découvrir leur état de santé et gérer les transactions JICS.

Rubriques

- Liste des programmes enregistrés
- Liste des services enregistrés
- État de santé
- Liste des transactions JICS disponibles
- Lancer une transaction JICS
- Lancer une transaction JICS (alternative)
- Répertorier les sessions actives

Liste des programmes enregistrés

- Méthode prise en charge : GET
- Trajectoire : `/programs`
- Renvoie la liste des programmes enregistrés, sous forme de page html. Chaque programme est désigné par son identifiant principal. Les anciens programmes modernisés et les programmes utilitaires (IDCAMS, IEBGENER, etc...) sont renvoyés dans la liste. Notez que les programmes utilitaires disponibles dépendent des applications Web utilitaires déployées sur votre serveur

Tomcat. Par exemple, les programmes de support de l'utilitaire z/OS peuvent ne pas être disponibles pour les actifs iSeries modernisés, car ils ne sont pas pertinents.

Liste des services enregistrés

- Méthode prise en charge : GET
- Trajectoire : `/services`
- Renvoie la liste des services d'exécution enregistrés, sous forme de page html. Les services fournis sont fournis par le moteur d'exécution AWS Blu Age sous forme d'utilitaires, qui peuvent être utilisés par exemple dans des scripts groovy. Les services de chargement Blusam (pour créer des ensembles de données Blusam à partir d'anciens ensembles de données) entrent dans cette catégorie.

Exemple de réponse :

```
<p>BluesamESDSFileLoader</p><p>BluesamKSDSFileLoader</p><p>BluesamRRDSFileLoader</p>
```

État de santé

- Méthode prise en charge : GET
- Trajectoire : `/`
- Renvoie un message simple, indiquant que l'application gapwalk est opérationnelle () `Jics application is running.`

Liste des transactions JICS disponibles

- Méthode prise en charge : GET
- Trajectoire : `/transactions`
- Renvoie une page HTML répertoriant toutes les transactions JICS disponibles. Cela n'a de sens que pour les environnements dotés d'éléments JICS (modernisation des éléments CICS existants).

Exemple de réponse :

```
<p>INQ1</p><p>MENU</p><p>MNT2</p><p>ORD1</p><p>PRNT</p>
```

Lancer une transaction JICS

- Méthodes prises en charge : GET, POST
- Trajectoire : `/jicstransrunner/{jtrans:.+}`
- Arguments :
  - Identifiant de transaction JICS (chaîne, obligatoire) : identifiant de la transaction JICS à lancer (8 caractères au maximum)
  - obligatoire : données d'entrée supplémentaires à transmettre à la transaction, sous forme de carte<String, Object>. Le contenu de cette carte sera utilisé pour alimenter la COMMAREA qui sera consommée par la transaction JICS. La carte peut être vide si aucune donnée n'est requise pour exécuter la transaction.
  - facultatif : entrées d'en-têtes HTTP, pour personnaliser l'environnement d'exécution pour la transaction donnée. Les clés d'en-tête suivantes sont prises en charge :
    - `jics-channel`: nom du CANAL JICS à utiliser par le programme qui sera lancé lors du lancement de cette transaction.
    - `jics-container`: nom du CONTENEUR JICS à utiliser pour le lancement de cette transaction JICS.
    - `jics-startcode`: le STARTCODE (chaîne de 2 caractères maximum) à utiliser au début de la transaction JICS. Voir STARTCODE pour les valeurs possibles (parcourez la page en bas de la page).
    - `jicxa-xid`: Le XID (structure XID de l'identifiant de transaction X/Open) d'une « transaction globale » (XA), initiée par l'appelant, à laquelle participera le lancement actuel de la transaction JICS.
- Renvoie une sérialisation `com.netfective.bluage.gapwalk.rt.shared.web.TransactionResultBean` JSON, représentant le résultat du lancement de la transaction JICS.

Pour plus d'informations sur les détails de la structure, consultezStructure des résultats du lancement de la transaction.

Lancer une transaction JICS (alternative)

- méthodes prises en charge : GET, POST
- chemin : `/jicstransaction/{jtrans:.+}`
- Arguments :

Identifiant de transaction JICS (chaîne, obligatoire)

identifiant de la transaction JICS à lancer (8 caractères au maximum)

obligatoire : données d'entrée supplémentaires à transmettre à la transaction, sous forme de carte<String, Object>

Le contenu de cette carte sera utilisé pour alimenter la COMMAREA qui sera consommée par la transaction JICS. La carte peut être vide si aucune donnée n'est requise pour exécuter la transaction.

facultatif : entrées d'en-têtes HTTP, pour personnaliser l'environnement d'exécution pour la transaction donnée.

Les clés d'en-tête suivantes sont prises en charge :

- `jics-channel`: nom du CANAL JICS à utiliser par le programme qui sera lancé lors du lancement de cette transaction.
- `jics-container`: nom du CONTENEUR JICS à utiliser pour le lancement de cette transaction JICS.
- `jics-startcode`: le STARTCODE (chaîne de 2 caractères maximum) à utiliser au début de la transaction JICS. Pour les valeurs possibles, voir STARTCODE (parcourez la page en bas de la page).
- `jicxa-xid`: Le XID (structure XID de l'identifiant de transaction X/Open) d'une « transaction globale » (XA), initiée par l'appelant, à laquelle participera le lancement actuel de la transaction JICS.

- Renvoie une sérialisation `com.netfective.bluage.gapwalk.rt.shared.web.RecordHolderBean` JSON, représentant le résultat du lancement de la transaction JICS. Les détails de la structure se trouvent dansStructure des résultats de l'enregistrement du lancement de la transaction.

Répertorier les sessions actives

- méthodes prises en charge : GET, POST
- chemin : `/activesessionlist`
- Arguments : aucun
- Renvoie une liste de sérialisations `com.netfective.bluage.gapwalk.application.web.sessiontracker.SessionTrackerObje`

au format JSON, représentant la liste des sessions utilisateur actives. Lorsque le suivi de session est désactivé, une liste vide est renvoyée.

Points de terminaison liés aux files d'attente de tâches

Les files d'attente d'offres d'emploi sont le support de AWS Blu Age pour le mécanisme de soumission de AS4 00 offres d'emploi. Les files d'attente de tâches sont utilisées dans AS4 00 pour exécuter des tâches sur des pools de threads spécifiques. Une file de tâches est définie par un nom et un nombre maximum de threads correspondant au nombre maximum de programmes pouvant être exécutés simultanément sur cette file d'attente. Si le nombre de tâches soumises dans la file d'attente est supérieur au nombre maximal de threads, les tâches attendront qu'un fil soit disponible.

Pour obtenir la liste exhaustive du statut d'une tâche dans une file d'attente, consultezÉtat possible d'une tâche dans une file d'attente.

Les opérations sur les files d'attente de tâches sont gérées via les points de terminaison dédiés suivants. Vous pouvez appeler ces opérations depuis l'URL de l'application Gapwalk avec l'URL racine suivante :. `http://server:port/gapwalk-application/jobqueue`

Rubriques

- Liste des files d'attente disponibles
- Démarrer ou redémarrer une file d'attente de tâches
- Soumettre une offre d'emploi pour lancement
- Liste de toutes les offres d'emploi soumises
- Libérez toutes les tâches « en attente »
- Libérer toutes les tâches « en attente » pour un nom de tâche donné
- Libérer une tâche donnée pour un numéro de tâche
- Soumettre une offre d'emploi selon un calendrier répétitif
- Répertorier tous les travaux récurrents soumis
- Annuler la planification d'une tâche répétitive

Liste des files d'attente disponibles

- Méthode prise en charge : GET
- Trajectoire : `list-queues`

- Renvoie la liste des files d'attente disponibles ainsi que leur statut, sous forme de liste JSON de valeurs-clés.

Exemple de réponse :

```
{"Default":"STAND_BY","queue1":"STARTED","queue2":"STARTED"}
```

Les statuts possibles d'une file d'attente de tâches sont les suivants :

STAND_BY

la file d'attente des tâches attend d'être démarrée.

DÉMARRÉE

la file d'attente des tâches est opérationnelle.

UNKNOWN

l'état de la file d'attente des tâches ne peut pas être déterminé.

Démarrer ou redémarrer une file d'attente de tâches

- Méthode prise en charge : POST
- Trajectoire : `/restart/{name}`
- Argument : nom de la file d'attente à démarrer/redémarrer, sous forme de chaîne - obligatoire.
- Le point de terminaison ne renvoie rien mais s'appuie plutôt sur le statut HTTP pour indiquer le résultat de l'opération de démarrage/redémarrage :

  HTTP 200

  l'opération de démarrage/redémarrage s'est bien déroulée : la file de tâches donnée est maintenant DÉMARRÉE.

  HTTP 404

  la file d'attente des tâches n'existe pas.

  HTTP 503

  une exception s'est produite lors de la tentative de démarrage/redémarrage (les journaux du serveur doivent être inspectés pour déterminer ce qui s'est mal passé).

Soumettre une offre d'emploi pour lancement

- Méthode prise en charge : POST
- Trajectoire : `/submit`
- Argument : obligatoire en tant que corps de requête, une sérialisation JSON d'un `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objet. Pour de plus amples informations, veuillez consulter Soumettre une tâche et planifier la saisie d'une tâche.
- Renvoie un fichier JSON contenant l'original `SubmitJobMessage` et un journal indiquant si le travail a été soumis ou non.

Liste de toutes les offres d'emploi soumises

- Méthode prise en charge : GET
- Trajectoire : `/list-jobs?status={status}&size={size}&page={page}&sort={sort}`
- Arguments :
  - page : numéro de page à récupérer (par défaut = 1)
  - taille : taille de la page (par défaut = 50, max = 300)
  - sort : L'ordre des tâches. (par défaut = « ExecutionId »). « ExecutionID » est actuellement la seule valeur prise en charge
  - statut : (facultatif) S'il est présent, il filtrera en fonction du statut.
- Renvoie une liste de toutes les tâches planifiées, sous forme de chaîne JSON. Pour un exemple de réponse, voirRéponse à la liste des tâches planifiées.

Libérez toutes les tâches « en attente »

- Méthode prise en charge : POST
- Trajectoire : `/release-all`
- Renvoie un message indiquant le résultat de l'opération de tentative de publication. Voici deux cas possibles :
  - HTTP 200 et un message « Toutes les offres d'emploi ont été publiées avec succès ! » si tous les jobs ont été publiés avec succès.
  - HTTP 503 et un message « Jobs not released. Une erreur inconnue s'est produite. Consultez le journal pour plus de détails » en cas de problème lors de la tentative de publication.

Libérer toutes les tâches « en attente » pour un nom de tâche donné

Pour un nom de tâche donné, plusieurs tâches peuvent être soumises, avec des numéros de tâche différents (l'unicité d'une tâche exécutée est accordée par deux <job name, job number>). Le point de terminaison tentera de publier toutes les soumissions de tâches portant le nom de tâche donné, qui sont « en attente ».

- Méthode prise en charge : POST
- Trajectoire : `/release/{name}`
- Arguments : le nom de la tâche à rechercher, sous forme de chaîne. Obligatoire.
- Renvoie un message indiquant le résultat de l'opération de tentative de publication. Voici deux cas possibles :
  - HTTP 200 et un message « Jobs in group <name>(<number of released jobs>) a été publié avec succès ! » les offres d'emploi ont été publiées avec succès.
  - HTTP 503 et un message « Les tâches du groupe <name>n'ont pas été publiées. Une erreur inconnue s'est produite. Consultez le journal pour plus de détails » en cas de problème lors de la tentative de publication.

Libérer une tâche donnée pour un numéro de tâche

Le point de terminaison tentera de publier la soumission de tâche unique qui est « en attente », pour le couple donné <job name, job number>.

- Méthode prise en charge : POST
- Trajectoire : `/release/{name}/{number}`
- Arguments :

  name

  le nom de la tâche à rechercher, sous forme de chaîne. Obligatoire.

  nombre

  le numéro de tâche à rechercher, sous forme de nombre entier. Obligatoire.

  renvoie

  un message indiquant le résultat de l'opération de tentative de publication. Voici deux cas possibles :

- HTTP 200 et un message « Job <name/number> released with success ! » si le job a été publié avec succès.
- <name/number>HTTP 503 et un message « Job>Non publié ». Une erreur inconnue s'est produite. Consultez le journal pour plus de détails » en cas de problème lors de la tentative de publication.

Soumettre une offre d'emploi selon un calendrier répétitif

Planifiez une tâche qui sera exécutée selon un calendrier répétitif.

- Méthode prise en charge : POST
- Trajectoire : `/schedule`
- Argument : le corps de la requête doit contenir une sérialisation JSON d'un `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objet.

Répertorier tous les travaux récurrents soumis

- Méthode prise en charge : GET
- Trajectoire : `/schedule/list?status={status}&size={size}&page={page}&sort={sort}`
- Arguments :
  1. page : numéro de page à récupérer (par défaut = 1)
  2. taille : taille de la page (par défaut = 50, max = 300)
  3. sort : L'ordre des tâches. (par défaut = « id »). « id » est la seule valeur prise en charge pour le moment.
  4. statut : (facultatif) S'il est présent, il filtrera en fonction du statut. Les valeurs possibles sont celles mentionnées dans la section 1.
  5. statut : (facultatif) S'il est présent, il filtrera en fonction du statut. Les valeurs possibles sont celles mentionnées dans la section 1.
  6. Renvoie une liste de toutes les tâches planifiées, sous forme de chaîne JSON.

Annuler la planification d'une tâche répétitive

Supprime une tâche créée selon un calendrier répétitif. Le statut de planification des tâches est défini sur INACTIF.

- Méthode prise en charge : GET
- Trajectoire : `/schedule/remove/{schedule_id}`
- `schedule_id`Argument : identifiant de la tâche planifiée à supprimer.

Points de terminaison REST de la console d'applications Blusam

Dans cette section, vous découvrirez la console d'application Blusam, une API conçue pour simplifier la gestion des ensembles de données VSAM modernisés. Les points de terminaison de l'application Web Blusam utilisent le chemin racine. `/bac`

Rubriques

- Points de terminaison liés aux ensembles de données
- Points de terminaison associés aux ensembles de données en masse
- Enregistrements
- Masques
- Autre
- Terminaux de gestion des utilisateurs BAC

Points de terminaison liés aux ensembles de données

Utilisez les points de terminaison suivants pour créer ou gérer un ensemble de données spécifique.

Rubriques

- Création d'un ensemble de données
- Charger un fichier
- Charger un ensemble de données (POST)
- Charger un ensemble de données (GET)
- Charger un ensemble de données depuis un compartiment Amazon S3
- Exporter un ensemble de données vers un compartiment Amazon S3
- Effacer un ensemble de données
- Supprimer un ensemble de données
- Compter les enregistrements d'ensembles de données

Création d'un ensemble de données

Vous pouvez utiliser ce point de terminaison pour créer une définition d'ensemble de données.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/createDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.

  type

  (obligatoire, chaîne) : le type de jeu de données. Les valeurs possibles sont les suivantes :`ESDS`,`KSDS`,`RRDS`.

  Taille de l'enregistrement

  (facultatif, chaîne) : taille maximale de chaque enregistrement de l'ensemble de données.

  Longueur fixe

  (facultatif, booléen) : indique si la longueur des enregistrements est fixe.

  compression

  (facultatif, booléen) : indique si le jeu de données est compressé.

  Activer le cache

  (facultatif, booléen) : indique si la mise en cache est activée pour l'ensemble de données.

  Clés alternatives

  (facultatif, liste des clés) :

  - offset (obligatoire, nombre)
  - longueur (obligatoire, nombre)
  - nom (obligatoire, numéro)
- Renvoie un fichier JSON représentant le nouvel ensemble de données créé.

Demande d'échantillon :

```
POST /api/services/rest/bluesamservice/createDataSet
{
  "name": "DATASET",
  "checked": false,
  "records": [],
  "primaryKey": {
    "name": "PK"
  },
  "alternativeKeys": [
    {
      "offset": 10,
      "length": 10,
      "name": "ALTK_0"
    }
  ],
  "type": "ESDS",
  "recordSize": 10,
  "compression": true,
  "cacheEnable": true
}
```

Exemple de réponse :

```
{
    "dataSet": {
      "name": "DATASET",
      "checked": false,
      "nbRecords": 0,
      "keyLength": -1,
      "recordSize": 10,
      "compression": false,
      "fixLength": true,
      "type": "ESDS",
      "cacheEnable": false,
      "cacheWarmup": false,
      "cacheEviction": "100ms",
      "creationDate": 1686744961234,
      "modificationDate": 1686744961234,
      "records": [],
      "primaryKey": {
        "name": "PK",
        "offset": null,
        "length": null,
```

```
        "columns": null,
        "unique": true
      },
      "alternativeKeys": [
        {
          "offset": 10,
          "length": 10,
          "name": "ALTK_0"
        }
      ],
      "readLimit": 0,
      "readEncoding": null,
      "initCharacter": null,
      "defaultCharacter": null,
      "blankCharacter": null,
      "strictZoned": null,
      "decimalSeparator": null,
      "currencySign": null,
      "pictureCurrencySign": null
    },
    "message": null,
    "result": true
  }
```

Charger un fichier

Vous pouvez utiliser ce point de terminaison pour télécharger des fichiers sur le serveur. Le fichier est stocké dans un dossier temporaire qui correspond à chaque utilisateur spécifique. Utilisez ce point de terminaison chaque fois que vous devez télécharger un fichier.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/upload`
- Arguments :

  dans le fichier

  (obligatoire, multipart/form-data) : Le fichier à télécharger.
- Renvoie une valeur booléenne reflétant le statut du téléchargement

Charger un ensemble de données (POST)

Après `createDataSet` avoir créé la définition de l'ensemble de données, vous pouvez charger les enregistrements associés au fichier téléchargé dans un ensemble de données spécifique.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/loadDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Renvoie l'état de la demande et le jeu de données chargé.

Charger un ensemble de données (GET)

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/loadDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.

  fichier de jeu de données

  (obligatoire, chaîne) : nom du fichier du jeu de données.
- Renvoie l'état de la demande et le jeu de données chargé.

Charger un ensemble de données depuis un compartiment Amazon S3

Charge un ensemble de données à l'aide d'un fichier listcat depuis un compartiment Amazon S3.

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/loadDataSetFromS3`
- Arguments :

Emplacement des fichiers ListCat 3

(obligatoire, chaîne) : l'emplacement du fichier listcat sur Amazon S3.

Emplacement des fichiers du jeu de données 3

(obligatoire, chaîne) : l'emplacement du fichier d'ensemble de données sur Amazon S3.

region

(obligatoire, chaîne) : l'Amazon S3 Région AWS où les fichiers sont stockés.

- Renvoie le nouvel ensemble de données créé

Demande d'échantillon :

```
/BAC/api/services/rest/bluesamservice/loadDataSetFromS3?region=us-
east-1&listcatFileS3Location=s3://bucket-name/listcat.json&datasetFileS3Location=s3://
bucket-name/dataset.DAT
```

Exporter un ensemble de données vers un compartiment Amazon S3

Exporte un ensemble de données vers le compartiment Amazon S3 spécifié.

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/exportDataSetToS3`
- Arguments :

s3Location

(obligatoire, chaîne) : l'emplacement Amazon S3 vers lequel exporter l'ensemble de données.

datasetName

(obligatoire, chaîne) : le nom du jeu de données à exporter.

region

(obligatoire, chaîne) : Région AWS celui du compartiment Amazon S3.

kmsKeyId

(facultatif, chaîne) : l' AWS KMS ID à utiliser pour le chiffrement de l'ensemble de données exporté vers le compartiment Amazon S3.

- Renvoie l'ensemble de données exporté

Demande d'échantillon :

```
/BAC/api/services/rest/bluesamservice/exportDataSetToS3?region=eu-
west-1&s3Location=s3://bucket-name/dump&datasetName=dataset
```

Effacer un ensemble de données

Efface tous les enregistrements d'un ensemble de données.

- Méthodes prises en charge : POST, GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/clearDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom du jeu de données à effacer.
- Renvoie le statut de la demande.

Supprimer un ensemble de données

Supprime la définition et les enregistrements de l'ensemble de données.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/deleteDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : nom du jeu de données à supprimer.
- Renvoie le statut de la demande et le jeu de données supprimé.

Compter les enregistrements d'ensembles de données

Ce point de terminaison renvoie le nombre d'enregistrements associés à un ensemble de données.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/countRecords`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Retours : le nombre d'enregistrements

Points de terminaison associés aux ensembles de données en masse

Utilisez les points de terminaison suivants pour créer ou gérer plusieurs ensembles de données à la fois.

Rubriques

- Exporter des ensembles de données (GET)
- Exporter des ensembles de données (POST)
- Création de plusieurs ensembles de données
- Répertorier tous les ensembles de données
- Lister directement tous les ensembles de données
- Liste directe de tous les ensembles de données par page
- Ensemble de données de streaming
- Supprimer tous les ensembles de données
- Obtenir des définitions d'ensembles de données à partir du fichier listcat
- Obtenir les définitions des ensembles de données à partir du fichier list cat téléchargé
- Obtenir un ensemble de données
- Charger listcat depuis un fichier JSON

Exporter des ensembles de données (GET)

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/exportDataSet`

- Arguments :

  datasetName

  (obligatoire, chaîne) : le nom du jeu de données à exporter.

  datasetOutputFile

  (obligatoire, chaîne) : le chemin du dossier dans lequel vous souhaitez stocker le jeu de données exporté sur le serveur.

  rdw

  (obligatoire, booléen) : si vous souhaitez que le mot descripteur d'enregistrement (RDW) fasse partie des enregistrements exportés. Si l'ensemble de données comporte des enregistrements de longueur fixe, la valeur de ce paramètre est ignorée.
- Renvoie l'état de la demande et le chemin d'accès au fichier contenant l'ensemble de données exporté (le cas échéant). Si l'ensemble de données est nul dans la réponse, cela signifie que le système n'a pas pu localiser un ensemble de données portant le nom donné.

Exporter des ensembles de données (POST)

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/exportDataSet`
- Arguments :

  Paramètres de vidange

  (obligatoire, BACRead Paramètres) : paramètres de lecture Bluesam.
- Renvoie l'état de l'ensemble de données exporté.

Création de plusieurs ensembles de données

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/createAllDataSets`
- Arguments :
  - Liste des ensembles de données

name

(obligatoire, chaîne) : le nom de l'ensemble de données.

type

(obligatoire, chaîne) : le type de jeu de données. Les valeurs possibles sont les suivantes :`ESDS`,`KSDS`,`RRDS`.

Taille de l'enregistrement

(facultatif, chaîne) : taille maximale de chaque enregistrement de l'ensemble de données.

Longueur fixe

(facultatif, booléen) : indique si la longueur des enregistrements est fixe.

compression

(facultatif, booléen) : indique si le jeu de données est compressé.

Activer le cache

(facultatif, booléen) : indique si la mise en cache est activée pour l'ensemble de données.

- Retours : le statut de la demande et le nouvel ensemble de données créé.

Répertorier tous les ensembles de données

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/listDataSet`
- Arguments : Aucun
- Retours : le statut de la demande et la liste des ensembles de données.

Lister directement tous les ensembles de données

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/directListDataSet`
- Arguments : Aucun

- Retours : le statut de la demande et la liste des ensembles de données.

Liste directe de tous les ensembles de données par page

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/directListDataSetByPage`
- Arguments :

  datasetName

  (obligatoire, chaîne) : le nom de l'ensemble de données.

  Numéro de page

  (obligatoire, int) : le numéro de page.

  pageSize

  (obligatoire, int) : le format de page.
- Retours : le statut de la demande et la liste des ensembles de données.

Ensemble de données de streaming

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/streamDataset`
- Arguments :

  datasetName

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Retours : un flux des ensembles de données demandés.

Supprimer tous les ensembles de données

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/removeAll`

- Arguments : Aucun
- Renvoie : une valeur booléenne qui représente le statut de la demande.

Obtenir des définitions d'ensembles de données à partir du fichier listcat

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/getDataSetsDefinitionFromListcat`
- Arguments :

  paramFilePath

  (obligatoire, chaîne) : chemin d'accès au fichier listcat.
- Retours : liste d'ensembles de données

Obtenir les définitions des ensembles de données à partir du fichier list cat téléchargé

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/getDataSetsDefinitionFromUploadedListcat`
- Arguments : Aucun
- Retours : liste d'ensembles de données

Obtenir un ensemble de données

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/getDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Renvoie l'ensemble de données demandé.

Charger listcat depuis un fichier JSON

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/loadListcatFromJsonFile`
- Arguments :

  filePath

  (obligatoire, chaîne) : chemin d'accès au fichier listcat.
- Retours : liste d'ensembles de données

Enregistrements

Utilisez les points de terminaison suivants pour créer ou gérer des enregistrements au sein d'un ensemble de données.

Rubriques

- Créer un enregistrement
- Lire un ensemble de données
- Supprimer un enregistrement
- Mettre à jour un enregistrement
- Enregistrer un enregistrement
- Valider un enregistrement
- Obtenez un arbre d'enregistrements

Créer un enregistrement

Vous pouvez utiliser ce point de terminaison pour créer un nouvel enregistrement.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/createRecord`
- Arguments :

dataset

(obligatoire, DataSet) : l'objet du jeu de données

masque

(obligatoire, masque) : l'objet du masque.

- Renvoie le statut de la demande et l'enregistrement créé.

Lire un ensemble de données

Vous pouvez utiliser ce point de terminaison pour lire un ensemble de données.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/readDataSet`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données.
- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Supprimer un enregistrement

Vous pouvez utiliser ce point de terminaison pour supprimer un enregistrement d'un ensemble de données.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/deleteRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  record

  (obligatoire, Enregistrement) : l'enregistrement à supprimer

- Renvoie le statut de la suppression.

Mettre à jour un enregistrement

Vous pouvez utiliser ce point de terminaison pour mettre à jour un enregistrement associé à un ensemble de données.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/updateRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  record

  (obligatoire, enregistrement) : l'enregistrement à mettre à jour
- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Enregistrer un enregistrement

Vous pouvez utiliser ce point de terminaison pour enregistrer un enregistrement dans un ensemble de données à l'aide d'un masque.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/saveRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  record

  (obligatoire, enregistrement) : l'enregistrement à enregistrer
- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Valider un enregistrement

Utilisez ce point de terminaison pour valider un enregistrement.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/validateRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données
- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Obtenez un arbre d'enregistrements

Utilisez ce point de terminaison pour obtenir l'arborescence hiérarchique d'un enregistrement.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/getRecordTree`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  record

  (obligatoire, Record) : l'enregistrement à récupérer
- Renvoie le statut de la demande et l'arborescence hiérarchique de l'enregistrement demandé.

Masques

Utilisez les points de terminaison suivants pour charger ou appliquer des masques à un ensemble de données.

Rubriques

- Masques de chargement

- Appliquer un masque
- Appliquer un filtre de masque

Masques de chargement

Vous pouvez utiliser ce point de terminaison pour récupérer tous les masques associés à un ensemble de données spécifique.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/loadMasks`
- Variables de chemin :

  Taille de l'enregistrement :... /LoadMasks/ {recordSize}

  (facultatif, numérique) : la taille de l'enregistrement, filtres chargés, masques qui correspondent à cette taille d'enregistrement
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données
- Renvoie le statut de la demande et la liste des masques.

Appliquer un masque

Vous pouvez utiliser ce point de terminaison pour appliquer un masque à un ensemble de données spécifique.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/applyMask`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

masque

(obligatoire, masque) : l'objet du jeu de données

- Renvoie l'état de la demande et le jeu de données avec le masque appliqué.

Appliquer un filtre de masque

Vous pouvez utiliser ce point de terminaison pour appliquer un masque et un filtre à un ensemble de données spécifique.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/crud/applyMaskFilter`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  masque

  (obligatoire, masque) : l'objet du jeu de données
- Renvoie le statut de la demande et le jeu de données avec le masque et le filtre appliqués.

Autre

Utilisez les points de terminaison suivants pour gérer le cache d'un ensemble de données ou vérifier les caractéristiques d'un ensemble de données

Rubriques

- Vérifiez le cache de préchauffage
- Vérifier le cache activé
- Activer le cache
- Vérifiez le cache RAM alloué
- Vérifier la persistance
- Vérifiez les types d'ensembles de données pris en charge

- Vérifier l'état du serveur

Vérifiez le cache de préchauffage

Vérifie si le cache de préchauffage est activé pour un ensemble de données spécifique.

- Méthodes prises en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/warmupCache`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Renvoie : vrai si le cache de préchauffage est activé et faux dans le cas contraire.

Vérifier le cache activé

Vérifie si le cache est activé pour un ensemble de données spécifique.

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/isEnableCache`
- Arguments : Aucun
- Renvoie vrai si la mise en cache est activée.

Activer le cache

- Méthodes prises en charge : POST
- Nécessite une authentification et les rôles ROLE_ADMIN et ROLE_SUPER_ADMIN.
- Trajectoire : `/api/services/rest/bluesamservice/enableDisableCache/{enable}`
- Arguments :

  activer

  (obligatoire, booléen) : s'il est défini sur true, il activera la mise en cache.

- Ne renvoie aucun

Vérifiez le cache RAM alloué

Vous pouvez utiliser ce point de terminaison pour récupérer la mémoire cache RAM allouée.

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/allocatedRamCache`
- Arguments : Aucun
- Renvoie : la taille de la mémoire sous forme de chaîne

Vérifier la persistance

- Méthodes prises en charge : GET
- Nécessite une authentification et le rôle ROLE_USER.
- Trajectoire : `/api/services/rest/bluesamservice/persistence`
- Arguments : Aucun
- Retours : la persistance utilisée sous forme de chaîne

Vérifiez les types d'ensembles de données pris en charge

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/getDataSetTypes`
- Nécessite une authentification et le rôle ROLE_USER.
- Arguments : Aucun
- Renvoie : la liste des types d'ensembles de données pris en charge sous forme de liste de chaînes.

Vérifier l'état du serveur

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamserver/serverIsUp`
- Arguments : Aucun

- Retours : Aucun. Le code d'état de réponse HTTP 200 indique que le serveur est opérationnel.

Terminaux de gestion des utilisateurs BAC

Utilisez les points de terminaison suivants pour gérer les interactions des utilisateurs.

Rubriques

- Connectez un utilisateur
- Vérifiez s'il existe au moins un utilisateur dans le système
- Enregistrer un nouvel utilisateur
- Obtenir des informations sur l'utilisateur
- Répertorier des utilisateurs
- Suppression d'un utilisateur
- Déconnectez l'utilisateur actuel

Connectez un utilisateur

- Méthode prise en charge : POST
- Trajectoire : `/api/services/security/servicelogin/login`
- Arguments : Aucun
- Renvoie la sérialisation JSON d'un `com.netfective.bluage.bac.entities.SignOn` objet, représentant l'utilisateur dont les informations d'identification sont fournies dans la demande en cours. Le mot de passe est masqué dans la vue dans l'objet renvoyé. Les rôles attribués à l'utilisateur sont listés.

Exemple de réponse :

```
{
     "login": "some-admin",
     "password": null,
     "roles": [
       {
         "id": 0,
         "roleName": "ROLE_ADMIN"
       }
```

```
    ]
  }
```

Vérifiez s'il existe au moins un utilisateur dans le système

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogin/hasAccount`
- Arguments : Aucun
- Renvoie la valeur booléenne `true` si au moins un utilisateur autre que le super administrateur par défaut a été créé. Retourne dans `false` le cas contraire.

Enregistrer un nouvel utilisateur

- Méthode prise en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/security/servicelogin/recorduser`
- Arguments : la sérialisation JSON d'un `com.netfective.bluage.bac.entities.SignOn` objet qui représente l'utilisateur à ajouter au stockage. Les rôles de l'utilisateur doivent être définis, sinon l'utilisateur risque de ne pas être en mesure d'utiliser la fonction BAC et les points de terminaison.
- Renvoie la valeur booléenne `true` si l'utilisateur a été créé avec succès. Retourne dans `false` le cas contraire.
- Exemple de demande JSON :

```
{
    "login": "simpleuser",
    "password": "simplepassword",
    "roles": [
      {
        "id": 2,
        "roleName": "ROLE_USER"
      }
    ]
  }
```

Voici les deux valeurs valides pour `roleName` :

- `ROLE_ADMIN`: peut gérer les ressources et les utilisateurs de Blusam.
- `ROLE_USER`: peut gérer les ressources de Blusam mais pas les utilisateurs.

Obtenir des informations sur l'utilisateur

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogin/userInfo`
- Arguments : Aucun
- Renvoie le nom d'utilisateur et le rôle de l'utilisateur actuellement connecté

Répertorier des utilisateurs

- Méthode prise en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/security/servicelogin/listusers`
- Arguments : Aucun
- Renvoie une liste de`com.netfective.bluage.bac.entities.SignOn`, sérialisée au format JSON.

Suppression d'un utilisateur

> **Important**
>
> Cette action ne peut pas être annulée. L'utilisateur supprimé ne pourra pas se reconnecter à l'application BAC.

- Méthode prise en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/security/servicelogin/deleteuser`
- Arguments : la sérialisation JSON d'un `com.netfective.bluage.bac.entities.SignOn` objet qui représente l'utilisateur à supprimer du stockage.
- Renvoie la valeur booléenne `true` si l'utilisateur a été correctement supprimé.

Déconnectez l'utilisateur actuel

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogout/logout`
- Arguments : Aucun
- Renvoie le message JSON `{"success":true}` si l'utilisateur actuel s'est déconnecté avec succès. La session HTTP associée sera invalidée.

Gérer la console d'applications JICS dans AWS Blu Age

Le composant JICS est le support AWS Blu Age pour la modernisation des ressources CICS existantes. L'application Web de console d'application JICS est dédiée à l'administration des ressources JICS. Les points de terminaison suivants permettent d'effectuer les tâches d'administration sans avoir à interagir avec l'interface utilisateur JAC. Chaque fois qu'un point de terminaison nécessite une authentification, la demande doit inclure les détails de l'authentification (nom d'utilisateur/mot de passe généralement, comme l'exige l'authentification de base). Les points de terminaison de l'application Web de la console d'application JICS utilisent le chemin racine. `/jac/`

Rubriques

- Gestion des ressources JICS
- Autre
- Points de terminaison de gestion des utilisateurs JAC

Gestion des ressources JICS

Tous les points de terminaison suivants sont liés à la gestion des ressources du JICS, ce qui permet aux administrateurs du JICS de gérer les ressources au quotidien.

Rubriques

- Liste des listes et des groupes JICS
- Récupérez les ressources JICS
- Liste des groupes JICS
- Lister les groupes JICS pour une liste donnée
- LISTER les ressources JICS pour un GROUPE donné

- LISTER les ressources JICS pour un GROUPE donné (alternative en utilisant un nom)
- Modification des GROUPES détenus de plusieurs LISTES
- Supprimer une LISTE
- Supprimer un GROUPE
- Supprimer une TRANSACTION
- Supprimer un PROGRAMME
- Supprimer un FICHIER
- Supprimer une TDQUEUE
- Supprimer un TSMODEL
- Supprimer des éléments
- Créez une LISTE
- Créez un GROUPE
- Considérations communes relatives à la création de RESSOURCES
- Création d'une TRANSACTION
- Création d'un PROGRAMME
- Création d'un FICHIER
- Créez une TDQUEUE
- Créez un TSMODEL
- Création d'éléments
- Mettre à jour une LISTE
- Mettre à jour un GROUPE
- Considérations courantes relatives à la mise à jour
- Mettre à jour une TRANSACTION
- Mettre à jour un PROGRAMME
- Mettre à jour un FICHIER
- Mettre à jour une TDQUEUE
- Mettre à jour un TSMODEL
- Mettre à jour les éléments
- Éléments en haut

- Récupérer des éléments
- Fonctionnement du JICS CRUD

Liste des listes et des groupes JICS

LIST et GROUPS sont les principales ressources de conteneur propriétaires au sein du composant JICS. Toutes les ressources JICS doivent appartenir à un GROUPE. Les groupes peuvent appartenir à des LISTES, mais cela n'est pas obligatoire. Les LISTS peuvent même ne pas exister dans un environnement JICS donné, mais la plupart du temps, les LISTS sont là pour donner une couche supplémentaire d'organisation aux ressources. Pour plus d'informations sur l'organisation des ressources du CICS, consultez les ressources du CICS.

- Méthode prise en charge : GET
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/listJicsListsAndGroups`
- Arguments : Aucun
- Renvoie : une liste d' JicsContainer objets sérialisés, à la fois LISTS et GROUPS, au format JSON.

Exemple de réponse :

```
[
    {
      "name": "Resources",
      "children": [
        {
          "jacType": "JACList",
          "name": "MURACHS",
          "isActive": true,
          "children": [
            {
              "jacType": "JACGroup",
              "name": "MURACHS",
              "isActive": true,
              "children": []
            }
          ]
        },
```

```
        {
          "jacType": "JACGroup",
          "name": "TEST",
          "isActive": true,
          "children": []
        }
      ],
      "isExpanded": true
    }
  ]
```

Récupérez les ressources JICS

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/retrieveJicsResources`
- Arguments : charge utile JSON qui représente les ressources JICS que vous souhaitez récupérer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.request.RetrieveOperationRequest` objet.
- Retours : liste d' JicsResource objets sérialisés. Les objets sont renvoyés sans ordre particulier et sont de types différents, tels que PROGRAM, TRANSACTION, FILE, etc.

Liste des groupes JICS

- Méthode prise en charge : GET
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/listJicsGroups`
- Arguments : Aucun
- Renvoie une liste d' JicsContainer objets sérialisés (GROUPS) au format JSON. Les GROUPES sont renvoyés sans leurs propres informations LIST.

Exemple de réponse :

```
[
```

```
  {
    "jacType": "JACGroup",
    "name": "MURACHS",
    "isActive": true,
    "children": []
  },
  {
    "jacType": "JACGroup",
    "name": "TEST",
    "isActive": true,
    "children": []
  }
]
```

Lister les groupes JICS pour une liste donnée

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/listGroupsForList`
- Arguments : une charge utile JSON, représentant la LISTE JICS dont vous recherchez les GROUPES. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACList` objet.

  Demande d'échantillon :

```
{
    "jacType":"JACList",
    "name":"MURACHS",
    "isActive":true
  }
```

- Renvoie une liste d' JicsContainer objets sérialisés (GROUPS) au format JSON, qui sont attachés à la LIST donnée. Les GROUPES sont renvoyés sans leurs propres informations LIST.

  Exemple de réponse :

```
[
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
```

```
      "isActive": true,
      "children": []
    }
  ]
```

LISTER les ressources JICS pour un GROUPE donné

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/listResourcesForGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS dont vous recherchez les ressources. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet. Il n'est pas nécessaire de spécifier tous les champs pour le GROUPE, mais le nom est obligatoire.

  Demande d'échantillon :

```
{
    "jacType":"JACGroup",
    "name":"MURACHS",
    "isActive":true
  }
```

- Renvoie une liste d' JicsResource objets sérialisés appartenant au GROUPE donné. Les objets sont renvoyés sans ordre particulier et sont de types différents, tels que PROGRAM, TRANSACTION, FILE, etc.

LISTER les ressources JICS pour un GROUPE donné (alternative en utilisant un nom)

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/listResourcesForGroupName`
- Arguments : le nom du GROUPE propriétaire des ressources que vous recherchez.
- Renvoie : une liste d' JicsResource objets sérialisés, appartenant au GROUPE donné. Les objets sont renvoyés sans ordre particulier et sont de types différents, tels que PROGRAM, TRANSACTION, FILE, etc.

Modification des GROUPES détenus de plusieurs LISTES

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/editGroupsList`
- Arguments : une représentation JSON d'une collection de LISTES avec des groupes enfants ;

  Demande d'échantillon :

```
[
  {
        "jacType": "JACList",
        "name": "MURACHS",
        "isActive": true,
        "children": [
          {
            "jacType": "JACGroup",
            "name": "MURACHS",
            "isActive": true,
            "children": []
          },
          {
            "jacType": "JACGroup",
            "name": "TEST",
            "isActive": true,
            "children": []
          }
        ]
  }
]
```

  Avant cette édition, seul le groupe nommé « MURACHS » appartenait à la LISTE nommée « MURACHS ». Avec cette édition, vous « ajoutez » le groupe nommé « TEST » à la LISTE nommée « MURACHS ».

- Renvoie une valeur booléenne. Si la valeur est « vraie », les modifications LISTS ont été correctement conservées dans le stockage JICS sous-jacent.

Supprimer une LISTE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteList`
- Arguments : une charge utile JSON, représentant la LISTE JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACList` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de la LIST a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer un GROUPE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression du GROUPE a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer une TRANSACTION

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteTransaction`
- Arguments : une charge utile JSON, représentant la transaction JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTransaction` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de TRANSACTION a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer un PROGRAMME

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteProgram`
- Arguments : une charge utile JSON, représentant le programme JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACProgram` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression du PROGRAMME a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer un FICHIER

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteFile`
- Arguments : une charge utile JSON, représentant le fichier JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACFile` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression du FICHIER a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer une TDQUEUE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteTDQueue`
- Arguments : une charge utile JSON, représentant la JICS TDQUEUE à supprimer. Il s'agit de la sérialisation JSON d'un fichier `com.netfective.bluage.jac.entities. JACTDQueue`objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de TDQUEUE a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer un TSMODEL

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteTSModel`
- Arguments : une charge utile JSON, représentant le JICS TSMODEL à supprimer. Il s'agit de la sérialisation JSON d'un fichier `com.netfective.bluage.jac.entities. JACTSModel`objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de TSMODEL a été effectuée avec succès sur le stockage JICS sous-jacent.

Supprimer des éléments

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/deleteElements`
- Arguments : charge utile JSON qui représente les éléments JICS à supprimer.
- Renvoie une valeur booléenne `true` indiquant que la suppression a été effectuée avec succès dans le stockage JICS sous-jacent.

Créez une LISTE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createList`
- Arguments : une charge utile JSON, représentant la liste JICS à créer. Il s'agit de la sérialisation JSON d'un fichier `com.netfective.bluage.jac.entities. JACList`objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la LIST a été créée avec succès dans le stockage JICS sous-jacent.

> **Note**
>
> La LISTE sera toujours créée vide. L'ajout de GROUPES à la LISTE nécessitera une autre opération.

Créez un GROUPE

- Méthode prise en charge : POST
- Nécessite une authentification et les rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le GROUPE a été correctement créé dans le stockage JICS sous-jacent.

> **Note**
>
> Le GROUPE sera toujours créé vide. L'attachement de RESSOURCES au GROUPE nécessitera des opérations supplémentaires (la création de ressources les associera automatiquement à un GROUPE donné).

Considérations communes relatives à la création de RESSOURCES

Tous les points de terminaison suivants sont liés à la création de JICS RESOURCES et partagent certaines contraintes communes : dans la charge utile de la demande à envoyer au point de terminaison, le `groupName` champ doit être valorisé.

Contrainte de propriété du GROUPE :

Aucune ressource ne peut être créée sans être attachée à un groupe existant, et le point de terminaison utilise le GroupName pour récupérer le groupe auquel cette ressource sera attachée. Le `groupName` doit pointer vers le nom d'un GROUPE existant. Un message d'erreur avec HTTP STATUS 400 sera envoyé s'il ne pointe pas vers un groupe existant dans le stockage sous-jacent JICS. `groupName`

Contrainte d'unicité au sein d'un GROUPE :

Une ressource donnée portant un nom donné doit être unique au sein d'un groupe donné. La vérification de l'unicité sera effectuée par chaque point de terminaison de création de ressources. Si la charge utile donnée ne respecte pas la contrainte d'unicité, le point de terminaison enverra une réponse avec HTTP STATUS 400 (BAD REQUEST). Voir l'exemple de réponse ci-dessous.

Exemple de charge utile : vous essayez de créer la transaction « ARIT » dans le groupe « TEST », mais une transaction portant ce nom existe déjà dans ce groupe.

```
{
    "jacType":"JACTransaction",
    "name":"ARIT",
    "groupName":"TEST",
    "isActive":true
  }
```

Vous recevez la réponse d'erreur suivante :

```
{
    "timestamp": 1686759054510,
    "status": 400,
    "error": "Bad Request",
    "path": "/jac/api/services/rest/jicsservice/createTransaction"
  }
```

L'inspection des journaux des serveurs confirmera l'origine du problème :

```
2023-06-14 18:10:54 default         TRACE - o.s.w.m.HandlerMethod
        - Arguments: [java.lang.IllegalArgumentException: Transaction already
 present in the group, org.springframework.security.web.header.HeaderWriterFilter
$HeaderWriterResponse@e34f6b8]
2023-06-14 18:10:54 default         ERROR - c.n.b.j.a.WebConfig                    -
 400
java.lang.IllegalArgumentException: Transaction already present in the group
 at
 com.netfective.bluage.jac.server.services.rest.impl.JicsServiceImpl.createElement(JicsServiceI
```

Création d'une TRANSACTION

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createTransaction`
- Arguments : une charge utile JSON, représentant la TRANSACTION JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTransaction` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TRANSACTION a été créée avec succès dans le stockage JICS sous-jacent.

Création d'un PROGRAMME

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createProgram`
- Arguments : une charge utile JSON, représentant le PROGRAMME JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACProgram` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le PROGRAMME a été créé avec succès dans le stockage JICS sous-jacent.

Création d'un FICHIER

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createFile`
- Arguments : une charge utile JSON, représentant le FICHIER JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACFile` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le FICHIER a été créé avec succès dans le stockage JICS sous-jacent.

Créez une TDQUEUE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createTDQueue`
- Arguments : une charge utile JSON, représentant la JICS TDQUEUE à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTDQueue` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TDQUEUE a été créée avec succès dans le stockage JICS sous-jacent.

Créez un TSMODEL

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createTSModel`
- Arguments : une charge utile JSON, représentant le JICS TSMODEL à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTSModel` objet.
- Renvoie une valeur booléenne `true` indiquant que la création d'éléments a été effectuée avec succès dans le stockage JICS sous-jacent.

Création d'éléments

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/createElements`
- Arguments : une charge utile JSON qui représente les éléments JICS à créer.
- Renvoie une valeur booléenne. Si la valeur est « vraie », les éléments ont été créés avec succès dans le stockage JICS sous-jacent.

Mettre à jour une LISTE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateList`
- Arguments : une charge utile JSON, représentant la LISTE JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACList` objet. Il n'est pas nécessaire de fournir les enfants de la LIST ; le mécanisme de mise à jour de la LISTE ne tiendra pas compte des enfants.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la LIST a été correctement mise à jour dans le stockage JICS sous-jacent.

La mise à jour de l'indicateur LIST « IsActive » se propagera à tous les éléments appartenant à la LISTE, c'est-à-dire à tous les GROUPES appartenant à la LISTE et à toutes les RESSOURCES détenues par ces GROUPES. Il s'agit d'un moyen pratique de désactiver un grand nombre de ressources en une seule opération, sur plusieurs GROUPES.

Mettre à jour un GROUPE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet. Il n'est pas nécessaire de fournir les enfants du GROUPE, le mécanisme de mise à jour du GROUPE n'en tiendra pas compte.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le GROUPE a été correctement mis à jour dans le stockage JICS sous-jacent.

> **Note**
>
> La mise à jour de l'indicateur « IsActive » du GROUPE se propagera à tous les éléments appartenant au GROUPE, c'est-à-dire à toutes les RESSOURCES détenues par le

GROUPE. Il s'agit d'un moyen pratique de désactiver un grand nombre de ressources en une seule opération au sein d'un GROUPE donné.

Considérations courantes relatives à la mise à jour

Tous les points de terminaison suivants concernent la mise à jour des RESSOURCES JICS. À l'aide de `groupName` ce champ, vous pouvez modifier le GROUPE propriétaire de n'importe quelle RESSOURCE JICS, à condition que la valeur du champ pointe vers un GROUPE existant dans le stockage JICS sous-jacent (sinon, vous recevrez une réponse BAD REQUEST (HTTP STATUS 400) de la part du point de terminaison).

Mettre à jour une TRANSACTION

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateTransaction`
- Arguments : une charge utile JSON, représentant la transaction JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTransaction` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TRANSACTION a été correctement mise à jour dans le stockage JICS sous-jacent.

Mettre à jour un PROGRAMME

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateProgram`
- Arguments : une charge utile JSON, représentant le PROGRAMME JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACProgram` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le PROGRAMME a été correctement mis à jour dans le stockage JICS sous-jacent.

Mettre à jour un FICHIER

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateFile`
- Arguments : une charge utile JSON, représentant le FICHIER JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACFile` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le FICHIER a été correctement mis à jour dans le stockage JICS sous-jacent.

Mettre à jour une TDQUEUE

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateTDQueue`
- Arguments : une charge utile JSON, représentant le JICS TDQUEUE à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTDQueue` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », TDQueue elle a été correctement mise à jour dans le stockage JICS sous-jacent.

Mettre à jour un TSMODEL

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateTSModel`
- Arguments : une charge utile JSON, représentant le JICS TSMODEL à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTSModel` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le TSMODEL a été correctement mis à jour dans le stockage JICS sous-jacent.

Mettre à jour les éléments

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/updateElements`
- Arguments : charge utile JSON qui représente les éléments à mettre à jour.
- Renvoie une valeur booléenne `true` indiquant que la mise à jour des éléments a été effectuée avec succès dans le stockage JICS sous-jacent.

Éléments en haut

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/upsertElements`
- Arguments : charge utile JSON qui représente les éléments à modifier.
- Renvoie une valeur booléenne `true` indiquant que l'élément upsert a été correctement utilisé dans le stockage JICS sous-jacent.

Récupérer des éléments

- Méthode prise en charge : GET
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Trajectoire : `/api/services/rest/jicsservice/retrieveElements`
- Arguments : Aucun
- Renvoie une liste de toutes les ressources JICS sérialisées.

Fonctionnement du JICS CRUD

- Méthode prise en charge : POST
- Nécessite une authentification et l'un des rôles suivants : ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER

- Trajectoire : `/api/services/rest/jicsservice/jicsCrudOperation`
- Arguments : une charge utile JSON qui représente les ressources JICS que vous recherchez. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.request.JicsCrudOperationRequest` objet.
- Renvoie une charge utile JSON qui représente la réponse. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.request.JicsCrudOperationResponse` objet.

Autre

Rubriques

- État de santé du serveur JICS

État de santé du serveur JICS

- Méthode prise en charge : GET
- Trajectoire : `/api/services/rest/jicsserver/serverIsUp`
- Arguments : Aucun
- Retours : Aucun. Une réponse HTTP STATUS 200 indique que le serveur est opérationnel.

Points de terminaison de gestion des utilisateurs JAC

Utilisez les points de terminaison suivants pour gérer les interactions des utilisateurs.

Rubriques

- Connexion d'un utilisateur
- Tester si au moins un utilisateur existe dans le système
- Enregistrer un nouvel utilisateur
- Informations sur l'utilisateur
- Lister les utilisateurs
- Suppression d'un utilisateur
- Déconnecter l'utilisateur actuel

Connexion d'un utilisateur

- Méthode prise en charge : POST
- Trajectoire : `/api/services/security/servicelogin/login`
- Arguments : Aucun
- Renvoie la sérialisation JSON d'un `com.netfective.bluage.jac.entities.SignOn` objet, représentant l'utilisateur dont les informations d'identification sont fournies dans la demande en cours. Le mot de passe est masqué dans la vue dans l'objet renvoyé. Les rôles attribués à l'utilisateur sont listés.

Exemple de réponse :

```
{
    "login": "some-admin",
    "password": null,
    "roles": [
      {
        "id": 0,
        "roleName": "ROLE_ADMIN"
      }
    ]
  }
```

Tester si au moins un utilisateur existe dans le système

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogin/hasAccount`
- Arguments : Aucun
- Renvoie la valeur booléenne `true` si au moins un utilisateur autre que le super administrateur par défaut a été créé. Retourne dans `false` le cas contraire.

Enregistrer un nouvel utilisateur

- Méthode prise en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/security/servicelogin/recorduser`

- Arguments : la sérialisation JSON d'un `com.netfective.bluage.jac.entities.SignOn` objet, représentant l'utilisateur à ajouter au stockage. Les rôles de l'utilisateur doivent être définis, sinon l'utilisateur risque de ne pas être en mesure d'utiliser la fonction JAC et les points de terminaison.
- Renvoie la valeur booléenne `true` si l'utilisateur a été créé avec succès. Retourne dans `false` le cas contraire.

Demande d'échantillon :

```
{
    "login": "simpleuser",
    "password": "simplepassword",
    "roles": [
      {
        "id": 2,
        "roleName": "ROLE_USER"
      }
    ]
  }
```

Seuls les rôles suivants peuvent être utilisés lors de l'enregistrement d'un nouvel utilisateur :

- ROLE_ADMIN : peut gérer les ressources et les utilisateurs du JICS.
- ROLE_USER : peut gérer les ressources JICS mais pas les utilisateurs.

Informations sur l'utilisateur

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogin/userInfo`
- Arguments : Aucun
- Renvoie le nom d'utilisateur et les rôles de l'utilisateur actuellement connecté.

Lister les utilisateurs

- Méthode prise en charge : GET
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/security/servicelogin/listusers`

- Arguments : Aucun
- Renvoie une liste de `com.netfective.bluage.jac.entities.SignOn`, sérialisée au format JSON.

Suppression d'un utilisateur

- Méthode prise en charge : POST
- Nécessite une authentification et le rôle ROLE_ADMIN.
- Trajectoire : `/api/services/security/servicelogin/deleteuser`
- Arguments : la sérialisation JSON d'un `com.netfective.bluage.jac.entities.SignOn` objet qui représente l'utilisateur à supprimer du stockage.
- Renvoie la valeur booléenne `true` si l'utilisateur a été correctement supprimé.

> **Important**
>
> Cette action ne peut pas être annulée. L'utilisateur supprimé ne pourra pas se reconnecter à l'application JAC.

Déconnecter l'utilisateur actuel

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogout/logout`
- Arguments : Aucun
- Renvoie le message JSON `{"success":true}` si l'utilisateur actuel s'est déconnecté avec succès. La session HTTP associée sera invalidée.

Structures de données pour les utilisateurs de AWS Blu Age

Vous pouvez en savoir plus sur les différentes structures de données du moteur AWS Blu Age dans la section suivante.

Rubriques

- Structure du message détaillée sur l'exécution des tâches

- Structure des résultats du lancement de la transaction
- Structure des résultats de l'enregistrement du lancement de la transaction
- État possible d'une tâche dans une file d'attente
- Soumettre une tâche et planifier la saisie d'une tâche
- Réponse à la liste des tâches planifiées
- Liste des réponses aux tâches répétitives

Structure du message détaillée sur l'exécution des tâches

Les détails de l'exécution de chaque tâche comporteront les champs suivants :

ID du script

l'identifiant du script appelé.

appelant

Adresse IP de l'appelant.

identifiant

identifiant unique d'exécution de la tâche.

startTime

date et heure auxquelles l'exécution de la tâche a commencé.

endTime

date et heure auxquelles l'exécution de la tâche s'est terminée.

status

un statut pour l'exécution de la tâche. Une valeur possible parmi :

- `DONE`: l'exécution de la tâche s'est terminée normalement.
- `TRIGGERED`: exécution de la tâche déclenchée mais pas encore lancée.
- `RUNNING`: l'exécution de la tâche est en cours.
- `KILLED`: l'exécution du travail a été supprimée.
- `FAILED`: échec de l'exécution de la tâche.

Résultat de l'exécution

un message résumant le résultat de l'exécution de la tâche. Ce message peut être soit un simple message si l'exécution de la tâche n'est pas encore terminée, soit une structure JSON contenant les champs suivants :

- ExitCode : code de sortie numérique ; les valeurs négatives indiquent des situations de défaillance.
- programme : dernier programme lancé par le job.
- status : une valeur possible parmi :
  - `Error`: lorsque ExitCode = -1 ; cela correspond à une erreur (technique) survenue lors de l'exécution du job.
  - `Failed`: lorsque exitcode = -2 ; Cela correspond à une défaillance survenant lors de l'exécution d'un programme de service (comme dans une situation ABEND).
  - `Succeeded`: lorsque ExitCode >= 0 ;
- StepName : nom de la dernière étape exécutée dans le job.

Mode d'exécution

SYNCHRONE ou ASYNCHRONE, selon la manière dont la tâche a été lancée.

Exemple de sortie :

```
{
    "scriptId": "INTCALC",
    "caller": "127.0.0.1",
    "identifier": "97d410be-efa7-4bd3-b7b9-d080e5769771",
    "startTime": "06-09-2023 11:42:41",
    "endTime": "06-09-2023 11:42:42",
    "status": "DONE",
    "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
 \"CBACT04C\", \"status\": \"Error\" }",
    "executionMode": "ASYNCHRONOUS"
  }
```

Structure des résultats du lancement de la transaction

La structure peut contenir les champs suivants :

Résultat

une chaîne représentant le résultat de l'exécution de la transaction. Les valeurs possibles sont :

- `Success`: l'exécution de la transaction s'est terminée correctement.
- `Failure`: l'exécution de la transaction ne s'est pas terminée correctement, certains problèmes sont survenus.

commarée

une chaîne représentant la valeur finale de COMMAREA, sous la forme d'un tableau d'octets codé par 64 octets. Il peut s'agir d'une chaîne vide.

Container Record

(Facultatif) une chaîne représentant le contenu de l'enregistrement du CONTENEUR sous la forme d'un tableau d'octets codé sur 64 octets.

Description du serveur

Peut contenir des informations sur le serveur qui a répondu à la demande (à des fins de débogage). Il peut s'agir d'une chaîne vide.

Un code Bend

(Facultatif) Si le programme référencé par la transaction lancée a été modifié, la valeur du code d'abend sera renvoyée sous forme de chaîne dans ce champ.

Exemples de réponses :

Réussite

```
{
    "outCome": "Success",
    "commarea": "",
    "serverDescription": ""
  }
```

Échec

```
{
    "outCome": "Failure",
    "commarea": "",
    "serverDescription": "",
    "abendCode": "AEIA"
```

```
}
```

Structure des résultats de l'enregistrement du lancement de la transaction

La structure peut contenir les champs suivants :

Enregistrer le contenu

une chaîne représentant le contenu de l'enregistrement de la COMMAREA sous la forme d'un tableau d'octets codé par 64 octets.

Container Record

une chaîne représentant le contenu de l'enregistrement du CONTENEUR sous la forme d'un tableau d'octets codé sur 64 octets.

Description du serveur

Peut contenir des informations sur le serveur qui a répondu à la demande (à des fins de débogage). Il peut s'agir d'une chaîne vide.

Exemples de réponses :

Réussite

```
{
    "recordContent": "",
    "serverDescription": ""
}
```

État possible d'une tâche dans une file d'attente

Dans une file d'attente, les tâches peuvent avoir le statut suivant :

ACTIF

La tâche est actuellement exécutée dans la file d'attente.

EXECUTION_WAIT

La tâche attend qu'un fil de discussion soit disponible.

PLANIFIÉ

Les tâches sont planifiées pour être exécutées à une date et à une heure spécifiques.

HOLD

Job attend d'être publié avant d'être exécuté.

TERMINÉ

Job exécuté avec succès.

ÉCHEC

L'exécution du Job a échoué.

UNKNOWN

Le statut est inconnu.

Soumettre une tâche et planifier la saisie d'une tâche

L'entrée de la tâche d'envoi et de la tâche de planification est la sérialisation JSON d'un `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objet. L'exemple d'entrée ci-dessous présente tous les champs correspondant à un tel haricot.

Exemple de saisie pour soumettre une offre d'emploi :

```
{
    "messageQueueName":null,
    "scheduleDate":null,
    "scheduleTime":null,
    "programName":"PTA0044",
    "programParams":
     {"wmind":"B"},
    "localDataAreaValue":"",
    "userName":"USER1",
    "jobName":"PTA0044",
    "jobNumber":9,
    "jobPriority":5,
    "executionDate":"20181231",
    "jobQueue":"queue1",
    "jobOnHold":false
}
```

Exemple de saisie pour une tâche planifiée :

```
{
```

```
    "scheduleCron": "*/2 * * * * ?",
    "programName":"LOGPGM",
    "programParams": {
        "cl_sbmjob_param_json": "[\"./output/schedule-job-log.txt\", \"Every 2
seconds!\"]"
    },
    "localDataAreaValue":"",
    "userName":"PVO",
    "jobName":"LOGGERJOB",
    "jobPriority":5,
    "jobQueue":"queue1",
    "scheduleMisfirePolicy": 4,
    "startTime": "2003/05/04 07:00:00.000 GMT-06:00",
    "endTime": "2003/05/04 07:00:07.000 GMT-06:00"
}
```

Numéro de poste

si le numéro de tâche est 0, le numéro de tâche sera automatiquement généré en utilisant le numéro suivant dans la séquence des numéros de tâche. Cette valeur doit être définie sur 0 (sauf à des fins de test).

Priorité de l'emploi

La priorité de tâche par défaut dans AS4 00 est 5. La plage valide est comprise entre 0 et 9, 0 étant la priorité la plus élevée.

jobOnHold

Si une tâche est soumise en attente, elle ne sera pas exécutée immédiatement, mais uniquement lorsque quelqu'un la « publie ». Une tâche peut être publiée à l'aide de l'API REST (/release ou / release-all).

ScheduleDate et ScheduleTime

Si ces valeurs ne sont pas nulles, la tâche sera exécutée à la date et à l'heure spécifiées.

Date

Peut être fourni avec format MMddyy ou dd MMyyyy (la taille de l'entrée déterminera le format utilisé)

Heure

Peut être fourni avec un format HHmm ou HHmmss (la taille de l'entrée déterminera le format utilisé)

Paramètres du programme

Sera transmis au programme sous forme de carte.

scheduleMisfirePolicy

Définit la stratégie utilisée lorsqu'un déclencheur n'est pas activé. Les valeurs possibles sont les suivantes :

1. Relâchez le premier raté et éliminez les autres ratés.
2. Soumettez une tâche en attente pour le premier raté et supprimez les autres ratés.
3. Éliminez le raté.
4. Libérez tous les ratés. La file d'attente exécutera toutes les tâches.

Réponse à la liste des tâches planifiées

Il s'agit de la structure du point de terminaison de la file d'attente des tâches list-jobs. Le message d'envoi de la tâche qui a été utilisé pour soumettre cette tâche fait partie de la réponse. Cela peut être utilisé à des fins de suivi, de test/de soumission à nouveau. Lorsqu'une tâche est terminée, les dates de début et de fin sont également renseignées.

```
[
  {
    "jobName": "PTA0044",
    "userName": "USER1",
    "jobNumber": 9,
    "jobPriority": 5,
    "status": "HOLD",
    "jobDelay": 0,
    "startDate": null,
    "endDate": null,
    "jobQueue": "queue1",
    "message": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "programName": "PTA0044",
      "programParams": {"wmind": "B"},
      "localDataAreaValue": "",
      "userName": "USER1",
      "jobName": "PTA0044",
      "jobNumber": 9,
```

```
      "jobPriority": 5,
      "executionDate": "20181231",
      "jobQueue": "queue1",
      "jobOnHold": true,
      "scheduleCron": null,
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "executionId": 1,
    "jobScheduledId": 0,
    "jobScheduledAt": null
  },
  {
    "jobName": "PTA0044",
    "userName": "USER1",
    "jobNumber": 9,
    "jobPriority": 5,
    "status": "COMPLETED",
    "jobDelay": 0,
    "startDate": "2022-10-13T22:48:34.025+00:00",
    "endDate": "2022-10-13T22:52:54.475+00:00",
    "jobQueue": "queue1",
    "message": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "programName": "PTA0044",
      "programParams": {"wmind": "B"},
      "localDataAreaValue": "",
      "userName": "USER1",
      "jobName": "PTA0044",
      "jobNumber": 9,
      "jobPriority": 5,
      "executionDate": "20181231",
      "jobQueue": "queue1",
      "jobOnHold": true,
      "scheduleCron": "*/20 * * * * ?",
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "executionId": 2,
    "jobScheduledId": 0,
```

```
    "jobScheduledAt": null
  }
]
```

Liste des réponses aux tâches répétitives

Il s'agit de la structure du point de terminaison de la file d'attente des the /schedule/list tâches.

```
[
  {
    "id": 1,
    "status": "ACTIVE",
    "jobNumber": 1,
    "userName": "PVO",
    "msg": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "startTime": "2024/03/07 21:12:00.000 UTC",
      "endTime": "2024/03/07 21:13:59.000 UTC",
      "programName": "LOGPGM",
      "programParams": {"cl_sbmjob_param_json": "[\"./output/schedule-job-log.txt\",
 \"Every 20 seconds!\"]"},
      "localDataAreaValue": "",
      "userName": "PVO",
      "jobName": "LOGGERJOB",
      "jobNumber": 1,
      "jobScheduleId": 1,
      "jobPriority": 5,
      "executionDate": null,
      "jobQueue": "queue1",
      "jobOnHold": false,
      "scheduleCron": "*/20 * * * * ?",
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "lastUpdatedAt": "2024-03-07T21:11:13.282+00:00",
    "lastUpdatedBy": ""
  }
]
```

Configurer AWS Blu Age Runtime (non géré)

Cette section explique les étapes à suivre pour configurer AWS Blu Age Runtime (non géré) sur votre AWS infrastructure. Avant de configurer votre AWS Blu Age Runtime (non géré) pour les applications, prenez connaissance des prérequis, des régions et des compartiments, ainsi que de la configuration des CloudWatch alarmes pour configurer et gérer votre environnement d'exécution.

Rubriques

- AWS Prérequis pour Blu Age Runtime
- Intégration à AWS Blue Age Runtime
- Exigences de configuration de l'infrastructure pour AWS Blu Age Runtime (non géré)
- AWS Artefacts de Blu Age Runtime
- Déployez AWS Blu Age Runtime sur Amazon EC2
- Déployez AWS Blu Age Runtime sur des conteneurs sur Amazon ECS et Amazon EKS
- Testez l' PlanetsDemo application

AWS Prérequis pour Blu Age Runtime

AWS Blu Age Runtime (non géré) est disponible en plusieurs the section called "AWS Notes de publication de Blu Age" versions. Si vous avez des projets de modernisation en cours, vous pourriez avoir besoin de versions incrémentielles du runtime à des fins de mise en œuvre et de test. Pour définir vos besoins, contactez votre responsable de livraison AWS Blu Age.

Avant de commencer le processus d'intégration (non géré) de AWS Blu Age Runtime, procédez comme suit :

- Assurez-vous d'avoir un AWS compte.
- Assurez-vous de disposer d'une application modernisée refactorisée avec Blu Age. AWS
- Choisissez une AWS région et l'une des options de calcul prises en charge pour AWS Blu Age Runtime (non géré).
- Choisissez la version de AWS Blu Age Runtime que vous souhaitez utiliser.
- Vérifiez the section called "Exigences de configuration de l'infrastructure" et validez les composants supplémentaires requis pour exécuter le AWS Blu Age Runtime (non géré).

> **Note**
>
> Si vous souhaitez tester les fonctionnalités de AWS Blu Age Runtime (non géré), vous pouvez utiliser l'application de démonstration`Planets Demo`, que vous pouvez télécharger depuis le PlanetsDemofichier -v1.zip.

Intégration à AWS Blue Age Runtime

Pour commencer, créez un AWS Supportdossier pour demander l'intégration afin d'accéder à AWS Blu Age Runtime. Incluez dans votre demande votre Compte AWS identifiant, la AWS région que vous souhaitez utiliser, un choix de calcul, ainsi que la version de AWS Blu Age Runtime. Si vous n'êtes pas sûr de la version dont vous avez besoin, contactez votre responsable de livraison AWS Blu Age. Si le code source de votre application est déjà généré par les outils de refactorisation de la modernisation du AWS mainframe, notez la valeur de la `gapwalk.version` balise dans le `pom.xml` fichier de votre base de code modernisée.

> **Note**
>
> Le AWS Blu Age Runtime est disponible en deux versions principales : les pré-versions Alpha et les Releases. Pour déterminer la version à utiliserthe section called "AWS Versionnage de Blu Age", consultez ou contactez votre responsable de livraison AWS Blu Age.

Régions et compartiments pour AWS Blu Age Runtime (non gérés)

Nous stockons les artefacts AWS Blu Age Runtime (non gérés) dans différents compartiments Amazon S3 par région et par choix de calcul. Pour accéder au bucket correspondant à votre environnement d'exécution Région AWS pour AWS Blu Age (non géré), utilisez le nom indiqué dans le tableau suivant.

Région AWS	Seau de déverrouillage	Seau de pré-sortie Alpha
USA Est (Ohio)	aws-bluage-runtime-artifacts-055777665268-us-est-2	aws-bluage-runtime-artifacts-dev-055777665268-us-est-2
USA Est (Virginie du Nord)	aws-bluage-runtime-artifacts-139023371234-us-est-1	aws-bluage-runtime-artifacts-dev-139023371234-us-est-1

Région AWS	Seau de déverrouillage	Seau de pré-sortie Alpha
USA Ouest (Californie du Nord)	aws-bluage-runtime-artifacts-788454048782-us-ouest-1	aws-bluage-runtime-artifacts-dev-788454048782-us-ouest-1
USA Ouest (Oregon)	aws-bluage-runtime-artifacts-836771190483-us-ouest-2	aws-bluage-runtime-artifacts-dev-836771190483-us-ouest-2
Canada (Centre)	aws-bluage-runtime-artifacts-637423580979-ca-central-1	aws-bluage-runtime-artifacts-dev-637423580979-ca-central-1
Europe (Irlande)	aws-bluage-runtime-artifacts-925278190477-eu-ouest-1	aws-bluage-runtime-artifacts-dev-925278190477-eu-west-1
Europe (Londres)	aws-bluage-runtime-artifacts-767397831990-eu-ouest-1	aws-bluage-runtime-artifacts-dev-767397831990-eu-ouest-1
Europe (Paris)	aws-bluage-runtime-artifacts-673009995881-eu-ouest-3	aws-bluage-runtime-artifacts-dev-673009995881-eu-west-3
Europe (Francfort)	aws-bluage-runtime-artifacts-485196800481-eu-central-1	aws-bluage-runtime-artifacts-dev-485196800481-eu-central-1
Europe (Stockholm)	aws-bluage-runtime-artifacts-654654484534-eu-nord-1	aws-bluage-runtime-artifacts-dev-654654484534-eu-nord-1
Europe (Milan)	aws-bluage-runtime-artifacts-654654328338-eu-sud-1	aws-bluage-runtime-artifacts-dev-654654328338-eu-sud-1
Europe (Espagne)	aws-bluage-runtime-artifacts-905417994954-eu-sud-2	aws-bluage-runtime-artifacts-dev-905417994954-eu-sud-2
Amérique du Sud (São Paulo)	aws-bluage-runtime-artifacts-737536804457-sa-est-1	aws-bluage-runtime-artifacts-dev-737536804457-sa-east-1

Région AWS	Seau de déverrouillage	Seau de pré-sortie Alpha
Asie-Pacifique (Tokyo)	aws-bluage-runtime-artifacts-445578176276-ap-northeast-1	aws-bluage-runtime-artifacts-dev-445578176276-ap-northeast-1
Asie-Pacifique (Séoul)	aws-bluage-runtime-artifacts-381492221498-ap-northeast-2	aws-bluage-runtime-artifacts-dev-381492221498-ap-northeast-2
Asie-Pacifique (Osaka)	aws-bluage-runtime-artifacts-905418229615-ap-northeast-3	aws-bluage-runtime-artifacts-dev-905418229615-ap-northeast-3
Asie-Pacifique (Singapour)	aws-bluage-runtime-artifacts-767397774613-ap-southeast-1	aws-bluage-runtime-artifacts-dev-767397774613-ap-southeast-1
Asie-Pacifique (Sydney)	aws-bluage-runtime-artifacts-726160321909-ap-southeast-2	aws-bluage-runtime-artifacts-dev-726160321909-ap-southeast-2
Asie-Pacifique (Mumbai)	aws-bluage-runtime-artifacts-905418353577-ap-south-1	aws-bluage-runtime-artifacts-dev-905418353577-ap-south-1
Afrique (Le Cap)	aws-bluage-runtime-artifacts-992382777663-af-south-1	aws-bluage-runtime-artifacts-dev-992382777663-af-south-1
Israël (Tel Aviv)	aws-bluage-runtime-artifacts-471112516508-il-central-1	aws-bluage-runtime-artifacts-dev-471112516508-il-central-1

Utiliser le AWS CLI pour répertorier le contenu du bucket

Une fois l'intégration terminée, vous pouvez répertorier le contenu du bucket en exécutant la AWS CLI commande suivante dans un terminal.

```
aws s3 ls bucket-name
```

Remplacez `bucket-name` par le nom du bucket correspondant Région AWS à votre nom indiqué dans le tableau précédent.

Cette commande renvoie une liste de dossiers correspondant aux différentes versions du moteur d'exécution (non géré) de AWS Blu Age Runtime, comme le suivant pour un bucket de version :

```
PRE 3.10.0/
PRE 4.0.0/
```

Ou ce qui suit pour un bucket de construction :

```
PRE 4.1.0-alpha.8/
PRE 4.1.0-alpha.9/
```

Nous vous recommandons d'utiliser la dernière version disponible. Si cela n'est pas possible, utilisez la version d'exécution qui a été validée lors de la phase de refactorisation de l'application. Pour répertorier les frameworks disponibles pour une version spécifique, exécutez la commande suivante :

`aws s3 ls s3://bucket-name/version/Framework/`

`bucket-name`Remplacez-le par le nom du bucket correspondant à votre Région AWS compte et `version` par la version de votre choix. Voici deux exemples.

Pour un compartiment de lancement :

`aws s3 ls s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/`

La commande renvoie une liste de frameworks, tels que :

```
2024-04-08 16:11:19  152040176 aws-bluage-runtime-4.0.0.tar.gz
2024-04-08 16:11:50         45 aws-bluage-runtime-4.0.0.tar.gz.checksumSHA256
2024-04-08 16:11:52  176518889 aws-bluage-webapps-4.0.0.tar.gz
2024-04-08 16:12:28         45 aws-bluage-webapps-4.0.0.tar.gz.checksumSHA256
```

Pour un bucket de construction :

`aws s3 ls s3://aws-bluage-runtime-artifacts-dev-139023371234-us-east-1/4.1.0-alpha.9/Framework/`

La commande renvoie une liste de frameworks, tels que :

```
2024-04-09 20:23:34  152304534 aws-bluage-runtime-4.1.0-alpha.9.tar.gz
2024-04-09 20:24:05         45 aws-bluage-runtime-4.1.0-alpha.9.tar.gz.checksumSHA256
2024-04-09 20:24:07  176262381 aws-bluage-webapps-4.1.0-alpha.9.tar.gz
2024-04-09 20:24:42         45 aws-bluage-webapps-4.1.0-alpha.9.tar.gz.checksumSHA256
```

Téléchargez le framework

Vous pouvez télécharger le framework par exemple pour mettre à niveau la version AWS Blu Age Runtime sur une EC2 instance Amazon existante.

`aws s3 cp s3://`*`bucket-name`*`/`*`version`*`/Framework/ `*`folder-of-your-choice`*` --recursive`

Où :

folder-of-your-choice

chemin du dossier dans lequel vous souhaitez télécharger le framework.

Par exemple : `aws s3 cp s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/ . --recursive`

Cette commande produit le résultat suivant :

```
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/aws-bluage-runtime-4.0.0.tar.gz.checksumSHA256 to ./aws-bluage-
runtime-4.0.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/aws-bluage-webapps-4.0.0.tar.gz.checksumSHA256 to ./aws-bluage-
webapps-4.0.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/aws-
bluage-webapps-4.0.0.tar.gz to ./aws-bluage-webapps-4.0.0.tar.gz
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/aws-
bluage-runtime-4.0.0.tar.gz to ./aws-bluage-runtime-4.0.0.tar.gz
```

Vous pouvez répertorier les fichiers du framework comme suit :

`ls -l`

Cette commande produit le résultat suivant :

```
total 230928
-rw-rw-r-- 1 cloudshell-user cloudshell-user 152040176 Apr  8 16:11 aws-bluage-
runtime-4.0.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Apr  8 16:11 aws-bluage-
runtime-4.0.0.tar.gz.checksumSHA256
-rw-rw-r-- 1 cloudshell-user cloudshell-user 176518889 Apr  8 16:11 aws-bluage-
webapps-4.0.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Apr  8 16:12 aws-bluage-
webapps-4.0.0.tar.gz.checksumSHA256
```

> **Note**
>
> L'accès aux artefacts peut être temporairement interrompu et les versions peuvent être supprimées pour des raisons de sécurité. Nous vous recommandons vivement de stocker les objets que vous utilisez dans votre propre compte. La version locale doit être utilisée comme référence dans vos architectures internes.

Exigences de configuration de l'infrastructure pour AWS Blu Age Runtime (non géré)

Cette rubrique décrit la configuration d'infrastructure minimale requise pour exécuter AWS Blu Age Runtime (non géré). Les procédures suivantes décrivent comment configurer AWS Blu Age Runtime (non géré) sur l'ordinateur de votre choix pour déployer une application modernisée sur le AWS Blu Age Runtime. Les ressources que vous créez doivent se trouver dans un Amazon VPC doté d'un sous-réseau dédié à votre domaine d'application.

Rubriques

- Exigences en matière d'infrastructure
- Types d' EC2 instances Amazon pour AWS Blu Age Runtime (sur Amazon EC2)
- Exécution de AWS Blu Age Runtime sur Amazon EC2
- Exécution de AWS Blu Age Runtime sur Amazon ECS sur Amazon EC2
- Exécution de AWS Blue Age Runtime sur Amazon EKS sur Amazon EC2
- Exécution de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Exigences en matière d'infrastructure

Création d'un groupe de sécurité

Si vous prévoyez de travailler sur des EC2 instances Amazon sur Amazon EKS, ignorez cette procédure car le processus de création du cluster Amazon EKS crée un groupe de sécurité en votre nom. Utilisez ce groupe de sécurité dans les procédures suivantes au lieu d'en créer un nouveau.

1. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
2. Dans le volet de navigation de gauche, sous Sécurité, sélectionnez Groupes de sécurité.
3. Dans le volet central, choisissez Create security group.
4. Dans le champ Nom du groupe de sécurité, entrez**`M2BluagePrivateLink-SG`**.
5. Dans la section Règles entrantes, choisissez Ajouter une règle.
6. Pour Type, choisissez HTTPS.
7. Dans Source, entrez votre VPC CIDR.
8. Dans la section Règles sortantes, choisissez Ajouter une règle.
9. Pour Type, choisissez HTTPS.
10. En regard de Destination, entrez **`0.0.0.0/0`**.
11. Sélectionnez Create security group (Créer un groupe de sécurité).

Création d'un point de terminaison Amazon VPC

1. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
2. Dans le volet de navigation de gauche, sous Virtual private cloud, sélectionnez Endpoints.
3. Dans le volet central, choisissez Create endpoint.
4. Dans la section Services, entrez **`SQS`** dans le champ de recherche, puis sélectionnez le service Amazon SQS correspondant à votre région.
5. Dans la section VPC, sélectionnez l'Amazon VPC que vous avez créé à l'étape précédente.
6. Dans la section Sous-réseaux, sélectionnez le sous-réseau que vous avez créé pour votre domaine d'application.
7. Dans la section Groupes de sécurité, sélectionnez le groupe de sécurité à partir de la procédure précédente.
8. Choisissez Créer un point de terminaison.

Créer une politique IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de gauche, sous Gestion des accès, sélectionnez Politiques.
3. Dans le volet central, choisissez Create policy.
4. Dans la section Éditeur de politique, sélectionnez JSON.
5. Remplacez tout le JSON que vous voyez dans l'éditeur par le JSON suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Si vous avez besoin de plus de détails pour personnaliser votre politique, contactez votre responsable de livraison ou votre responsable de compte AWS Blu Age.

6. Choisissez Next (Suivant).
7. Entrez le nom de la stratégie, puis choisissez Create policy.

Créer un rôle IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de gauche, sous Gestion des accès, sélectionnez Rôles.
3. Dans le volet central, choisissez Create role.

4. Dans la section Cas d'utilisation, en fonction de votre choix de calcul, choisissez l'une des options suivantes :

   - EC2(pour Amazon EC2 et Amazon EKS sur Amazon EC2)
   - Elastic Container Service puis EC2Role for Elastic Container Service (pour Amazon ECS sur Amazon EC2)
   - Elastic Container Service puis Elastic Container Service Task (pour Amazon ECS géré par Fargate)

5. Choisissez Next (Suivant).
6. Dans le champ de recherche, entrez le nom de la politique que vous avez créée précédemment.
7. Cochez la case située à gauche de votre police d'assurance.

   > **Note**
   >
   > Si vous ne pouvez pas ajouter de stratégie, terminez de créer le rôle, puis mettez-le à jour pour ajouter la politique.

8. Choisissez Next (Suivant).
9. Entrez un nom pour le rôle, puis choisissez Créer un rôle.

Types d' EC2 instances Amazon pour AWS Blu Age Runtime (sur Amazon EC2)

Vous trouverez ci-dessous une liste des types d' EC2 instances Amazon que vous pouvez utiliser pour AWS Blu Age Runtime (sur Amazon EC2) lors de la création d' EC2 instances Amazon ou lors de la définition de nœuds de travail Amazon EKS.

Vérifiez que l'instance qui vous intéresse est disponible dans la région que vous souhaitez déployer.

```
t3.small
t3.medium
t3.large
t3.xlarge
t3.2xlarge
t2.small
t2.medium
t2.large
t2.xlarge
t2.2xlarge
```

```
r7a.medium
r7a.large
r7a.xlarge
r7a.2xlarge
r7a.4xlarge
r7a.8xlarge
r7a.12xlarge
r7a.16xlarge
r7a.24xlarge
r7a.32xlarge
r7a.48xlarge
r7a.metal-48xl
r7i.large
r7i.xlarge
r7i.2xlarge
r7i.4xlarge
r7i.8xlarge
r7i.12xlarge
r7i.16xlarge
r7i.24xlarge
r7i.48xlarge
r7i.metal-24xl
r7i.metal-48xl
r6i.xlarge
r6i.large
r6i.4xlarge
r6i.2xlarge
r5b.xlarge
r5b.large
r5b.2xlarge
r3.xlarge
m6i.xlarge
m6i.large
m6i.8xlarge
m6i.4xlarge
m6i.2xlarge
m6i.16xlarge
m5zn.xlarge
m5zn.large
m5zn.3xlarge
m5zn.2xlarge
m5.xlarge
m5.large
m5.8xlarge
```

```
m5.4xlarge
m5.2xlarge
m5.16xlarge
m5.12xlarge
c6i.xlarge
c6i.large
c6i.8xlarge
c6i.4xlarge
c6i.2xlarge
c6i.16xlarge
c5.xlarge
c5.large
c5.9xlarge
c5.4xlarge
c5.2xlarge
c5.18xlarge
c5.12xlarge
```

Exécution de AWS Blu Age Runtime sur Amazon EC2

Pour créer une EC2 instance Amazon, procédez comme suit.

Création d'une EC2 instance Amazon

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Sélectionnez Launch instance (Lancer une instance).
3. Pour Type d'instance, choisissez l'un des types répertoriés dansthe section called "Types d' EC2 instances Amazon pour AWS Blu Age Runtime (sur Amazon EC2)".
4. Dans la section Paire de clés, choisissez une paire de clés existante ou créez-en une nouvelle.
5. Dans la section Paramètres réseau, choisissez Sélectionner un groupe de sécurité existant.
6. Pour les groupes de sécurité communs, choisissez M2 BluagePrivateLink -SG.
7. Développez la section Détails avancés.
8. Pour le profil d'instance IAM, choisissez le rôle IAM que vous avez créé précédemment.
9. Sélectionnez Launch instance (Lancer une instance).

Installez l'application sur l' EC2 instance Amazon

1. Lorsque l'état de l' EC2 instance Amazon passe à En cours d'exécution, connectez-vous à l'instance.

2. Installez les composants logiciels suivants sur l'instance :

   - Environnement d'exécution Java (JRE) 17.
   - Apache Tomcat 10.
   - AWS Blu Age Runtime (sur Amazon EC2). Installez le moteur d'exécution AWS Blu Age à la racine du dossier d'installation d'Apache Tomcat (certains fichiers seront ajoutés tandis que d'autres seront remplacés).

   Pour installer les applications Web supplémentaires fournies avec l'archive AWS Blu Age Runtime, configurez une instance secondaire du serveur Apache Tomcat et décompressez l'archive des applications Web à cet emplacement. Pour obtenir des instructions complètes, veuillez consulter the section called “AWS Artefacts de Blu Age Runtime”.

Exécution de AWS Blu Age Runtime sur Amazon ECS sur Amazon EC2

1. Créez un cluster Amazon ECS, avec des EC2 instances Amazon comme infrastructure sous-jacente. Consultez Getting started with Windows on Amazon EC2 dans le manuel du développeur Amazon Elastic Container Service.
2. Spécifiez le rôle IAM que vous avez créé lors des étapes précédentes.
3. Choisissez l'un des types d'instance répertoriés dansthe section called “Types d' EC2 instances Amazon pour AWS Blu Age Runtime (sur Amazon EC2)”.
4. Dans Paramètres réseau pour les EC2 instances Amazon, choisissez le groupe de sécurité que vous avez créé lors des étapes précédentes.

Exécution de AWS Blue Age Runtime sur Amazon EKS sur Amazon EC2

1. Créez un cluster Amazon EKS. Consultez la section Création d'un cluster Amazon EKS dans le guide de l'utilisateur Amazon EKS.
2. Comme indiqué précédemment, un groupe de sécurité est créé en votre nom. Vous pouvez utiliser ce groupe de sécurité lorsque vous créez le point de terminaison Amazon VPC.
3. Créez un groupe de nœuds. Spécifiez le rôle IAM que vous avez créé lors des étapes précédentes.
4. Choisissez l'un des types d'instance répertoriés dansthe section called “Types d' EC2 instances Amazon pour AWS Blu Age Runtime (sur Amazon EC2)”.

5. Amazon EKS attribuera automatiquement le groupe de sécurité aux EC2 instances Amazon générées.

Exécution de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Créez un cluster Amazon ECS avec AWS Fargate (sans serveur) comme infrastructure sous-jacente. Consultez Getting started with Fargate dans le guide du développeur Amazon Elastic Container Service.

AWS Artefacts de Blu Age Runtime

AWS Les artefacts Blu Age Runtime sont les composants permettant de déployer et d'exécuter des applications modernisées. Ce document décrit les différents types d'artefacts disponibles, leurs emplacements de stockage et la façon d'y accéder.

AWS Artefacts Blu Age Runtime (non gérés)

Accès et stockage des artefacts

Les artefacts AWS Blu Age Runtime pour les déploiements non gérés sont stockés dans des compartiments S3 spécifiques à une région. Chaque version possède son propre dossier dédié, ce qui permet une gestion et un accès faciles aux versions.

Il existe deux types de seaux :

Seaux de déverrouillage

Les compartiments de version contiennent des répertoires pour les versions les plus récemment déployées et suivent la convention de dénomination :`aws-bluage-runtime-artifacts-<accountId>-<region>`.

Seaux de pré-lancement

Les buckets de pré-publication contiennent des répertoires pour les versions alpha correspondant aux dernières pré-versions incrémentielles de courte durée et suivent la convention de dénomination :. `convention: aws-bluage-runtime-artifacts-dev-<accountId>-<region>`

L'accès aux compartiments de production et de pré-version est accordé indépendamment. Pour plus d'informations sur la procédure de demande d'accès et pour plus de détails sur l'organisation du compartiment S3, consultezthe section called “Intégration à AWS Blue Age Runtime ”.

Contenu des artefacts

Dans les compartiments Release et Pre-release, vous trouverez :

aws-bluage-runtime-x.y.z.tar.gz

Cette archive, où x.y.z représente le numéro de version (`major.minor.patch`selon le versionnement sémantique, voirthe section called "AWS Versionnage de Blu Age"), contient les principaux composants de AWS Blu Age Runtime essentiels à l'exécution des applications AWS Blu Age, notamment :

- Gapwalk : composant essentiel du AWS Blu Age Runtime, conçu pour combler le fossé entre les applications existantes et les environnements cloud natifs modernes. Il sert de couche de compatibilité qui permet aux applications modernisées par AWS Blu Age de fonctionner efficacement sur les plateformes contemporaines.
- bluage.bin : le fichier binaire de base du AWS Blu Age Runtime. Ce fichier est essentiel au fonctionnement du moteur d'exécution.
- Toutes les bibliothèques et fichiers de support nécessaires au fonctionnement de AWS Blu Age Runtime.

aws-bluage-webapps-x.y.z.tar.gz

Cette archive, dans laquelle x.y.z suit le même schéma de version que ci-dessus, inclut les applications Web et les bibliothèques nécessaires à la gestion et au contrôle des déploiements AWS Blu Age :

- Fichier WAR BAC (console Blusam), utilisé pour surveiller la base de données Blusam.
- Fichier WAR JAC (console JICS), utilisé pour surveiller la base de données JICS.
- Bibliothèques de soutien nécessaires.

Fichiers supplémentaires

- Des fichiers Checksum qui vous permettent de vérifier l'intégrité des deux archives Blu Age conformément à la convention de dénomination :
  - Pour Runtime : `aws-bluage-runtime-x.y.z.tar.gz.checksumSHA256`
  - Pour les applications Web : `aws-bluage-webapps-x.y.z.tar.gz.checksumSHA256`

- Les fichiers de rapport CVE (pour les versions finales uniquement) répertorient les éléments présents dans CVEs cette version et suivent la convention de dénomination :
  - Pour Runtime : `Bluage-Runtime-x.y.z-CVEs.txt`
  - Pour les applications Web : `Bluage-Webapps-x.y.z-CVEs.txt`

Pour en savoir plus sur la manière dont les vulnérabilités de sécurité sont corrigées, consultez la présentation de la version d'AWS Mainframe Modernization Refactor with AWS Blu Age.

> **Note**
>
> Bien que nous nous efforcions de lancer nos produits sans CVEs, de nouveaux produits CVEs peuvent apparaître ultérieurement. Le fichier de rapport CVE est régulièrement mis à jour pour refléter l'état le plus récent.

Artefacts du développeur : AWS Blu Age Runtime

Accès et stockage des artefacts

Les artefacts du AWS Blu Age Developer Runtime sont stockés dans des compartiments S3 dédiés. Ce runtime inclut à la fois les versions préliminaires Release et Alpha. L'accès à ces artefacts est géré via les demandes de la boîte à outils AWS Blu Age. Une fois votre demande traitée et approuvée, vous aurez accès au bucket approprié à partir de celui Compte AWS indiqué dans votre demande.

Bucket Runtime pour développeurs

Le compartiment principal du Developer Runtime est le suivant : `s3://toolbox-dev-runtime`

Pour plus d'informations sur les demandes d'accès et sur la compréhension de la structure du bucket, consultez la documentation AWS Blu Age Runtimes dédiée au développement et aux applications spéciales.

Contenu de l'Artifact

Les artefacts d'exécution des développeurs incluent généralement :

gapwalk-x.y.z-dev.tar.gz

Cette archive contient la version de développement du composant Gapwalk, qui est un élément crucial du AWS Blu Age Runtime. Il est conçu pour relier les applications existantes aux environnements cloud natifs modernes.

gapwalk-runtime-x.y.z-javadoc.zip

Ce fichier zip contient la JavaDoc documentation du runtime Gapwalk. JavaDocfournit une documentation détaillée sur les API, particulièrement utile pour les développeurs travaillant à l'intégration ou à l'extension du runtime Gapwalk.

gapwalk-webapps-x.y.z-javadoc.zip

Tout comme le runtime JavaDoc, ce fichier zip contient la JavaDoc documentation spécifique aux applications Web Gapwalk. Cette documentation est essentielle pour les développeurs qui travaillent avec ou personnalisent les composants Web du système Gapwalk.

Déployez AWS Blu Age Runtime sur Amazon EC2

Pour savoir comment configurer AWS Blu Age Runtime (non géré) sur Amazon EC2, comment mettre à jour la version d'exécution, comment surveiller votre déploiement à l'aide des CloudWatch alarmes Amazon et comment ajouter des dépendances sous licence, consultez les rubriques de cette section. Ces instructions s'appliquent lorsque vous créez des EC2 instances Amazon ainsi que lorsque vous utilisez Amazon ECS sur Amazon EC2 ou Amazon EKS sur Amazon EC2.

Rubriques

- Configurer AWS Blu Age Runtime (non géré) sur Amazon EC2
- Mettez à niveau le AWS Blu Age Runtime sur Amazon EC2
- Configurer les CloudWatch alarmes Amazon AWS Blu Age Runtime (sur Amazon EC2)
- Configurer les dépendances sous licence dans AWS Blu Age Runtime sur Amazon EC2

Configurer AWS Blu Age Runtime (non géré) sur Amazon EC2

Cette rubrique explique comment configurer et déployer l' PlanetsDemo exemple d'application à l'aide de AWS Blu Age Runtime (non géré) sur Amazon EC2.

Rubriques

- Prérequis

- Configuration
- Testez l'application déployée

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- Configurez le AWS CLI en suivant les étapes décrites dans Configuration de l'AWS CLI.
- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "Intégration à AWS Blue Age Runtime ".
- Créez une EC2 instance Amazon à l'aide de l'un des types d'instances pris en charge. Pour plus d'informations, consultez Commencer avec les instances Amazon EC2 Linux.
- Assurez-vous de pouvoir vous connecter à l' EC2 instance Amazon avec succès, par exemple à l'aide de SSM.

> **Note**
>
> Tout au long de ce guide, le chemin d'installation de Tomcat est supposé être`/m2-anywhere/tomcat-gapwalk/velocity`. Assurez-vous d'utiliser ce chemin lorsque vous suivez les instructions ci-dessous ou adaptez les instructions suivantes au chemin de votre choix.

- Téléchargez et extrayez AWS Blu Age Runtime (sur Amazon EC2). Copiez le contenu du répertoire Velocity dans`/m2-anywhere/tomcat-gapwalk/velocity`. Assurez-vous de placer le `bluage.bin` fichier exactement à l'emplacement spécifié par la variable d'environnement CATALINA_HOME décrite sous CATALINA_HOME et CATALINA_BASE dans la documentation d'Apache Tomcat. Pour obtenir des instructions sur la façon de récupérer les artefacts de AWS Blu Age Runtime, notamment des informations sur le stockage, l'accès et le contenu, consultezthe section called "AWS Artefacts de Blu Age Runtime".
- Téléchargez l'archive de PlanetsDemo l'application.
- Décompressez l'archive et chargez l'application dans le compartiment Amazon S3 de votre choix.
- Créez une base de données Amazon Aurora PostgreSQL pour JICS. Le AWS Blu Age Runtime exécutera automatiquement le `PlanetsDemo-v1/jics/sql/initJics.sql` script lors du premier démarrage. Pour plus d'informations sur la création d'une base de données Amazon Aurora PostgreSQL, consultez Création et connexion à un cluster de base de données Aurora PostgreSQL.

Configuration

Pour configurer l' PlanetsDemo exemple d'application, procédez comme suit.

1. Connectez-vous à votre EC2 instance Amazon et accédez au dossier situé sous votre `conf` dossier d'installation d'Apache Tomcat 10. Ouvrez le `catalina.properties` fichier pour le modifier et remplacez la ligne `common.loader` commençant par la ligne suivante.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Accédez au dossier `/m2-anywhere/tomcat-gapwalk/velocity /webapps/webapps`.
3. Copiez les PlanetsDemo fichiers binaires disponibles `PlanetsDemo-v1/webapps/` dans le dossier depuis le compartiment Amazon S3 à l'aide de la commande suivante.

```
aws s3 cp s3://path-to-demo-app-webapps/ . --recursive
```

> **Note**
>
> `path-to-demo-app-webapps`Remplacez-le par l'URI Amazon S3 correct pour le compartiment dans lequel vous avez précédemment décompressé l' PlanetsDemo archive.

4. Copiez le contenu du `PlanetsDemo-v1/config/` dossier dans`/m2-anywhere/tomcat-gapwalk/velocity /config/`.
5. Fournissez les informations de connexion pour la base de données que vous avez créée dans le cadre des prérequis dans l'extrait de code suivant du fichier. `application-main.yml` Pour plus d'informations, voir Création et connexion à un cluster de base de données Aurora PostgreSQL.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

6. Démarrez votre serveur Apache Tomcat et vérifiez les journaux.

```
/m2-anywhere/tomcat-gapwalk/velocity/startup.sh

tail -f /m2-anywhere/tomcat-gapwalk/velocity/logs/catalina.log
```

   Si vous trouvez des codes d'erreur commençant par un C suivi d'un chiffre, par exemple CXXXX, notez les messages d'erreur. Par exemple, le code d'erreur C5102 est une erreur courante indiquant une configuration d'infrastructure incorrecte.

Testez l'application déployée

Pour un exemple de test de l' PlanetsDemo application, consultezthe section called "Testez l' PlanetsDemo application".

Mettez à niveau le AWS Blu Age Runtime sur Amazon EC2

Ce guide explique comment mettre à niveau le AWS Blu Age Runtime sur Amazon EC2.

Rubriques

- Prérequis
- Mettre à niveau le AWS Blu Age Runtime dans l' EC2instance Amazon
- Mettre à niveau le AWS Blu Age Runtime dans un conteneur

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- Pour vérifier s'il existe des instructions spécifiques pour votre version, consultezthe section called "Mise à niveau de AWS Blue Age".
- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "Intégration à AWS Blue Age Runtime ".
- Assurez-vous que vous disposez d'une EC2 instance Amazon contenant la dernière version de AWS Blu Age Runtime. Pour plus d'informations, consultez Commencer avec les instances Amazon EC2 Linux.
- Assurez-vous de pouvoir vous connecter à l' EC2 instance Amazon avec succès, par exemple à l'aide de SSM.

- Téléchargez la version du AWS Blu Age Runtime vers laquelle vous souhaitez effectuer la mise à niveau. Pour plus d'informations, voir the section called " Configurer AWS Blu Age Runtime (non géré)" Le framework se compose de deux fichiers binaires : `aws-bluage-runtime-x.x.x.x.tar.gz` et`aws-bluage-webapps-x.x.x.x.tar.gz`.

Mettre à niveau le AWS Blu Age Runtime dans l' EC2instance Amazon

Procédez comme suit pour mettre à niveau le AWS Blu Age Runtime.

1. Connectez-vous à votre EC2 instance Amazon et remplacez l'utilisateur par su en exécutant la commande suivante.

   ```
   sudo su
   ```

   Vous devez disposer du privilège de superutilisateur pour exécuter des commandes dans ce didacticiel.
2. Créez deux dossiers, un pour chaque fichier binaire.
3. Donnez à chaque dossier le même nom que le fichier binaire.
4. Copiez chaque fichier binaire dans le dossier correspondant.

   > **Warning**
   >
   > L'extraction de chaque binaire produit un dossier portant le même nom. Par conséquent, si vous extrayez les deux fichiers binaires au même endroit l'un après l'autre, vous remplacerez le contenu.

5. Pour extraire les fichiers binaires, utilisez les commandes suivantes. Exécutez les commandes dans chaque dossier.

   ```
   tar xvf aws-bluage-runtime-x.x.x.x.tar.gz
   tar xvf aws-bluage-webapps-x.x.x.x.tar.gz
   ```

6. Arrêtez les services Apache Tomcat à l'aide des commandes suivantes.

   ```
   systemctl stop tomcat.service
   systemctl stop tomcat-webapps.service
   ```

7. Remplacez le contenu de <your-tomcat-path>/shared/ par le contenu de aws-bluage-runtime-x.x.x.x/velocity/shared/.
8. Remplacez <your-tomcat-path>/webapps/gapwalk-application.war par aws-bluage-runtime-x.x.x.x/velocity/webapps/gapwalk-application.war.
9. Remplacez les fichiers war dans <your-tomcat-path>/webapps/, à savoir bac.war et jac.war, par les mêmes fichiers provenant de aws-bluage-webapps-x.x.x.x/velocity/webapps/.
10. Démarrez les services Apache Tomcat en exécutant les commandes suivantes.

```
systemctl start tomcat.service
systemctl start tomcat-webapps.service
```

11. Consultez les journaux.

Pour vérifier l'état de l'application déployée, exécutez les commandes suivantes.

```
curl http://localhost:8080/gapwalk-application/
```

Le message suivant doit apparaître.

```
Jics application is running
```

```
curl http://localhost:8181/jac/api/services/rest/jicsservice/
```

Le message suivant doit apparaître.

```
Jics application is running
```

```
curl http://localhost:8181/bac/api/services/rest/bluesamserver/serverIsUp
```

La réponse doit être vide.

Le moteur d'exécution de AWS Blu Age a été correctement mis à niveau.

Mettre à niveau le AWS Blu Age Runtime dans un conteneur

Procédez comme suit pour mettre à niveau le AWS Blu Age Runtime.

1. Reconstruisez votre image Docker avec la version de AWS Blu Age Runtime souhaitée. Pour obtenir des instructions, consultez the section called "Configurer AWS Blu Age Runtime (non géré) sur Amazon EC2".
2. Transférez votre image Docker dans votre référentiel Amazon ECR.
3. Arrêtez et redémarrez votre service Amazon ECS ou Amazon EKS.
4. Consultez les journaux.

Le AWS Blu Age Runtime a été correctement mis à niveau.

Configurer les CloudWatch alarmes Amazon AWS Blu Age Runtime (sur Amazon EC2)

Vous pouvez CloudWatch configurer la réception du journal de votre application et ajouter une alarme pour vous avertir d'éventuelles erreurs. Cela vous permet d'avoir des notifications plus visibles chaque fois que vos applications déployées rencontrent des exceptions. Les sections suivantes vous aident à comprendre et à découvrir la configuration de la CloudWatch journalisation et de la configuration des alarmes.

Déploiement de la CloudWatch journalisation

Par défaut, le AWS Blu Age Runtime contient un fichier journal nommé`logback-cloudwatch.yml`. Ce fichier est référencé dans le `application-main.yml` fichier, mais cette référence est commentée.

```
# logging:
#  config: classpath:logback-cloudwatch.xml
```

Les deux fichiers se trouvent dans le dossier de configuration, et en décommentant les lignes ci-dessus, la fonctionnalité peut être activée. CloudWatch la journalisation peut être configurée, comme expliqué dans les sections suivantes.

Configuration de la CloudWatch journalisation

Le contenu `logback-cloudwatch.xml` du fichier par défaut est le suivant.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration>
<configuration>

    <appender name="console" class="ch.qos.logback.core.ConsoleAppender">
```

```
        <encoder>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </encoder>
    </appender>

    <appender name="cloudwatch"
 class="com.netfective.bluage.runtime.cloudwatchlogger.CloudWatchAppender">
        <logGroup>BluAgeRuntimeOnEC2-Logs</logGroup>
        <logStream>%date{yyyy-MM-dd,UTC}.%instanceId.%uuid</logStream>
        <layout>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </layout>
        <appender-ref ref="console" />
    </appender>

    <root level="INFO">
        <appender-ref ref="cloudwatch" />
    </root>
</configuration>
```

Tout ce qui se trouve en dehors de l'<appender name="cloudwatch"/>élément est une configuration de logback standard. Ce fichier contient deux annexes : un annexe de console pour envoyer les journaux à la console et un CloudWatch annexe pour envoyer les journaux. CloudWatch

L'`level`attribut de l'`root`élément indique le niveau de journalisation de l'ensemble de l'application.

Les valeurs requises à l'intérieur de la balise <appender name="cloudwatch"/> sont les suivantes :

- <logGroup/>:Définit le nom du groupe de journaux dans CloudWatch. Si la valeur n'est pas spécifiée, elle est définie par défaut sur`BluAgeRuntimeOnEC2-Logs`. Si le groupe de journaux n'existe pas, il sera créé automatiquement. Ce comportement peut être modifié par le biais de la configuration, décrite ci-dessous.
- <logStream/>: définit le nom du LogStream (à l'intérieur du groupe de journaux) dans. CloudWatch

Valeurs facultatives :

- <region/>: remplace la région dans laquelle le flux de log sera écrit. Par défaut, les journaux sont envoyés dans la même région que l' EC2 instance.
- <layout/>: le modèle que les messages du journal utiliseront.

- <maxbatchsize/>: nombre maximal de messages de journal auxquels envoyer CloudWatch par opération.
- <maxbatchtimemillis/>: durée en millisecondes pendant laquelle les CloudWatch journaux peuvent être écrits.
- <maxqueuewaittimemillis/>: durée en millisecondes nécessaire pour essayer d'insérer des demandes dans la file d'attente interne du journal.
- <internalqueuesize/>: taille maximale de la file d'attente interne.
- <createlogdests/>: créez un groupe de journaux et un flux de journaux s'ils n'existent pas.
- <initialwaittimemillis/>: durée pendant laquelle vous souhaitez que le thread soit mis en veille au démarrage. Cette attente initiale permet une accumulation initiale de journaux.
- <maxeventmessagesize/>: taille maximale d'un événement de journal. Les journaux dont la taille dépasse cette taille ne seront pas envoyés.
- <truncateeventmessages/>: tronque les messages trop longs.
- <printrejectedevents/>: Activez l'annexe d'urgence.

CloudWatch configuration

Pour que la configuration ci-dessus envoie correctement les journaux vers CloudWatch, mettez à jour votre rôle de profil d'instance Amazon EC2 IAM afin de lui accorder des autorisations supplémentaires pour le groupe de journaux `BluAgeRuntimeOnEC2-Logs` et ses flux de journaux :

- `logs:CreateLogStream`
- `logs:DescribeLogStreams`
- `logs:CreateLogGroup`
- `logs:PutLogEvents`
- `logs:DescribeLogGroups`

Configuration de l'alarme

Grâce aux CloudWatch journaux, vous pouvez ensuite configurer différentes métriques et alarmes, en fonction de votre application et de vos besoins. Plus précisément, vous pouvez configurer des alarmes proactives pour les alertes d'utilisation, afin d'être averti en cas d'erreur susceptible de mettre votre application en période de grâce (et, en fin de compte, de l'empêcher de fonctionner). Pour ce faire, vous pouvez ajouter une métrique concernant la chaîne « Error C5001 » dans les

journaux, qui met en évidence les erreurs de connexion au système de contrôle AWS Blu Age. Vous pouvez ensuite définir une alarme qui réagit à cette métrique.

Configurer les dépendances sous licence dans AWS Blu Age Runtime sur Amazon EC2

Ce guide explique comment configurer des dépendances sous licence supplémentaires que vous pouvez utiliser avec AWS Blu Age Runtime sur Amazon EC2.

Rubriques

- Prérequis
- Présentation
- Configuration des dépendances pour les applications Web JAC et BAC

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called “AWS Prérequis pour Blu Age Runtime”Complet etthe section called “Intégration à AWS Blue Age Runtime ”.
- Assurez-vous que vous disposez d'une EC2 instance Amazon contenant la dernière version de AWS Blu Age Runtime (sur Amazon EC2). Pour plus d'informations, consultez Commencer avec les instances Amazon EC2 Linux.
- Assurez-vous de pouvoir vous connecter à l' EC2 instance Amazon avec succès, par exemple à l'aide de SSM.
- Obtenez les dépendances suivantes à partir de leurs sources.

Oracle Database

Fournissez un pilote de base de données Oracle. Nous avons testé la fonctionnalité AWS Blu Age Runtime (sur Amazon EC2) avec la version ojdbc11-23.3.0.23.09.jar, mais une version plus récente est peut-être compatible.

Connexion IBM MQ

Fournissez un client IBM MQ. Nous avons testé la fonctionnalité AWS Blu Age Runtime (sur Amazon EC2) avec la version com.ibm.mq.jakarta.client-9.3.4.1.jar, mais une version plus récente est peut-être compatible.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- bcprov-jdk15to18-1.76.jar
- bcpkix-jdk15to18-1.76.jar
- bcutil-jdk15to18-1.76.jar

Fichiers d'imprimante DDS

Fournissez la bibliothèque de rapports Jasper (https://community.jaspersoft). com/download-jaspersoft/community-édition). Nous avons testé la fonctionnalité AWS Blu Age Runtime (sur Amazon EC2) avec jasperreports-6.16.0.jar, mais une version plus récente pourrait être compatible.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- castor-core-1.4.1.jar
- castor-xml-1.4.1.jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7.js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar
- commons-beanutils-1.9.4.jar
- commons-collections-3.2.jar

Présentation

Pour installer les dépendances, procédez comme suit.

1. Connectez-vous à votre EC2 instance Amazon et remplacez l'utilisateur par su en exécutant la commande suivante.

   ```
   sudo su
   ```

   Vous devez disposer du privilège de superutilisateur pour exécuter des commandes dans ce didacticiel.

2. Accédez au dossier <your-tomcat-path>/extra/.

```
cd <your-tomcat-path>/extra/
```

3. Copiez l'une des dépendances ci-dessus selon les besoins dans ce dossier.
4. Arrêtez et démarrez le tomcat.service en exécutant les commandes suivantes.

```
systemctl stop tomcat.service
```

```
systemctl start tomcat.service
```

5. Vérifiez l'état du service pour vous assurer qu'il fonctionne.

```
systemctl status tomcat.service
```

6. Vérifiez les journaux.

Configuration des dépendances pour les applications Web JAC et BAC

1. Si votre base de données JICS est hébergée sur Oracle, vous devez fournir le pilote de base de données Oracle dans<your-tomcat-path>/extra.
2. Créez le dossier s'il n'est pas déjà présent.
3. Arrêtez et redémarrez votre serveur Apache Tomcat.
4. Vérifiez les journaux.

Déployez AWS Blu Age Runtime sur des conteneurs sur Amazon ECS et Amazon EKS

Vous pouvez utiliser les rubriques de cette section pour savoir comment configurer AWS Blu Age Runtime sur des conteneurs afin de le déployer sur Amazon ECS (géré par Amazon EC2 ou AWS Fargate) et Amazon EKS géré par Amazon EC2, comment mettre à jour la version d'exécution, comment surveiller votre déploiement à l'aide des CloudWatch alarmes Amazon et comment ajouter des dépendances sous licence.

> **Note**
>
> Ceci n'est pas compatible avec Amazon EKS géré par AWS Fargate.

Rubriques

- Configurer AWS Blu Age Runtime sur un conteneur
- Mettre à niveau le AWS Blu Age Runtime sur un conteneur
- Configurer les CloudWatch alarmes Amazon pour AWS Blu Age Runtime sur un conteneur
- Configurer les dépendances sous licence dans AWS Blu Age Runtime sur un conteneur

Configurer AWS Blu Age Runtime sur un conteneur

Cette rubrique explique comment configurer et déployer l' PlanetsDemo exemple d'application à l'aide de AWS Blu Age Runtime sur un conteneur docker.

AWS Blu Age Runtime on container est disponible pour Amazon ECS géré par Amazon EC2, Amazon ECS géré par AWS Fargate et Amazon EKS géré par Amazon EC2. Il n'est pas compatible avec Amazon EKS géré par AWS Fargate.

Rubriques

- Prérequis
- Configuration
- Testez l'application déployée

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- Configurez le AWS CLI en suivant les étapes décrites dans Configuration de l'AWS CLI.
- the section called “AWS Prérequis pour Blu Age Runtime”Complet etthe section called “Intégration à AWS Blue Age Runtime ”.
- Téléchargez les fichiers binaires de AWS Blu Age Runtime. Pour obtenir des instructions, consultez the section called “Intégration à AWS Blue Age Runtime ”.
- Téléchargez les fichiers binaires d'Apache Tomcat 10.
- Téléchargez l'archive de PlanetsDemo l'application.

- Créez une base de données Amazon Aurora PostgreSQL pour JICS et exécutez `PlanetsDemo-v1/jics/sql/initJics.sql` la requête dessus. Pour plus d'informations sur la création d'une base de données Amazon Aurora PostgreSQL, consultez Création et connexion à un cluster de bases de données Aurora PostgreSQL.

Configuration

Pour configurer l' PlanetsDemo exemple d'application, procédez comme suit.

1. Après avoir téléchargé les fichiers binaires d'Apache Tomcat, extrayez le contenu et accédez au `conf` dossier. Ouvrez le `catalina.properties` fichier pour le modifier et remplacez la ligne `common.loader` commençant par la ligne suivante.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Compressez le dossier Apache Tomcat en utilisant la commande tar pour créer une archive « tar.gz ».
3. Préparez un Dockerfile pour créer votre image personnalisée en fonction des fichiers binaires d'exécution et des fichiers binaires du serveur Apache Tomcat fournis. Consultez l'exemple de Dockerfile suivant. L'objectif est d'installer Apache Tomcat 10, suivi d'AWS Blu Age Runtime (pour Amazon ECS géré par AWS Fargate) extrait à la racine du répertoire d'installation d'Apache Tomcat 10, puis d'installer l'exemple d'application modernisée nommé. PlanetsDemo

> **Note**
>
> Le contenu des scripts install-gapwalk.sh et install-app.sh, utilisés dans cet exemple de Dockerfile, est répertorié après le Dockerfile.

```
FROM --platform=linux/x86_64 amazonlinux:2

RUN mkdir -p /workdir/apps
WORKDIR /workdir
COPY install-gapwalk.sh .
COPY install-app.sh .
RUN chmod +x install-gapwalk.sh
```

```
RUN chmod +x install-app.sh

# Install Java and AWS CLI v2-y
RUN yum install sudo java-17-amazon-corretto unzip tar -y
RUN sudo yum remove awscli -y
RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
RUN sudo unzip awscliv2.zip
RUN sudo ./aws/install

# Installation dir
RUN mkdir -p /usr/local/velocity/installation/gapwalk
# Copy PlanetsDemo archive to a dedicated apps dir
COPY PlanetsDemo-v1.zip /workdir/apps/

# Copy resources (tomcat, blu age runtime) to installation dir
COPY tomcat.tar.gz /usr/local/velocity/installation/tomcat.tar.gz
COPY aws-bluage-runtime-4.x.x.tar.gz /usr/local/velocity/installation/gapwalk/
gapwalk.tar.gz

# run relevant installation scripts
RUN ./install-gapwalk.sh
RUN ./install-app.sh

EXPOSE 8080
EXPOSE 8081
# ...

WORKDIR /bluage/tomcat.gapwalk/velocity
# Run Command to start Tomcat server
CMD ["sh", "-c", "sudo bin/catalina.sh run"]
```

Voici le contenu de `install-gapwalk.sh`.

```
# Vars
TEMP_DIR=/bluage-on-fargate/tomcat.gapwalk/temp

# Install
echo "Installing Gapwalk and Tomcat"
sudo rm -rf /bluage-on-fargate
mkdir -p ${TEMP_DIR}
# Copy Blu Age runtime and tomcat archives to temporary extraction dir
```

```
sudo cp /usr/local/velocity/installation/gapwalk/gapwalk.tar.gz ${TEMP_DIR}
sudo cp /usr/local/velocity/installation/tomcat.tar.gz ${TEMP_DIR}
# Create velocity dir
mkdir -p /bluage/tomcat.gapwalk/velocity
# Extract tomcat files
tar -xvf ${TEMP_DIR}/tomcat.tar.gz -C ${TEMP_DIR}
# Copy all tomcat files to velocity dir
cp -fr ${TEMP_DIR}/apache-tomcat-10.x.x/* /bluage/tomcat.gapwalk/velocity
# Remove default webapps of Tomcat
rm -f /bluage-on-fargate/tomcat.gapwalk/velocity/webapps/*
# Extract Blu Age runtime at velocity dir
tar -xvf ${TEMP_DIR}/gapwalk.tar.gz -C /bluage/tomcat.gapwalk
# Remove temporary extraction dir
sudo rm -rf ${TEMP_DIR}
```

Voici le contenu de`install-app.sh`.

```
#!/bin/sh

APP_DIR=/workdir/apps
TOMCAT_GAPWALK_DIR=/bluage-on-fargate/tomcat.gapwalk

unzip ${APP_DIR}/PlanetsDemo-v1.zip -d ${APP_DIR}
cp -r ${APP_DIR}/webapps/* ${TOMCAT_GAPWALK_DIR}/velocity/webapps/
cp -r ${APP_DIR}/config/* ${TOMCAT_GAPWALK_DIR}/velocity/config/
```

4. Fournissez les informations de connexion pour la base de données que vous avez créée dans le cadre des prérequis dans l'extrait suivant du `application-main.yml` fichier, qui se trouve dans le dossier. `{TOMCAT_GAPWALK_DIR}/config` Pour plus d'informations, voir Création et connexion à un cluster de base de données Aurora PostgreSQL.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

5. Créez et transférez l'image dans votre référentiel Amazon ECR. Pour obtenir des instructions, consultez la section Envoyer une image Docker dans le guide de l'utilisateur d'Amazon Elastic Container Registry. Ensuite, en fonction de votre situation, créez un pod Amazon EKS ou une

définition de tâche Amazon ECS à l'aide de votre image Amazon ECR, puis déployez-la sur votre cluster. Pour en savoir plus sur leur création, consultez les sections Création d'une définition de tâche à l'aide de la console dans le guide du développeur Amazon Elastic Container Service (Amazon ECS) et Déployer un exemple d'application dans le guide de l'utilisateur Amazon EKS.

6. Plus précisément, pour Amazon ECS géré au AWS Fargate cas par cas, lors de la création de la définition de tâche, utilisez le rôle IAM que vous avez créé dans le cadre de la configuration initiale de l'infrastructure. Ensuite, lors de la création du service, développez la section Mise en réseau et configurez le VPC, les sous-réseaux et le groupe de sécurité que vous avez créés dans le cadre de la configuration initiale de l'infrastructure. Voir les exigences de configuration de l'infrastructure pour AWS Blu Age Runtime (non géré).

Testez l'application déployée

Pour un exemple de test de l' PlanetsDemo application, consultezthe section called “Testez l' PlanetsDemo application”.

Mettre à niveau le AWS Blu Age Runtime sur un conteneur

Ce guide explique comment mettre à niveau le AWS Blu Age Runtime sur un conteneur. Pour ce faire, vous devez d'abord remplir certaines conditions préalables, puis utiliser l'image Docker pour mettre à niveau le AWS Blu Age Runtime.

Rubriques

- Prérequis
- Améliorez le AWS Blu Age Runtime

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called “AWS Prérequis pour Blu Age Runtime”Complet etthe section called “Intégration à AWS Blue Age Runtime ”.
- Téléchargez la version du AWS Blu Age Runtime vers laquelle vous souhaitez effectuer la mise à niveau. Pour de plus amples informations, veuillez consulter the section called “Intégration à AWS Blue Age Runtime ”. Le framework se compose de deux fichiers binaires : `aws-bluage-runtime-x.x.x.x.tar.gz` et`aws-bluage-webapps-x.x.x.x.tar.gz`.

Améliorez le AWS Blu Age Runtime

Procédez comme suit pour mettre à niveau le AWS Blu Age Runtime.

1. Reconstruisez votre image Docker avec la version de AWS Blu Age Runtime souhaitée. Pour obtenir des instructions, consultez the section called "Configurer AWS Blu Age Runtime sur un conteneur".
2. Transférez votre image Docker dans votre référentiel Amazon ECR.
3. Arrêtez et redémarrez votre service Amazon ECS ou Amazon EKS.
4. Vérifiez les journaux.

Le AWS Blu Age Runtime est correctement mis à niveau.

Configurer les CloudWatch alarmes Amazon pour AWS Blu Age Runtime sur un conteneur

Vous pouvez configurer CloudWatch des notifications plus visibles chaque fois que vos applications déployées rencontrent des exceptions. Cela vous permet de surveiller le journal de votre application redirigé vers CloudWatch et d'ajouter une alarme pour vous avertir d'éventuelles erreurs.

Configuration de l'alarme

Avec CloudWatch les journaux, vous pouvez configurer autant de métriques et d'alarmes que vous le souhaitez, en fonction de votre application et de vos besoins.

Plus précisément, vous pouvez configurer des alarmes proactives pour les alertes d'utilisation directement lors de la création de votre cluster, afin d'être averti en cas d'erreur. Pour mettre en évidence les erreurs de connexion au système de contrôle AWS Blu Age, ajoutez une métrique concernant la chaîne « Error C » dans les journaux. Vous pouvez ensuite définir une alarme qui réagit à cette métrique.

Configurer les dépendances sous licence dans AWS Blu Age Runtime sur un conteneur

Cette rubrique décrit comment configurer des dépendances sous licence supplémentaires que vous pouvez utiliser avec AWS Blu Age Runtime sur un conteneur.

Rubriques

- Prérequis

- Présentation

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "Intégration à AWS Blue Age Runtime ".
- Obtenez les dépendances suivantes à partir de leur source.

Oracle Database

Fournissez un pilote de base de données Oracle. Par exemple, ojdbc11-23.3.0.23.09.jar.

Connexion IBM MQ

Fournissez un client IBM MQ. Par exemple, com.ibm.mq.jakarta.client-9.3.4.1.jar.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- bcprov-jdk15to18-1.76.jar
- bcpkix-jdk15to18-1.76.jar
- bcutil-jdk15to18-1.76.jar

Fichiers d'imprimante DDS

Fournissez la bibliothèque de rapports Jasper (https://community.jaspersoft). com/download-jaspersoft/community-édition). Par exemple, jasperreports-6.16.0.jar, mais une version plus récente peut être compatible.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- castor-core-1.4.1.jar
- castor-xml-1.4.1.jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7.js8.jar
- javax.inject-1.jar

- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar
- commons-beanutils-1.9.4.jar
- commons-collections-3.2.jar

Présentation

Pour installer les dépendances, procédez comme suit.

1. Copiez l'une des dépendances ci-dessus selon les besoins dans votre dossier de génération d'image Docker.
2. Si votre base de données JICS est hébergée sur Oracle, fournissez le pilote de base de données Oracle dans*your-tomcat-path*`/extra`.
3. Sur votre Dockerfile, copiez ces dépendances dans. *your-tomcat-path*`/extra`
4. Créez votre image Docker, puis envoyez-la vers Amazon ECR.
5. Arrêtez et redémarrez votre service Amazon ECS ou Amazon EKS.
6. Consultez les journaux.

Testez l' PlanetsDemo application

Pour vérifier l'état de l' PlanetsDemo application déployée, exécutez les commandes suivantes après le remplacement `load-balancer-DNS-namelistener-port`, `web-binary-name` avec les valeurs correctes pour votre configuration.

```
curl http://load-balancer-DNS-name:listener-port/gapwalk-application/
```

Si l'application est en cours d'exécution, le message de sortie suivant s'affiche :`Jics application is running`.

Exécutez ensuite la commande suivante.

```
curl http://load-balancer-DNS-name:listener-port/jac/api/services/rest/jicsservice/
```

Si l'application est en cours d'exécution, le message de sortie suivant s'affiche :`Jics application is running`.

```
Jics application is running
```

Si vous avez configuré Blusam, vous pouvez vous attendre à une réponse vide lorsque vous exécutez la commande suivante.

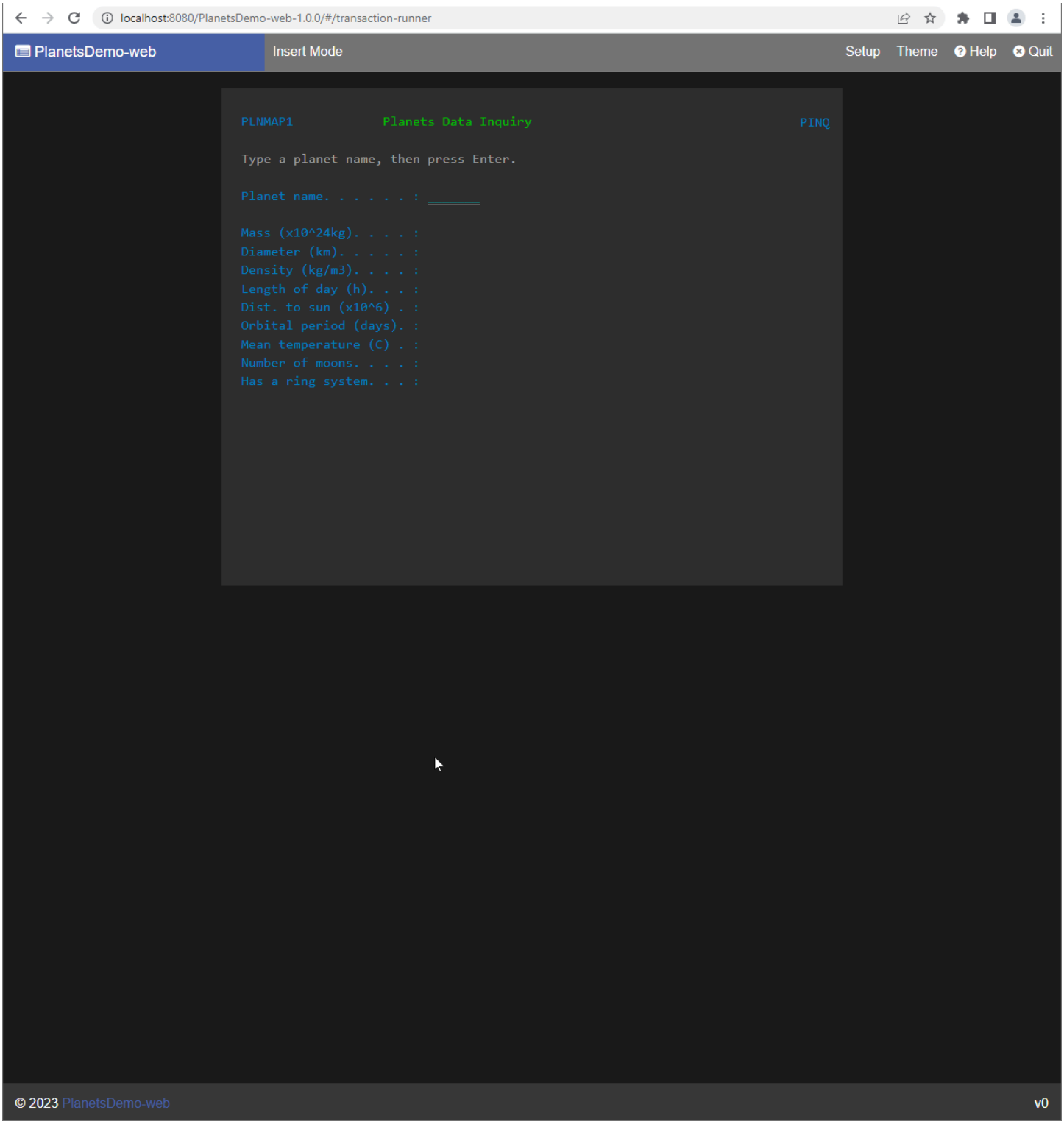
```
curl http://load-balancer-DNS-name:listener-port/bac/api/services/rest/bluesamserver/
serverIsUp
```

Notez le nom du binaire Web (PlanetsDemo-web-1.0.0, s'il est inchangé). Pour accéder à l' PlanetsDemo application, utilisez une URL au format suivant.

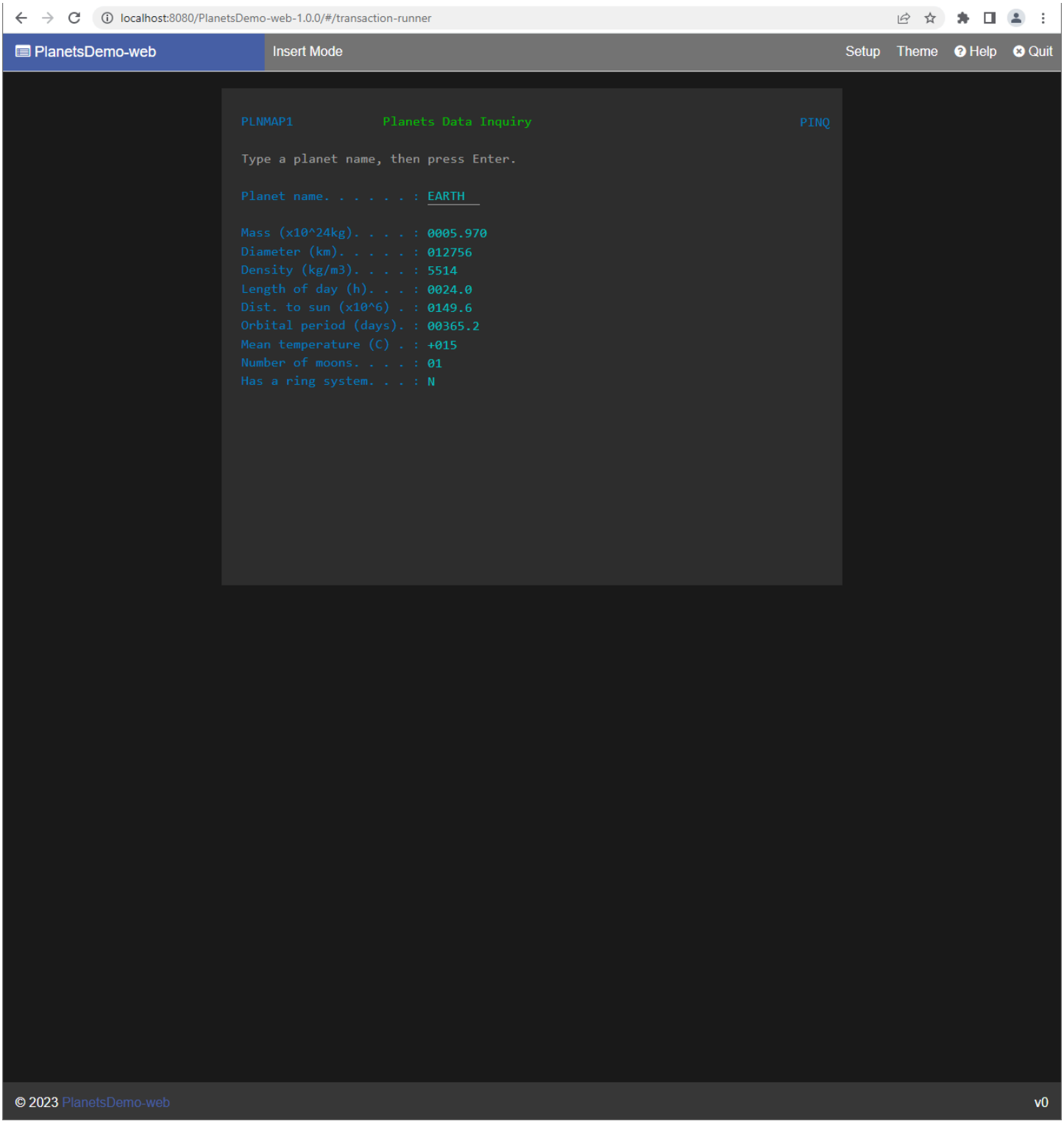
```
https://load-balancer-DNS-name:listener-port/web-binary-name
```

Après le démarrage de PlanetsDemo l'application, la page d'accueil s'affiche.

Entrez PINQ dans la zone de texte, puis appuyez sur Entrée. La page de demande de données s'affiche.



Par exemple, entrez EARTH dans le champ de PlanetsDemo nom, puis appuyez sur Entrée. La page de la planète que vous avez saisie s'affiche.



AWS Blu Age Runtime est disponible dans les régions suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Californie du Nord), États-Unis Ouest (Oregon), Canada (centre), Europe (Irlande), région Europe (Londres), région Europe (Paris), Europe (Francfort), région Europe (Stockholm), région Europe (Milan), région Europe (Espagne), Amérique du Sud (São Paulo), Asie

Pacifique (Tokyo), Asie-Pacifique (Séoul), Asie-Pacifique (Osaka), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Mumbai), Afrique (Le Cap) et Israël (Tel Aviv).

Modifiez le code source avec Blu Age Developer IDE

Si vous utilisez le moteur d'exécution AWS Blu Age AWS géré, vous pouvez utiliser Blu Age Developer pour modifier le code source généré. Vous pouvez le faire si vous devez mettre à jour le code modernisé pour une raison ou une autre, ou si une partie de l'ancien code source ne peut pas être modernisée. Vous accédez à Blu Age Developer via Amazon AppStream 2.0. Cette section explique comment configurer Blu Age Developer sur AppStream 2.0. Il explique également comment utiliser Blu Age Developer pour mettre à jour le code source, à l'aide de l'exemple d'application PlanetsDemo.

Rubriques

- Tutoriel : Configuration de la AppStream version 2.0 pour AWS Blu Age Developer IDE
- Tutoriel : Utiliser AWS Blu Age Developer sur AppStream 2.0

Tutoriel : Configuration de la AppStream version 2.0 pour AWS Blu Age Developer IDE

AWS La modernisation du mainframe fournit plusieurs outils via Amazon AppStream 2.0. AppStream 2.0 est un service de streaming d'applications sécurisé et entièrement géré qui vous permet de diffuser des applications de bureau aux utilisateurs sans avoir à réécrire les applications. AppStream La version 2.0 fournit aux utilisateurs un accès instantané aux applications dont ils ont besoin avec une expérience utilisateur réactive et fluide sur l'appareil de leur choix. L'utilisation de la AppStream version 2.0 pour héberger des outils spécifiques au moteur d'exécution permet aux équipes d'application des clients d'utiliser les outils directement depuis leur navigateur Web, en interagissant avec les fichiers d'application stockés dans des compartiments ou des référentiels Amazon S3. CodeCommit

Pour plus d'informations sur la prise en charge des navigateurs dans la AppStream version 2.0, consultez la section Configuration système requise et prise en charge des fonctionnalités (navigateur Web) dans le guide d'administration Amazon AppStream 2.0. Si vous rencontrez des problèmes lors de l'utilisation de la AppStream version 2.0, consultez la section Résolution AppStream des problèmes liés aux utilisateurs de la AppStream version 2.0 dans le guide d'administration Amazon 2.0.

Ce document décrit comment configurer l'IDE AWS Blu Age Developer sur un parc AppStream 2.0.

Rubriques

- Prérequis
- Étape 1 : Créer un compartiment Amazon S3
- Étape 2 : associer une politique au compartiment S3
- Étape 3 : télécharger des fichiers dans le compartiment Amazon S3
- Étape 4 : Téléchargez les AWS CloudFormation modèles
- Étape 5 : Créez la flotte avec AWS CloudFormation
- Étape 6 : accéder à une instance
- Nettoyage des ressources

Prérequis

Pour les nouveaux utilisateurs, procédez comme suit :

1. Accédez à la console AppStream 2.0 à la https://console.aws.amazon.com/appstream2/maison.
2. Sélectionnez Get started (Mise en route).
3. Choisissez Skip.

> **Important**
>
> Amazon AppStream 2.0 utilise des rôles IAM pour gérer vos ressources AppStream 2.0 et AWS créera ces rôles lorsque vous le ferez.

Téléchargez ensuite le fichier d'archive qui contient les artefacts dont vous avez besoin pour configurer AWS Blu Age Developer IDE sous AppStream 2.0.

> **Note**
>
> Il s'agit d'un fichier volumineux. Si vous rencontrez des problèmes avec le délai d'expiration de l'opération, nous vous recommandons d'utiliser une EC2 instance Amazon pour améliorer les performances de chargement et de téléchargement. Pour plus d'informations sur le

lancement et la connexion à une EC2 instance Amazon, consultez Get started with Amazon EC2.

Étape 1 : Créer un compartiment Amazon S3

Créez un compartiment Amazon S3 Région AWS identique à la flotte AppStream 2.0 que vous allez créer. Ce compartiment contiendra les artefacts dont vous avez besoin pour terminer ce didacticiel. Pour plus d'informations sur les compartiments, consultez la section Création d'un compartiment.

Étape 2 : associer une politique au compartiment S3

Attachez la politique suivante au bucket que vous créez pour ce didacticiel. Pour plus d'informations sur l'attachement d'une politique au compartiment S3, consultez la section Ajout d'une politique de compartiment.

Assurez-vous de le `amzn-s3-demo-bucket` remplacer par le nom réel du bucket que vous créez.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowAppStream2.0ToRetrieveObjects",
        "Effect": "Allow",
        "Principal": {
            "Service": "appstream.amazonaws.com"
        },
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }]
}
```

Étape 3 : télécharger des fichiers dans le compartiment Amazon S3

Décompressez les fichiers que vous avez téléchargés dans le prérequis et chargez le `appstream` dossier dans votre compartiment. Le téléchargement de ce dossier crée la structure appropriée dans votre compartiment. Pour plus d'informations, consultez la section Chargement d'objets dans le guide de l'utilisateur Amazon S3.

Étape 4 : Téléchargez les AWS CloudFormation modèles

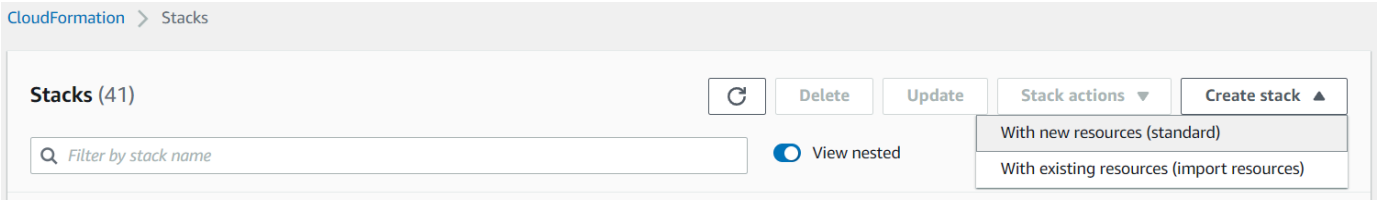
Téléchargez les AWS CloudFormation modèles suivants. Vous avez besoin de ces modèles pour créer et alimenter le parc AppStream 2.0.

- cfn-m2-.yaml appstream-elastic-fleet-linux
- cfn-m2- -linux.yaml appstream-bluage-dev-tools
- cfn-m2-.yaml appstream-bluage-shared-linux
- cfn-m2-.yaml appstream-chrome-linux
- cfn-m2-.yaml appstream-eclipse-jee-linux
- cfn-m2-.yaml appstream-pgadmin-linux

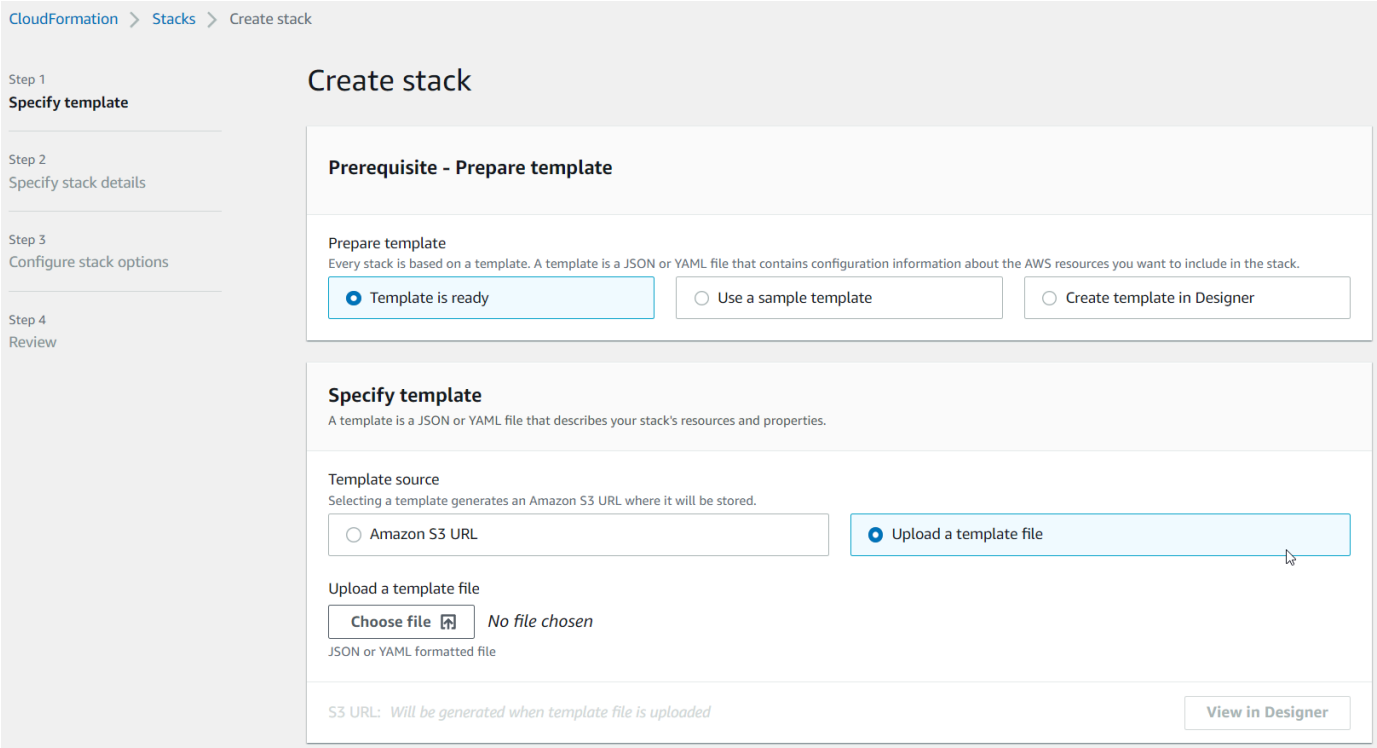
Étape 5 : Créez la flotte avec AWS CloudFormation

Au cours de cette étape, vous allez utiliser le `cfn-m2-appstream-elastic-fleet-linux.yaml` AWS CloudFormation modèle pour créer une flotte et une pile AppStream 2.0 afin d'héberger l'IDE AWS Blu Age Developer. Après avoir créé le parc et la pile, vous exécuterez les autres AWS CloudFormation modèles que vous avez téléchargés à l'étape précédente pour installer l'IDE Developer et les autres outils requis.

1. Accédez AWS CloudFormation à la console AWS de gestion, puis sélectionnez Stacks.
2. Dans Stacks, choisissez Create stack et With new Resources (standard) :



3. Dans Créer une pile, choisissez Choisir un modèle existant et Charger un fichier modèle :



4. Choisissez Choisir un fichier, puis naviguez jusqu'au fichier`cfn-m2-appstream-elastic-fleet-linux.yaml`. Choisissez Next (Suivant).
5. Dans Spécifier les détails de la pile, fournissez les informations suivantes :

   - Nom de la pile.
   - Votre groupe de sécurité par défaut et les deux sous-réseaux de ce groupe de sécurité.

   > **Note**
   >
   > Les deux sous-réseaux du groupe de sécurité doivent se trouver dans des zones de disponibilité différentes.

6. Choisissez Next (Suivant).
7. Naviguez vers le bas de la page et choisissez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM avec des noms personnalisés. .
8. Choisissez Next (Suivant).
9. Vérifiez les informations, puis choisissez Soumettre.
10. Après avoir créé le parc, créez des CloudFormation piles avec tous les autres modèles téléchargés pour terminer la configuration des applications. Veillez à effectuer la mise à jour à BucketNamechaque fois pour pointer vers le compartiment S3 approprié. Vous pouvez

le modifier BucketNamedans la CloudFormation console. Vous pouvez également modifier directement les fichiers modèles et mettre à jour la `S3Bucket` propriété.

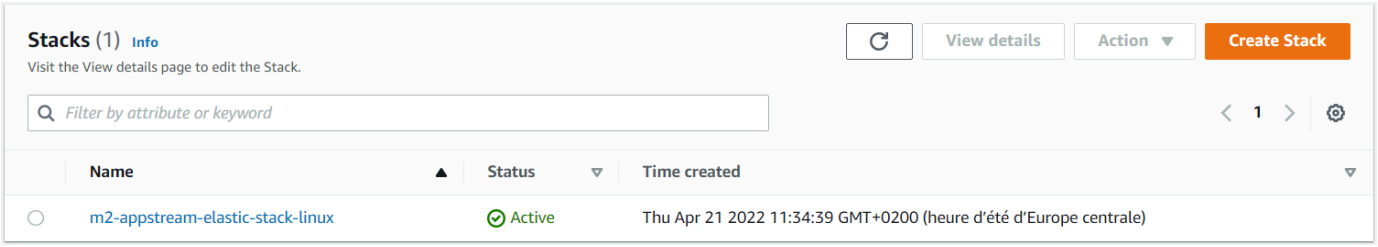
> **Note**
>
> Les modèles téléchargés s'attendent à trouver des ressources dans un compartiment S3 dont la structure de dossiers est appelée`appstream/bluage/developer-ide/`. Le compartiment doit se trouver dans le même emplacement Région AWS que le parc que vous avez créé.

> **Important**
>
> Exécutez tous les CloudFormation scripts téléchargés à l'étape 4 pour configurer correctement votre application.

Étape 6 : accéder à une instance

Après avoir créé et démarré la flotte, vous pouvez créer un lien temporaire pour accéder à la flotte via le client natif.

1. Accédez à la AppStream version 2.0 dans le AWS Management Console et choisissez la pile créée précédemment :



2. Sur la page des détails de la pile, choisissez la pile, puis sélectionnez Associer une flotte.
3. Dans l'invite, choisissez la flotte que vous avez créée et démarrée précédemment.
4. Choisissez Associer.
5. Choisissez la pile associée et dans le menu Actions, choisissez Create Streaming URL, entrez un ID utilisateur arbitraire et une date d'expiration de l'URL, puis choisissez Get URL. Vous obtenez une URL que vous pouvez utiliser pour diffuser vers un navigateur ou vers le client natif. Nous vous recommandons de diffuser sur le client natif.

Nettoyage des ressources

Pour la procédure de nettoyage de la pile et des flottes créées, voir Créer une flotte et une pile AppStream 2.0.

Lorsque vous avez supprimé les objets AppStream 2.0, vous ou l'administrateur du compte pouvez également nettoyer les compartiments S3 pour les paramètres de l'application et les dossiers personnels.

> **Note**
>
> Le dossier d'accueil d'un utilisateur donné étant unique dans toutes les flottes, vous devrez peut-être le conserver si d'autres piles AppStream 2.0 sont actives sur le même compte.

Vous ne pouvez pas utiliser la console AppStream 2.0 pour supprimer des utilisateurs. Vous devez plutôt utiliser l'API du service avec le AWS CLI. Pour plus d'informations, consultez la section Administration du groupe d'utilisateurs dans le guide d'administration Amazon AppStream 2.0.

Tutoriel : Utiliser AWS Blu Age Developer sur AppStream 2.0

Ce didacticiel vous explique comment accéder à AWS Blu Age Developer sur AppStream 2.0 et comment l'utiliser avec un exemple d'application afin de tester les fonctionnalités. Lorsque vous aurez terminé ce didacticiel, vous pourrez suivre les mêmes étapes avec vos propres applications.

Rubriques

- Étape 1 : Créer une base de données
- Étape 2 : Accès à l'environnement
- Étape 3 : configurer le moteur d'exécution
- Étape 4 : démarrer l'IDE Eclipse
- Étape 5 : configurer un projet Maven
- Étape 6 : Configuration d'un serveur Tomcat
- Étape 7 : Déployer sur Tomcat
- Étape 8 : Création de la base de données JICS
- Étape 9 : démarrer et tester l'application
- Étape 10 : Déboguer l'application
- Nettoyage des ressources

Étape 1 : Créer une base de données

Au cours de cette étape, vous utilisez Amazon RDS pour créer une base de données PostgreSQL gérée que l'application de démonstration utilise pour stocker les informations de configuration.

1. Ouvrez la console Amazon RDS.
2. Choisissez Bases de données > Créer une base de données.
3. Choisissez Standard create > PostgreSQL, conservez la version par défaut, puis choisissez Free tier.
4. Choisissez un identifiant d'instance de base de données.
5. Pour les paramètres d'identification, choisissez Gérer les informations d'identification principales dans AWS Secrets Manager. Pour plus informations, consultez Gestion des mots de passe avec Amazon RDS et AWS Secrets Manager (français non garanti) dans le Guide de l'utilisateur Amazon RDS (français non garanti).
6. Assurez-vous que le VPC est le même que celui que vous utilisez pour l'instance AppStream 2.0. Vous pouvez demander cette valeur à votre administrateur.
7. Pour le groupe de sécurité VPC, choisissez Create New.
8. Réglez l'accès public sur Oui.
9. Conservez toutes les autres valeurs par défaut. Passez en revue ces valeurs.
10. Choisissez Créer une base de données.

Pour rendre le serveur de base de données accessible depuis votre instance, sélectionnez le serveur de base de données dans Amazon RDS. Sous Connectivité et sécurité, choisissez le groupe de sécurité VPC pour le serveur de base de données. Ce groupe de sécurité a déjà été créé pour vous et doit avoir une description similaire à celle décrite dans la section Créé par la console de gestion RDS. Choisissez Action > Modifier les règles entrantes, choisissez Ajouter une règle et créez une règle de type PostgreSQL. Pour la source des règles, utilisez le groupe de sécurité par défaut. Vous pouvez commencer à saisir le nom de la source dans le champ Source, puis accepter l'ID suggéré. Enfin, choisissez Enregistrer les règles.

Étape 2 : Accès à l'environnement

Au cours de cette étape, vous accédez à l'environnement de développement AWS Blu Age AppStream 2.0.

1. Contactez votre administrateur pour savoir comment accéder correctement à votre instance AppStream 2.0. Pour obtenir des informations générales sur les clients et les configurations possibles, consultez la section Méthodes d'accès et clients AppStream 2.0 du guide d'administration Amazon AppStream 2.0. Envisagez d'utiliser le client natif pour une expérience optimale.
2. Dans la AppStream version 2.0, choisissez Desktop.

Étape 3 : configurer le moteur d'exécution

Au cours de cette étape, vous allez configurer le moteur d'exécution AWS Blu Age. Vous devez configurer le moteur d'exécution au premier lancement, puis à nouveau si vous êtes informé d'une mise à niveau du moteur d'exécution. Cette étape permet de remplir votre `.m2` dossier.

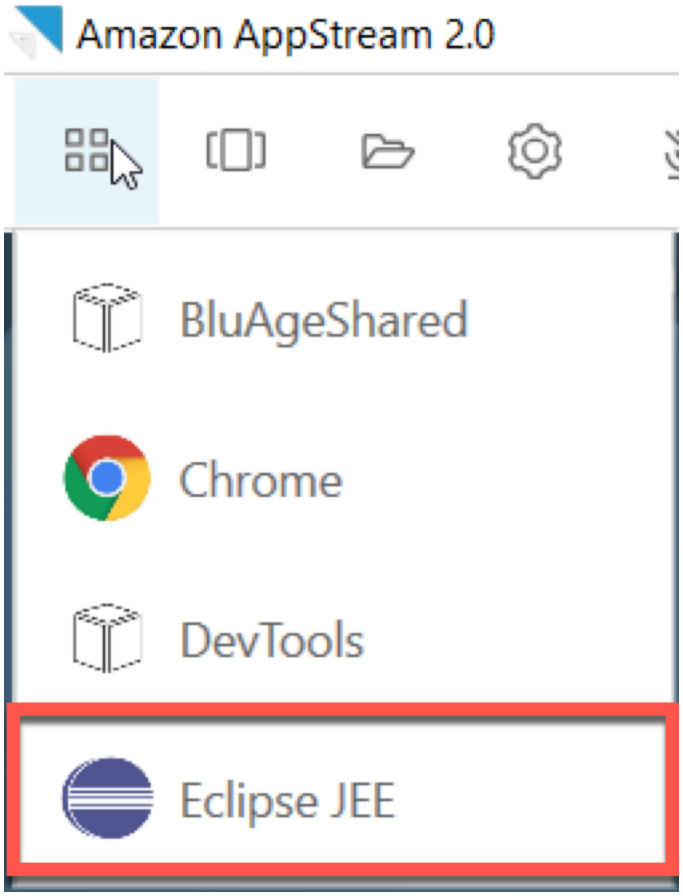
1. Choisissez Applications dans la barre de menu, puis Terminal.
2. Entrez la commande suivante :

```
~/_install-velocity-runtime.sh
```

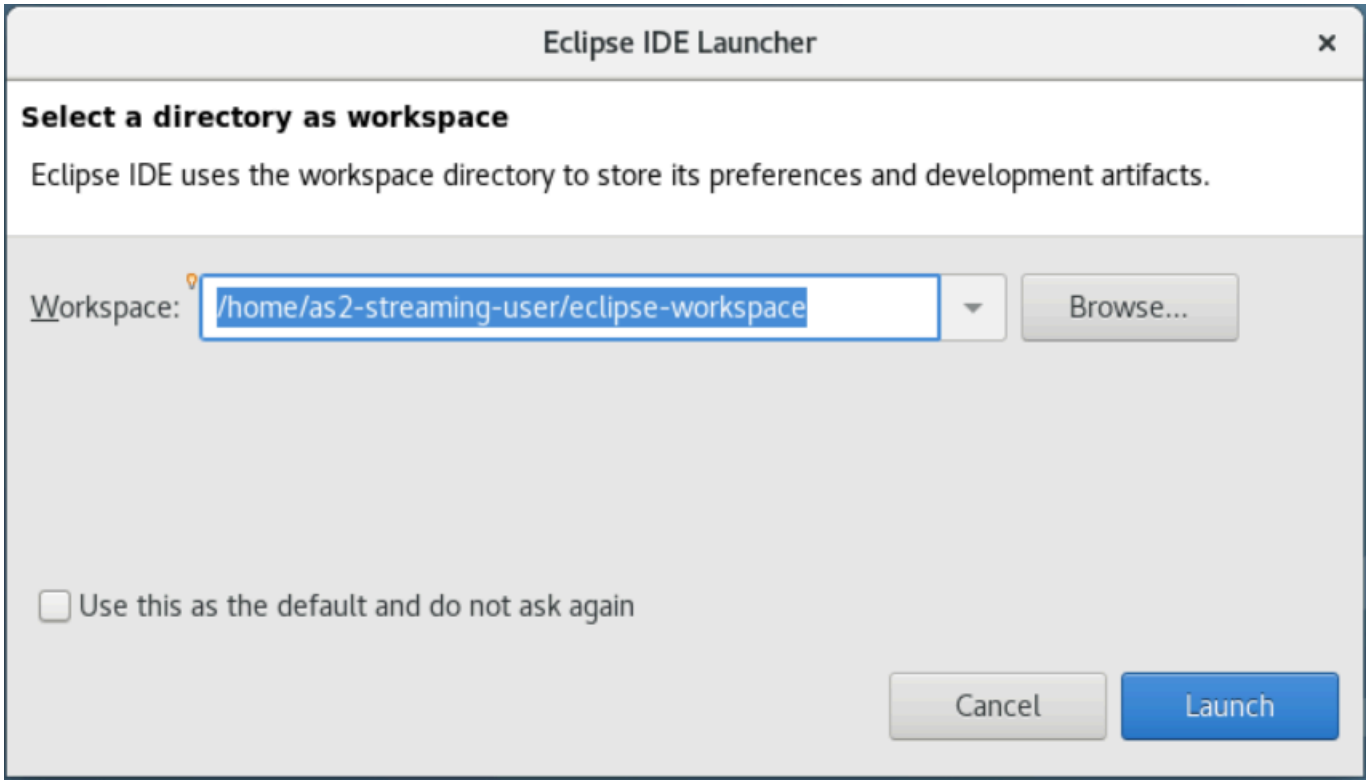
Étape 4 : démarrer l'IDE Eclipse

Au cours de cette étape, vous démarrez l'IDE Eclipse et choisissez un emplacement dans lequel vous souhaitez créer un espace de travail.

1. Dans la AppStream version 2.0, choisissez l'icône Lancer l'application dans la barre d'outils, puis choisissez Eclipse JEE.



2. Lorsque le lanceur s'ouvre, entrez l'emplacement où vous souhaitez créer votre espace de travail, puis choisissez Launch.



Vous pouvez éventuellement lancer Eclipse depuis la ligne de commande, comme suit :

```
~/eclipse &
```

Étape 5 : configurer un projet Maven

Au cours de cette étape, vous importez un projet Maven pour l'application de démonstration Planets.

1. Téléchargez le PlanetsDemofichier -pom.zip dans votre dossier d'accueil. Pour ce faire, vous pouvez utiliser la fonctionnalité « Mes fichiers » du client natif.
2. Utilisez l'outil de ligne de `unzip` commande pour extraire les fichiers.
3. Naviguez dans le dossier décompressé et ouvrez la racine `pom.xml` de votre projet dans un éditeur de texte.
4. Modifiez la `gapwalk.version` propriété afin qu'elle corresponde au moteur d'exécution AWS Blu Age installé.

   Si vous n'êtes pas sûr de la version installée, lancez la commande suivante dans un terminal :

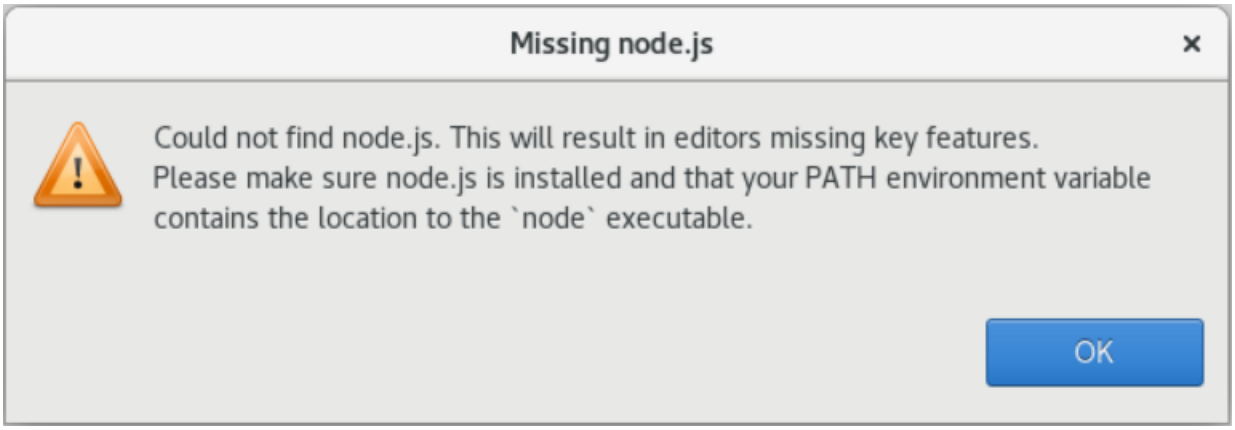
   ```
   cat ~/runtime-version.txt
   ```

Cette commande imprime la version d'exécution actuellement disponible, par exemple`3.1.0-b3257-dev`.

> **Note**
>
> N'incluez pas le `-dev` suffixe dans. `gapwalk.version` Par exemple, une valeur valide serait`<gapwalk.version>3.1.0-b3257</gapwalk.version>`.

5. Dans Eclipse, choisissez Fichier, puis Importer. Dans la fenêtre de dialogue Importer, développez Maven et choisissez Existing Maven Projects. Choisissez Next (Suivant).
6. Dans Import Maven Projects, indiquez l'emplacement des fichiers extraits et choisissez Terminer.

   Vous pouvez ignorer la fenêtre contextuelle suivante en toute sécurité. Maven télécharge une copie locale de `node.js` pour créer la partie Angular (*-web) du projet :

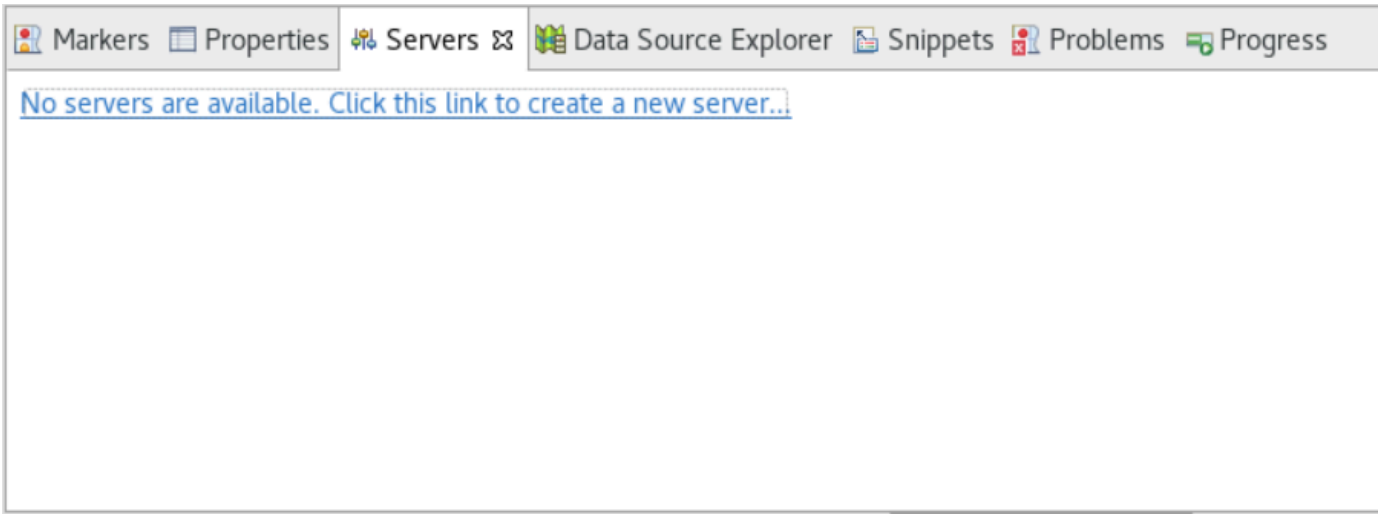
   

   Attendez la fin de la construction. Vous pouvez suivre la construction dans la vue Progression.
7. Dans Eclipse, sélectionnez le projet et choisissez Exécuter en tant que. Choisissez ensuite l'installation de Maven. Une fois l'installation de Maven réussie, le `war` fichier est créé sous. `PlanetsDemoPom/PlanetsDemo-web/target/PlanetsDemo-web-1.0.0.war`

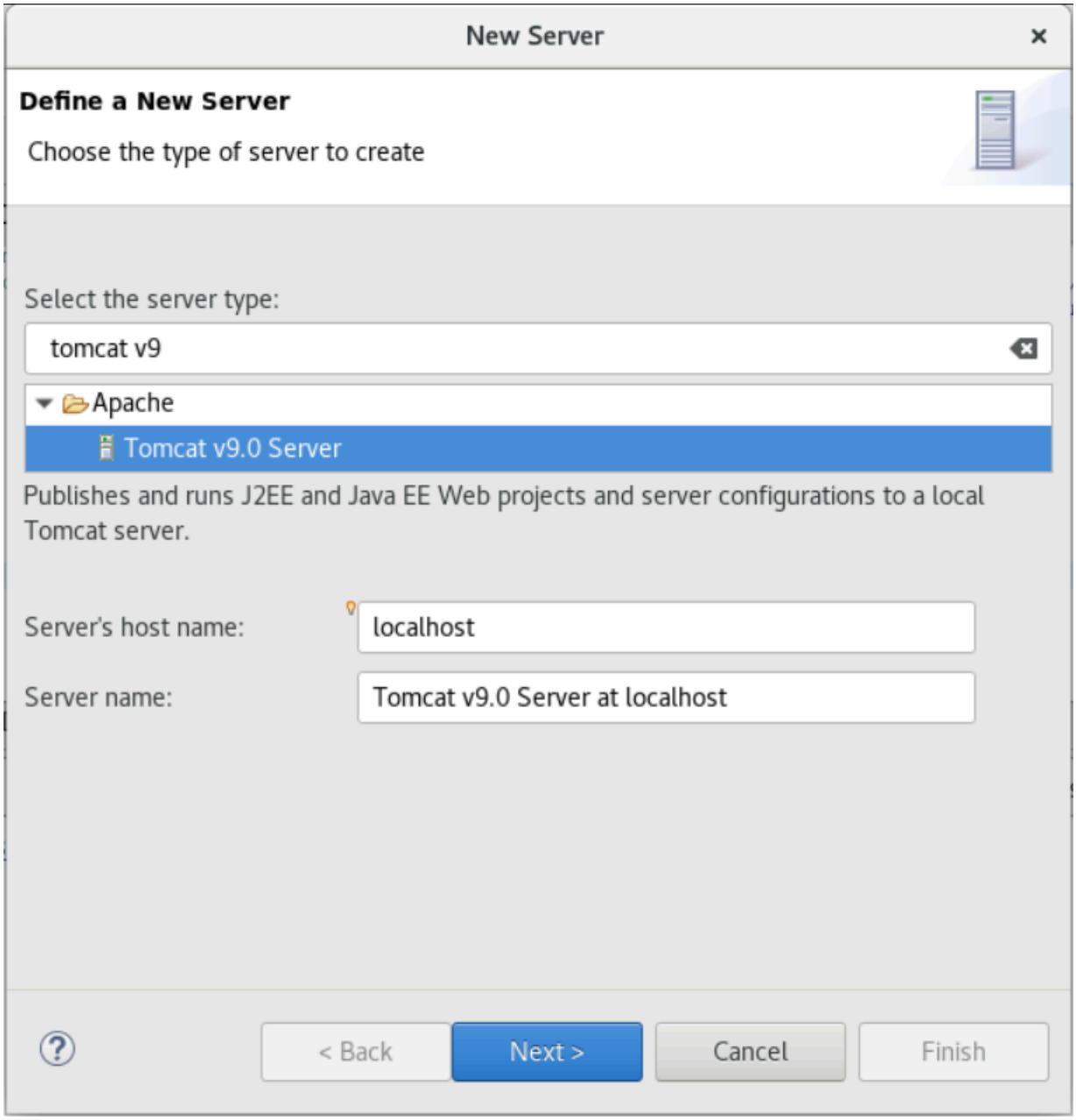
Étape 6 : Configuration d'un serveur Tomcat

Au cours de cette étape, vous configurez un serveur Tomcat sur lequel vous déployez et démarrez votre application compilée.

1. Dans Eclipse, choisissez Fenêtre > Afficher la vue > Serveurs pour afficher la vue Serveurs :

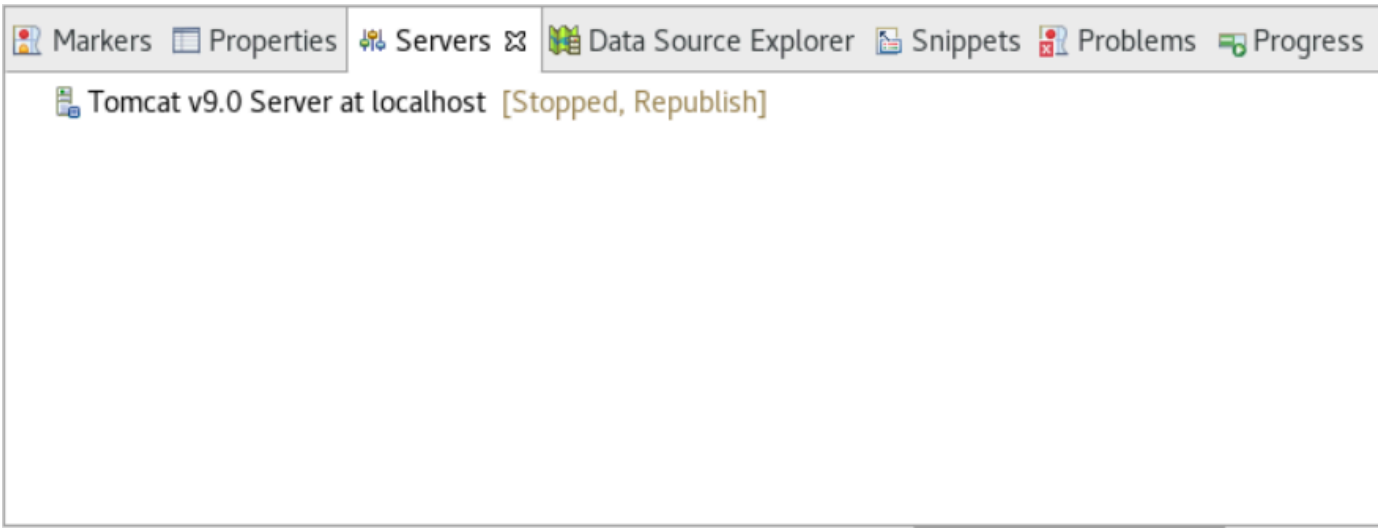


2. Choisissez Aucun serveur n'est disponible. Cliquez sur ce lien pour créer un nouveau serveur... . L'assistant Nouveau serveur apparaît. Dans le champ Sélectionnez le type de serveur de l'assistant, entrez tomcat v9, puis choisissez Tomcat v9.0 Server. Ensuite, sélectionnez Suivant.



3. Choisissez Browse, puis choisissez le dossier Tomcat à la racine du dossier Home. Laissez le JRE à sa valeur par défaut et choisissez Terminer.

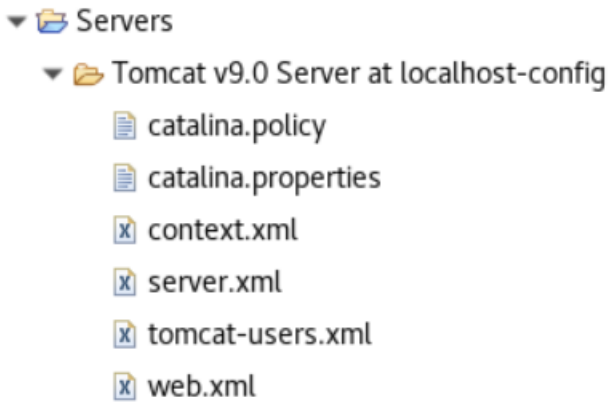
   Un projet Servers est créé dans l'espace de travail, et un serveur Tomcat v9.0 est désormais disponible dans la vue Serveurs. C'est ici que l'application compilée sera déployée et démarrée :



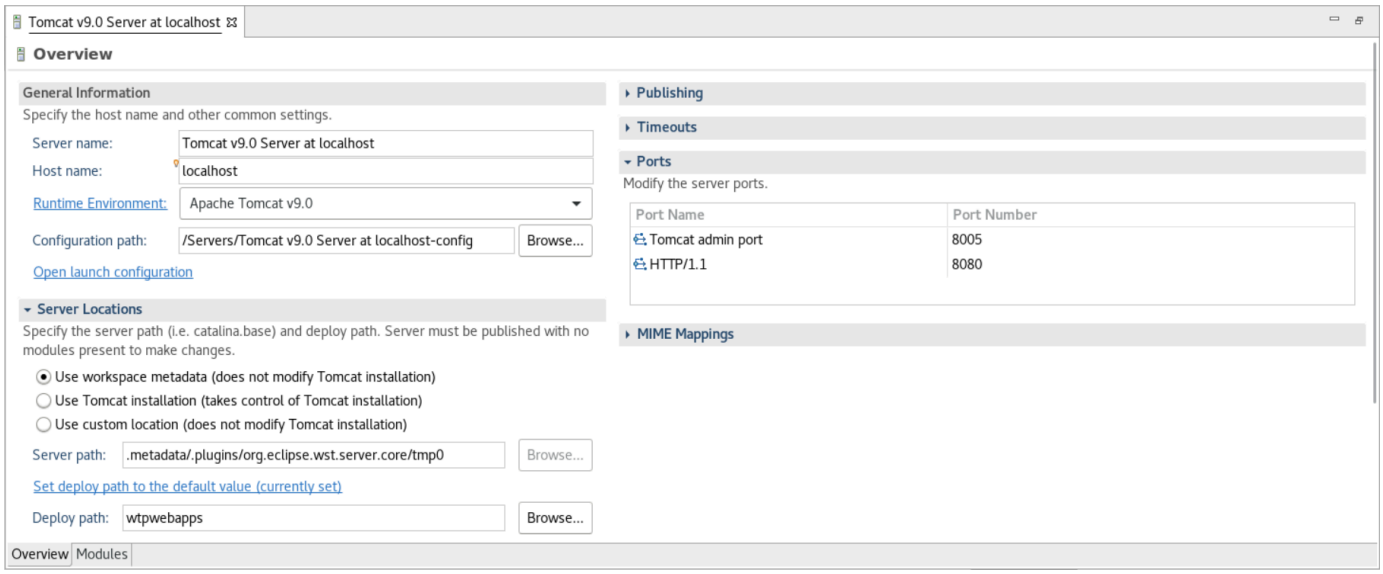
Étape 7 : Déployer sur Tomcat

Au cours de cette étape, vous déployez l'application de démonstration Planets sur le serveur Tomcat afin de pouvoir exécuter l'application.

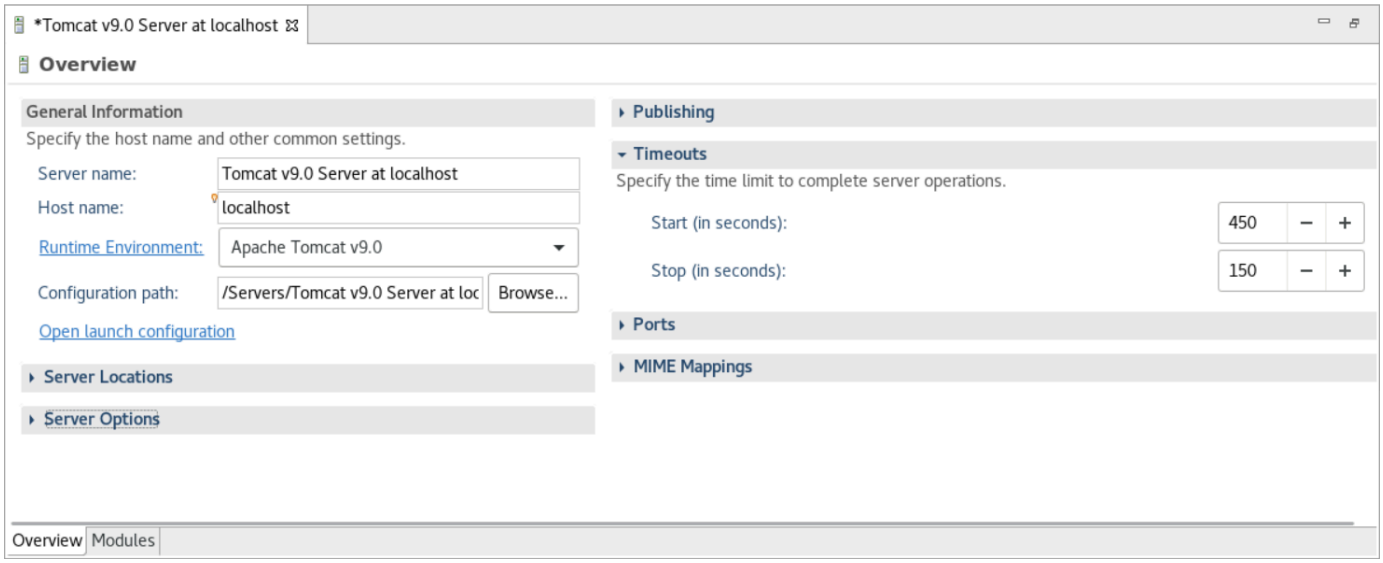
1. Sélectionnez le `PlanetsDemo-web` fichier et choisissez Exécuter sous > Installation de Maven. Sélectionnez `PlanetsDemo-web` à nouveau et choisissez Refresh pour vous assurer que le frontend compilé par npm est correctement compilé dans un fichier .war et remarqué par Eclipse.
2. Téléchargez le PlanetsDemofichier -runtime.zip sur l'instance, puis décompressez le fichier à un emplacement accessible. Cela garantit que l'application de démonstration peut accéder aux dossiers et fichiers de configuration dont elle a besoin.
3. Copiez le contenu de `PlanetsDemo-runtime/tomcat-config` dans le `Servers/Tomcat v9.0...` sous-dossier que vous avez créé pour votre serveur Tomcat :



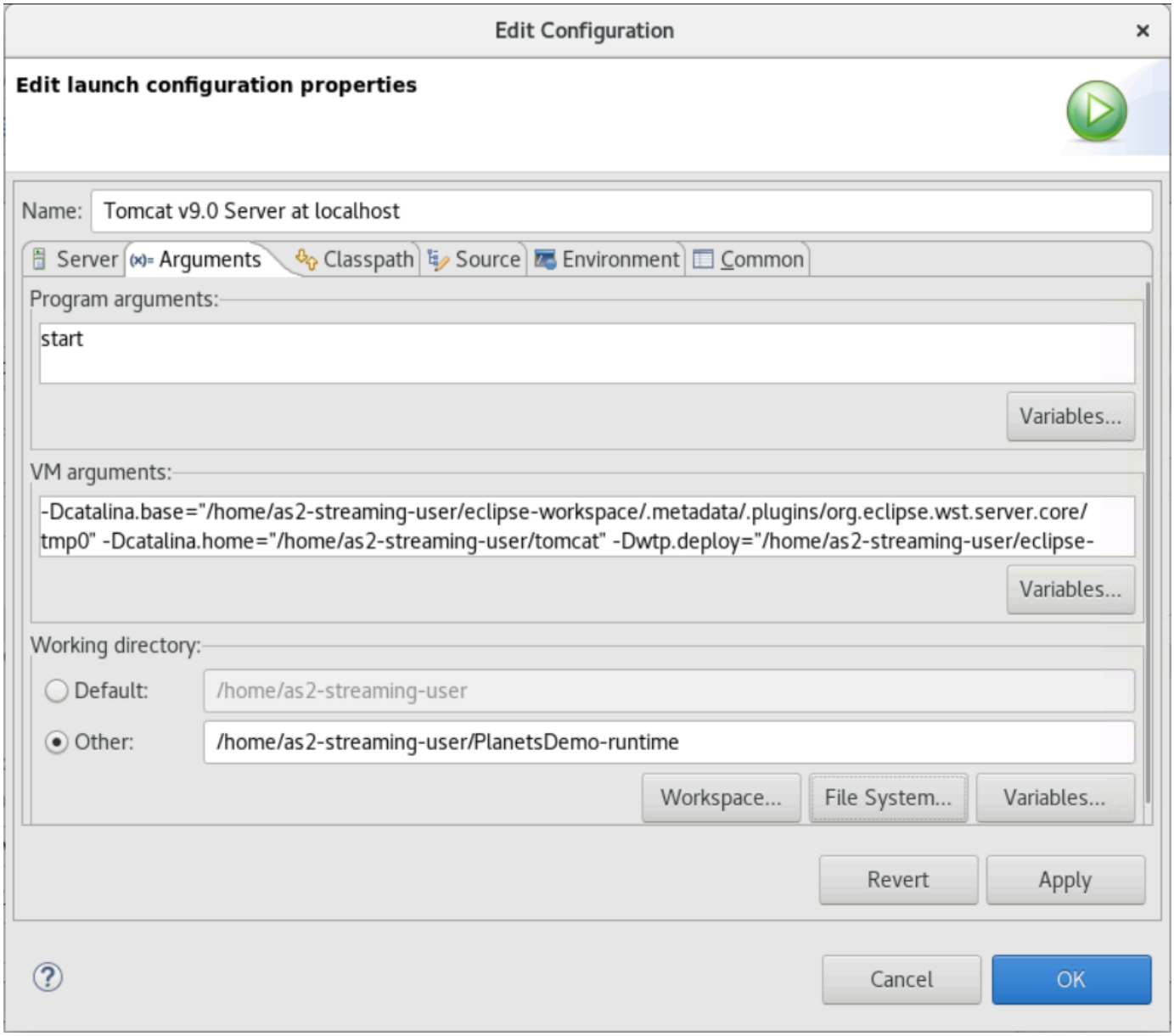
4. Ouvrez l'entrée `tomcat v9.0` du serveur dans la vue Serveurs. L'éditeur de propriétés du serveur apparaît :



5. Dans l'onglet Aperçu, augmentez les valeurs des délais d'attente à 450 secondes pour le démarrage et à 150 secondes pour l'arrêt, comme indiqué ici :



6. Choisissez Ouvrir la configuration de lancement. Un magicien apparaît. Dans l'assistant, accédez au dossier Arguments et, dans le répertoire de travail, sélectionnez Autre. Choisissez Système de fichiers, puis accédez au `PlanetsDemo-runtime` dossier décompressé précédemment. Ce dossier doit contenir un sous-dossier direct appelé config.

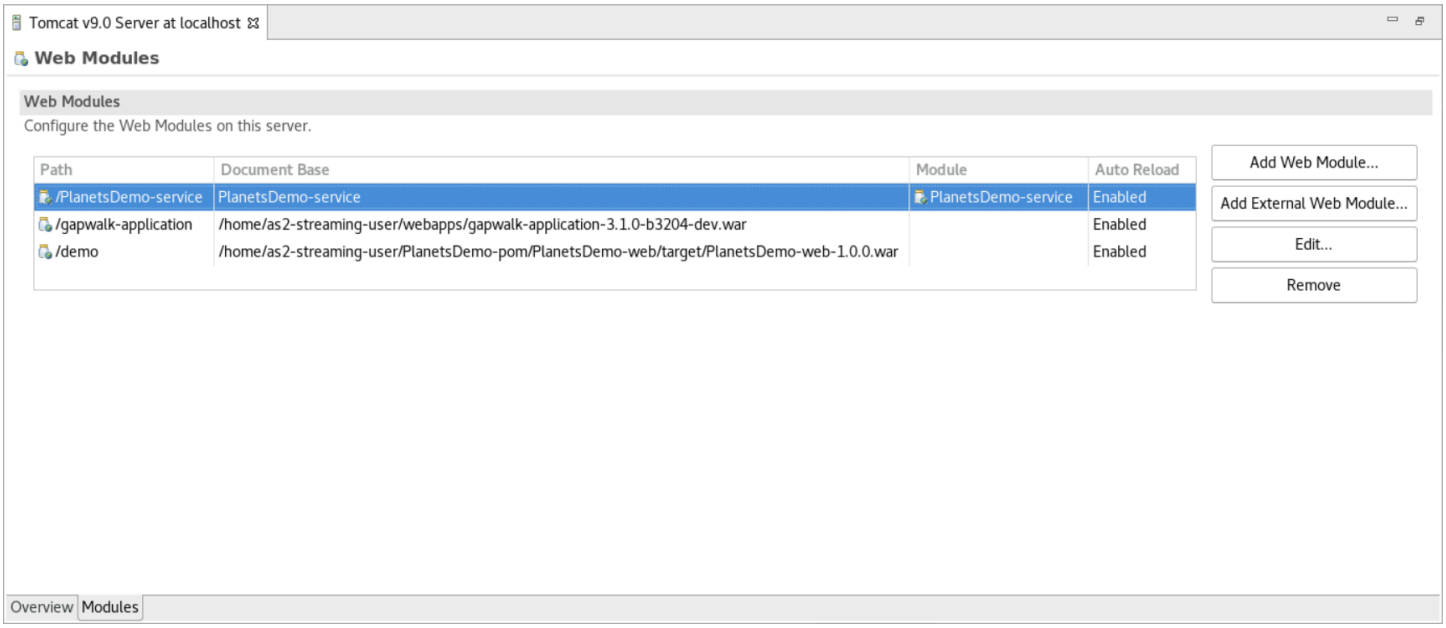


7. Choisissez l'onglet Modules de l'éditeur de propriétés du serveur et apportez les modifications suivantes :

   - Choisissez Ajouter un module Web, puis ajoutez`PlanetsDemo-service`.
   - Choisissez Ajouter un module Web externe. La fenêtre de dialogue Ajouter un module Web apparaît. Effectuez les modifications suivantes :
     - Dans la base de documents, choisissez Parcourir et accédez à `~/webapps/gapwalk-application...war`
     - Dans Path, entrez`/gapwalk-application`.
   - Choisissez OK.

- Choisissez à nouveau Ajouter un module Web externe et apportez les modifications suivantes :
  - Pour Document base, entrez le chemin d'accès au frontend .war (in) `PlanetsDemo-web/target`
  - Pour Path, entrez `/demo`
- Sélectionnez OK
- Enregistrez les modifications de l'éditeur (Ctrl + S).

Le contenu de l'éditeur doit maintenant être similaire à ce qui suit.



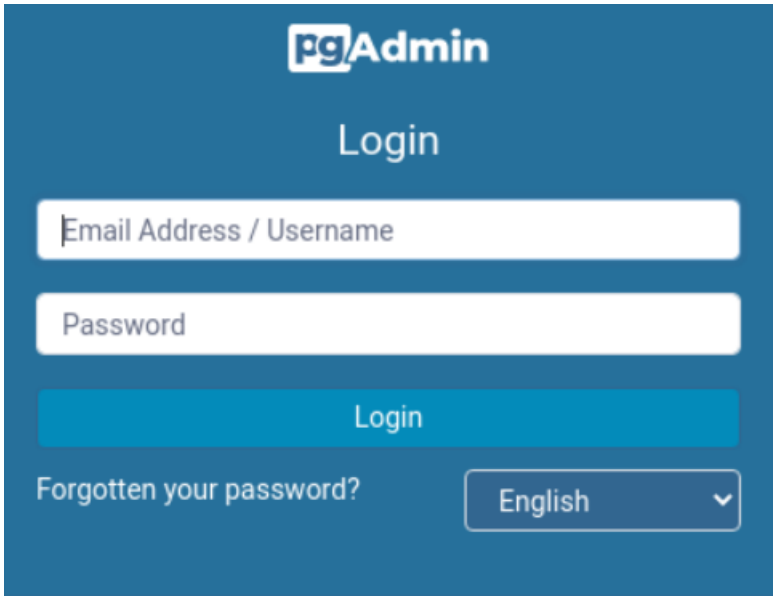
Étape 8 : Création de la base de données JICS

Au cours de cette étape, vous vous connectez à la base de données que vous avez créée dansÉtape 1 : Créer une base de données.

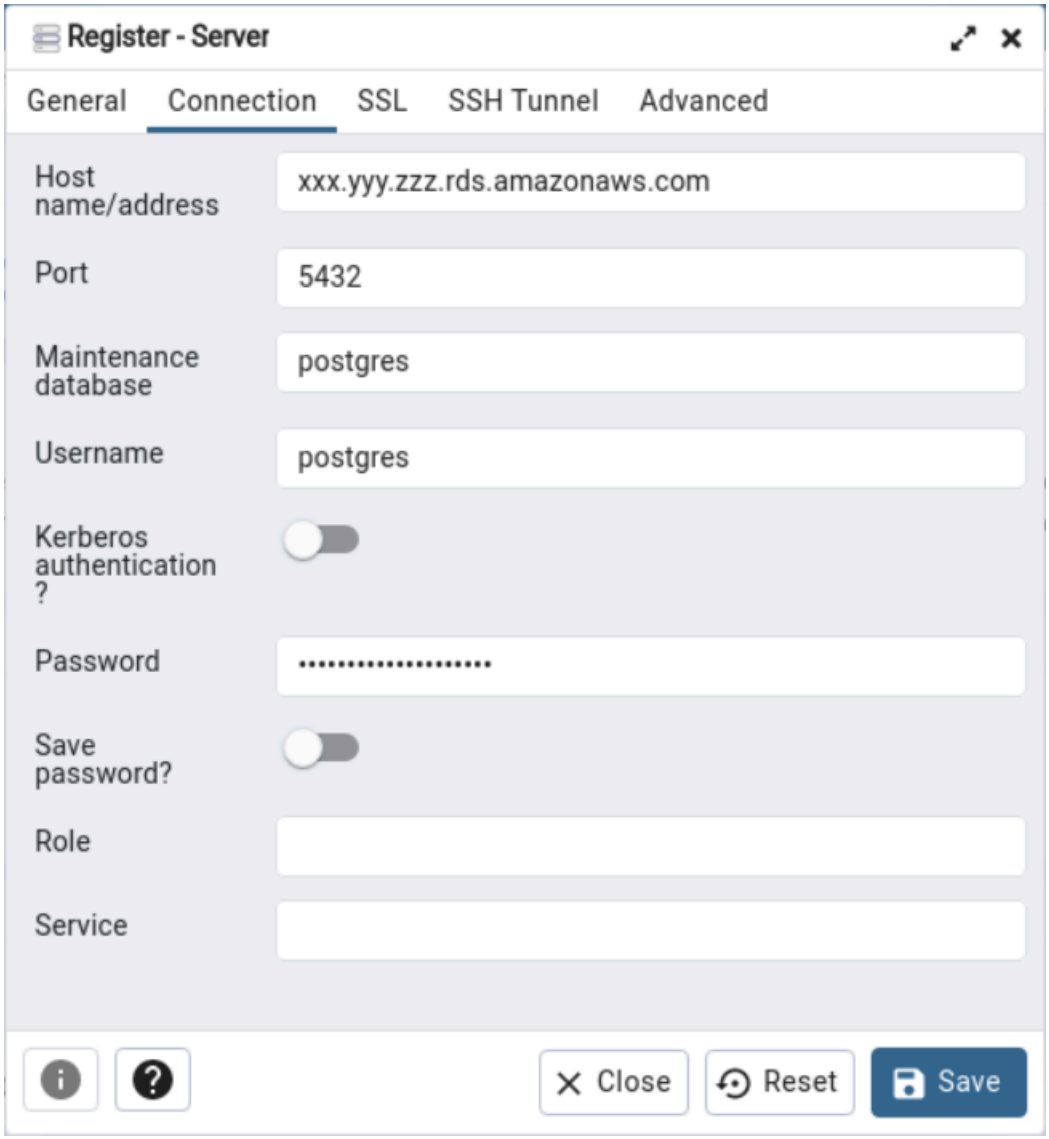
1. À partir de l'instance AppStream 2.0, exécutez la commande suivante dans un terminal pour le lancer `pgAdmin` :

   ```
   ./pgadmin-start.sh
   ```

2. Choisissez une adresse e-mail et un mot de passe comme identifiants de connexion. Prenez note de l'URL fournie (généralement http://127.0.0.1:5050). Lancez Google Chrome dans l'instance, copiez-collez l'URL dans le navigateur et connectez-vous à l'aide de vos identifiants.



3. Une fois connecté, choisissez Ajouter un nouveau serveur et entrez les informations de connexion à la base de données créée précédemment comme suit.

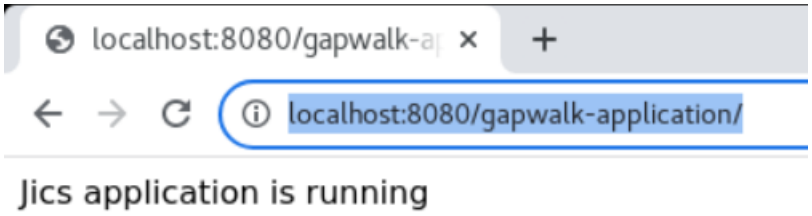


4. Lorsque vous vous connectez au serveur de base de données, utilisez Objet > Créer > Base de données et créez une nouvelle base de données nommée jics.
5. Modifiez les informations de connexion à la base de données utilisées par l'application de démonstration. Ces informations sont définies dans`PlanetsDemo-runtime/config/application-main.yml`. Recherchez l'`jicsDs`entrée. Pour récupérer les valeurs pour `username` et`password`, dans la console Amazon RDS, accédez à la base de données. Dans l'onglet Configuration, sous Master Credentials ARN, choisissez Gérer dans Secrets Manager. Ensuite, dans la console Secrets Manager, dans le secret, choisissez Retrieve secret value.

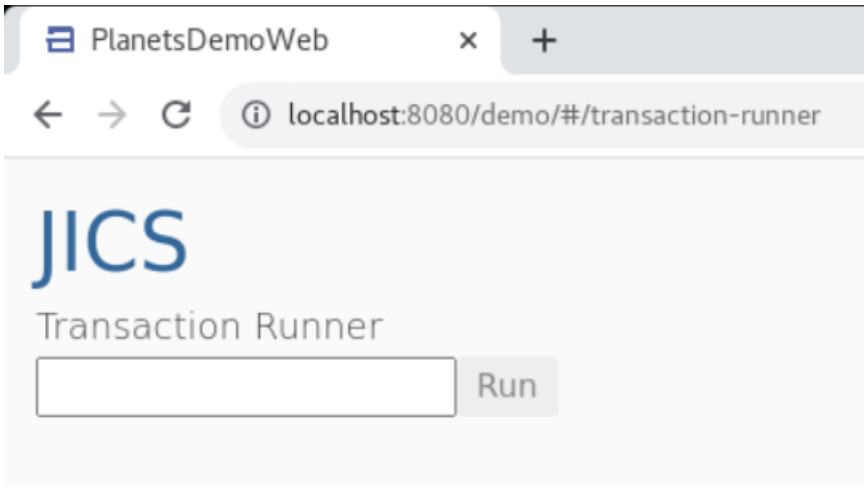
Étape 9 : démarrer et tester l'application

Au cours de cette étape, vous lancez le serveur Tomcat et l'application de démonstration afin de pouvoir le tester.

1. Pour démarrer le serveur Tomcat et les applications précédemment déployées, sélectionnez l'entrée du serveur dans la vue Serveurs et choisissez Démarrer. Une console affiche les journaux de démarrage.
2. Vérifiez l'état du serveur dans la vue Serveurs ou attendez le message de démarrage du serveur en [xxx] millisecondes dans la console. Après le démarrage du serveur, vérifiez que l'application gapwalk est correctement déployée. Pour ce faire, accédez à l'URL http://localhost:8080/gapwalk-application dans un navigateur Google Chrome. Vous devriez voir ce qui suit.

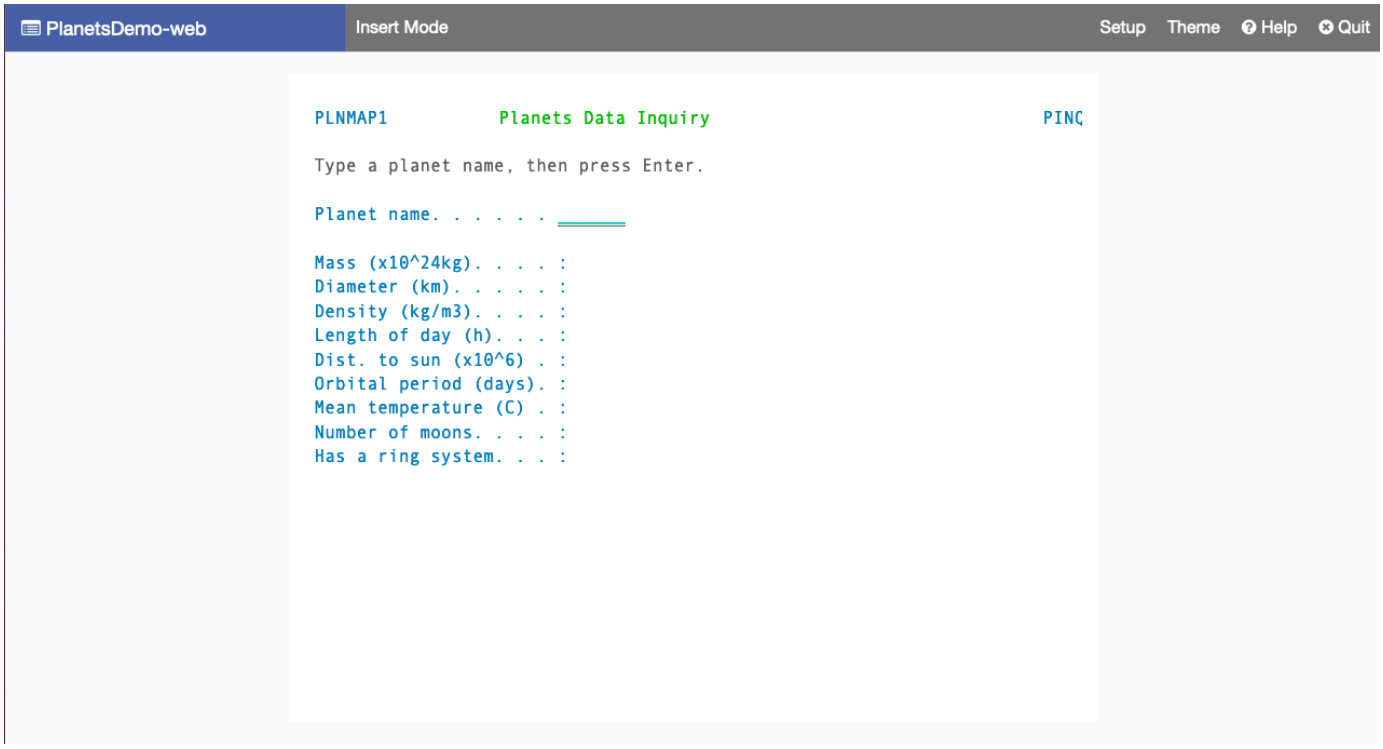


3. Accédez à l'interface de l'application déployée depuis Google Chrome à l'adresse http://localhost:8080/demo. La page Transaction Launcher suivante devrait apparaître.

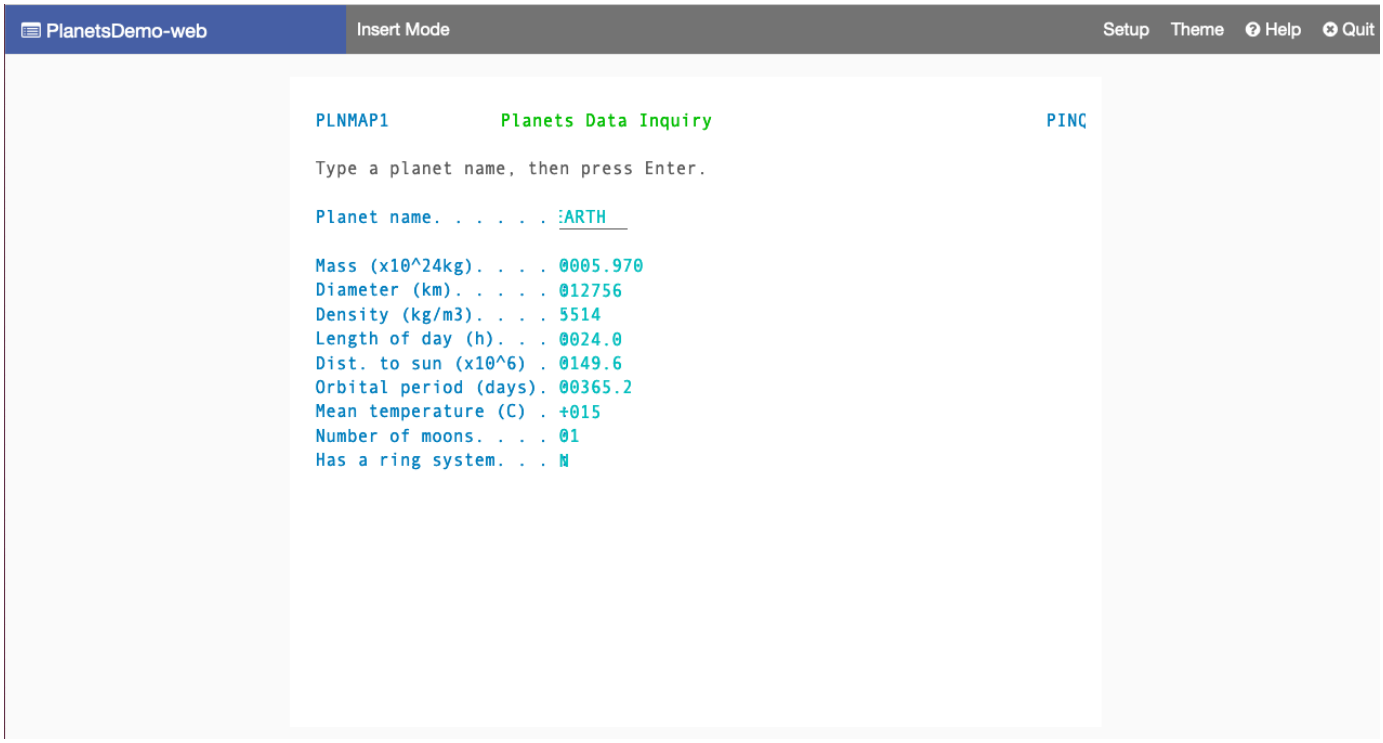


4. Pour démarrer la transaction de l'application, entrez `PINQ` dans le champ de saisie et choisissez Exécuter (ou appuyez sur Entrée).

   L'écran de l'application de démonstration devrait apparaître.



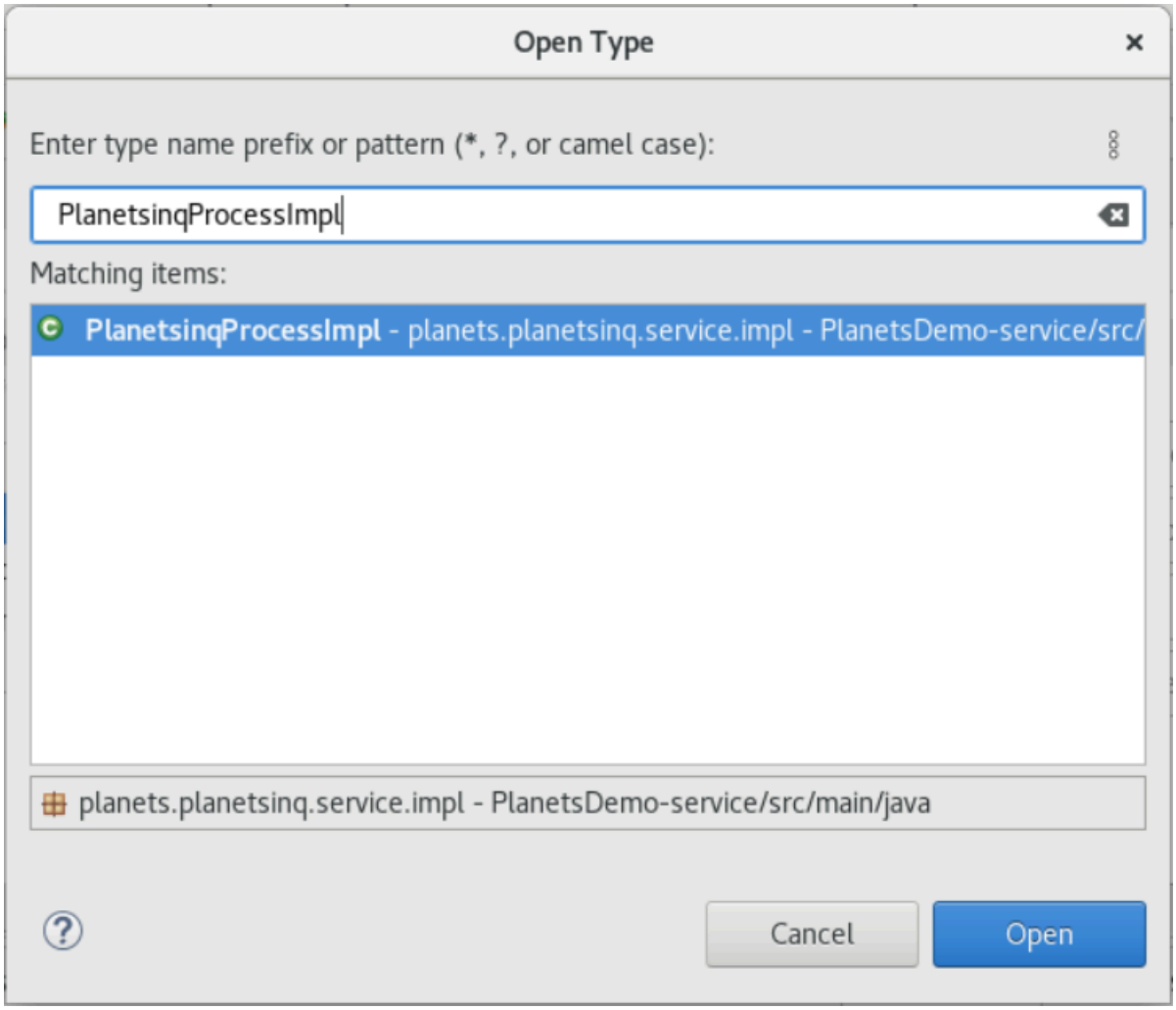
5. Tapez le nom d'une planète dans le champ correspondant et appuyez sur Entrée.



Étape 10 : Déboguer l'application

Au cours de cette étape, vous testez à l'aide des fonctionnalités de débogage standard d'Eclipse. Ces fonctionnalités sont disponibles lorsque vous travaillez sur une application modernisée.

1. Pour ouvrir la classe de service principale, appuyez sur Ctrl + Shift + T. Puis entrez. `PlanetsinqProcessImpl`



2. Accédez à la `searchPlanet` méthode et placez-y un point d'arrêt.
3. Sélectionnez le nom du serveur, puis sélectionnez Redémarrer dans le débogage.
4. Répétez les étapes précédentes. C'est-à-dire, accédez à l'application, entrez le nom d'une planète et appuyez sur Entrée.

   Eclipse arrêtera l'application dans la `searchPlanet` méthode. Vous pouvez maintenant l'examiner.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin de ne pas avoir à payer de frais supplémentaires. Procédez comme suit :

- Si l'application Planets est toujours en cours d'exécution, arrêtez-la.
- Supprimez la base de données que vous avez créée dansÉtape 1 : Créer une base de données. Pour plus d'informations, consultez Suppression d'une instance de base de données.

AWS FAQ sur Blu Age

Général

1. Quel est l'objectif principal de la capacité de refactorisation de AWS Blu Age ?

   La fonctionnalité de refactorisation refactorise le code monolithique existant en Java à l'aide d'applications distribuées contemporaines utilisant des langages et des frameworks modernes, selon un modèle de refactorisation automatique. Ce modèle implique d'analyser automatiquement le code existant, de comprendre ses fonctionnalités et de le convertir en code moderne équivalent tout en préservant la logique métier. Le processus inclut la modernisation non seulement du code, mais également de l'ensemble de la pile d'applications, des dépendances et de l'infrastructure à l'aide d'outils et de processus automatisés. La solution vise à accélérer la modernisation tout en maintenant l'équivalence fonctionnelle et les performances. Cela inclut la transformation du code des applications ainsi que des bases de données et des magasins de données associés, tout en mettant en œuvre les meilleures pratiques et les modèles de conception du cloud.
2. Quelles applications mainframe sont prises en charge par AWS Blu Age ?

   AWS Blu Age soutient actuellement la modernisation d'IBM z/OS applications mainframe écrites en COBOL, PL/I, JCL (Job Control Language) et s'appuyant sur le gestionnaire de transactions CICS (Customer Information Control System), les écrans BMS (Basic Mapping Support), les écrans IMS MFS, les bases de données, les DB2 bases de données IMS, les fichiers de données GDG (Generation Data Groups) et VSAM (Virtual Storage Access Method). Pour plus de détails, consultez AWS Blu Insights.
3. Quels langages d'ordinateur central peuvent être modernisés par AWS Blu Age ?

   AWS Blu Age transforme le code COBOL et PL/I en Java, en Groovy, JCLs les écrans (BMS ou MFS) en HTML (avec Sass) et JavaScript (applications angulaires — React n'est pas pris

en charge pour le moment), permettant ainsi la modernisation des applications mainframe existantes vers des architectures cloud natives. Ces technologies sont choisies pour leur adoption généralisée, leur écosystème robuste et leurs fonctionnalités cloud natives. Angular fournit une couche d'interface utilisateur moderne et réactive qui remplace les anciennes interfaces à écran vert. Il permet de créer des applications Web dynamiques et conviviales accessibles sur différents appareils et plateformes. Son architecture basée sur les composants permet un développement frontal maintenable et évolutif. La transformation se traduit par des applications distribuées qui suivent les modèles architecturaux modernes et les meilleures pratiques.

4. Comment AWS Blu Age arrive-t-il à trouver un équilibre entre les contraintes héritées et les avantages du cloud ?

   AWS Blu Age atteint l'équilibre en préservant la logique et les fonctionnalités essentielles de l'entreprise tout en introduisant des fonctionnalités cloud natives. Il garantit que les applications modernisées conservent la logique métier existante nécessaire tout en tirant parti de l'évolutivité, de l'agilité et des pratiques opérationnelles modernes du cloud. Cette approche aide les entreprises à maintenir la continuité de leurs activités tout en tirant parti des avantages de l'infrastructure cloud.

5. Quel est le rôle de l'architecture orientée services dans l'application modernisée ?

   L'architecture orientée services joue un rôle fondamental dans la décomposition des applications monolithiques en composants modulaires plus faciles à gérer. AWS Blu Age crée des applications orientées services et orientées objet qui améliorent la maintenabilité et l'évolutivité. Cette approche architecturale permet aux entreprises d'améliorer leur efficacité commerciale et de se préparer à une éventuelle adoption future des microservices.

6. Quels aspects de la pile d'applications sont inclus dans le processus de refactorisation ?

   Le processus de refactorisation inclut la pile logicielle complète : code de l'application, dépendances, bases de données et infrastructure (par exemple, options de mise en cache, support de messagerie, etc.). Il couvre la transformation des anciens langages de programmation, des systèmes de base de données, des fichiers de données et des composants d'infrastructure associés. Cette approche globale garantit que tous les aspects de l'application sont modernisés de manière cohérente, ce qui se traduit par une pile d'applications modernes entièrement transformée.

7. Le processus de modernisation de AWS Blu Age élimine-t-il le besoin de tests ou de contrôles d'assurance qualité sur l'application Java modernisée ?

Non, le processus de modernisation de AWS Blu Age n'élimine pas le besoin de tests ou de contrôles d'assurance qualité sur l'application Java modernisée.

8. Que signifie AWS Blu Age JAC ?

   JAC est l'abréviation de JICS Administration Console

9. Comment puis-je accéder à l'outillage AWS Blu Age ?

   AWS Les outils Blu Age sont accessibles via la console AWS via AWS Mainframe Modernization (M2) Refactor, avec un accès aux fonctionnalités basé sur votre niveau d'accréditation. Commencez par le centre de transformation pour évaluer la refactorisation automatique de votre code source en Java. Pour obtenir des conseils détaillés, reportez-vous à la documentation AWS Blu Insights. Après la modernisation, vous pouvez déployer des applications à l'aide d'options d'exécution gérées ou non gérées. Pour plus d'informations sur ces choix de déploiement, consultez la documentation sur la modernisation AWS du mainframe.

10. Comment dimensionner (charge de travail et calendrier) un projet ?

    Consultez les estimations de AWS Blu Insights pour plus d'informations à ce sujet ou contactez votre responsable de compte.

11. Existe-t-il des exigences spécifiques pour la maintenance des solutions migrées vers Java AWS Blu Age ?

    Non, il n'existe aucune exigence spécifique pour la maintenance des solutions migrées vers Java AWS Blu Age.

12. Quelles sont les spécifications techniques et la compatibilité du code généré par AWS Blu Age ?

    AWS Le code généré par Blu Age est conçu avec des caractéristiques techniques spécifiques et une large compatibilité. Bien qu'il ne supporte pas JPA, il utilise l'exécution directe du SQL avec des requêtes externalisées. Le code s'appuie sur des bibliothèques spécifiques à l'exécution pour l'équivalence fonctionnelle, la génération de services Web et les implémentations MQ. Le code généré peut être importé dans n'importe quel IDE Java à des fins de développement, de test, de construction et de déploiement, mais les bibliothèques requises doivent être importées en conséquence. Alors que Maven est intégré par défaut au service de modernisation du AWS mainframe pour les processus de construction, des outils alternatifs tels que Gradle peuvent être utilisés en modifiant le format d'emballage après la transformation. La plateforme offre de la flexibilité en termes d'outils de développement et de contrôle des sources, avec des formations

disponibles pour les équipes de développement gérant le code. Pour plus d'informations, consultez la section Architecture de haut niveau de AWS Blu Age Runtime.

AWS Blue Age Runtime

1. Où puis-je trouver des informations sur AWS Blu Age Runtime ?

   Consultez la documentation relative à la configuration d'AWS Blu Age Runtime (non géré) sur l'environnement d'exécution non géré qui détaille le processus de configuration, l'intégration, la récupération des artefacts, le déploiement, etc.

2. Où puis-je trouver AWS Blu Age Runtime pour les développeurs ?

   Le AWS Blu Age Runtime pour les développeurs est disponible dans Blu Age Toolbox pour les personnes certifiées L3.

3. Les dépendances JAR de AWS Blu Age sont-elles téléchargées dans le référentiel Maven du client pour le développement local ?

   Les bibliothèques peuvent être importées à EC2 l'aide d'une AMI qui peut être utilisée pour configurer l'environnement de développement, de test et de production. Une formation et des habilitations seront dispensées à l'équipe pour maintenir/améliorer le code d'application généré. Pour plus d'informations, consultez la section Architecture de haut niveau de AWS Blu Age Runtime.

4. À quoi fait référence le terme « Gapwalk » dans les fichiers JAR distribués par AWS Blu Age Runtime ?

   Pour plus d'informations sur Gapwalk, consultez la section Artefacts de AWS Blu Age Runtime.

5. Comment demander l'accès au AWS Blu Age Runtime non géré ?

   Suivez les instructions d'intégration de AWS Blu Age Runtime pour demander l'accès au AWS Support centre.

6. Quels sont les environnements d'exécution pris en charge pour les applications refactorisées AWS Blu Age ?

   Pour découvrir la gamme complète des options d'exécution pour vos applications modernisées, nous vous recommandons de consulter le guide des options d'exécution de Blu Age.

7. Quand utilise-t-on le AWS Blu Age Runtime ?

Un AWS Blu Age Runtime est nécessaire pour prendre en charge l'exécution des applications refactorisées AWS Blu Age. Un environnement d'exécution est nécessaire pendant les projets de refactoring basés sur AWS Blu Age pour tester les applications refactorisées. Une fois le projet de refactorisation terminé, un environnement d'exécution est également nécessaire pour maintenir, tester et exécuter les applications AWS refactorisées Blu Age en production.

8. Comment AWS distribue-t-on les nouvelles versions de AWS Blu Age Runtime ?

   Pour M2 Managed Runtime, les mises à jour, y compris les correctifs, les versions mineures et majeures, sont disponibles dans la AWS console et AWS CLI. Ils incluent les mises à jour du système d'exploitation, les modifications du moteur et des dépendances, généralement dans les 30 jours suivant la disponibilité générale. AWS est responsable des composants pris en charge et applique automatiquement les mises à jour aux instances de modernisation du AWS mainframe. Il en va de même pour d'autres environnements tels que Custom Runtime, Linux AMI et sur site.

9. À quelle fréquence les nouvelles versions majeures et mineures de l'environnement d'exécution AWS Blu Age sont-elles publiées ?

   Les nouvelles versions sont publiées une fois par mois ou tous les deux mois, et les clients peuvent décider quand et comment mettre à niveau leurs instances d'exécution. Pour plus d'informations, consultez la page de gestion des versions de AWS Blu Age.

10. Comment puis-je AWS fournir une assistance pour AWS Blu Age Runtime ?

    Support est fourni par le biais AWS Support duquel les problèmes sont résolus en soulevant un ticket, et le SLA standard s'applique. Pour plus d'informations, consultez la section Cycle de vie des composants de modernisation du AWS mainframe.

11. En quoi consiste la modernisation du AWS mainframe AWS Blue Age Runtime ?

    Le AWS Blu Age Runtime inclut des bibliothèques de boîtes à outils pour accélérer la modernisation, faciliter les intégrations dans le cloud et améliorer la qualité et la maintenabilité du code. Il permet également une plus grande automatisation de la modernisation en facilitant les transitions entre les architectures existantes et les architectures cloud. Le moteur d'exécution prend en charge la gestion des verbes hérités et des structures de données et des représentations de mémoire à l'aide d'expressions idiomatiques Java. Il permet de créer des applications modernisées basées sur des techniques de programmation orientées objet et capables de reproduire les flux de contrôle existants. Il modernise les anciens ensembles de données VSAM ou le support des bases de données hiérarchiques IMS à l'aide d'une base de données relationnelle telle qu'Amazon Aurora. Il fournit des remplacements Java pour les utilitaires système existants

(IDCAMS, IEBGENER, DFSORT, etc.) et les anciens systèmes de gestion des transactions (CICS, IMS). Il facilite les intégrations dans le cloud avec la mise en cache dans Amazon ElastiCache et la prise en charge des solutions AWS de messagerie (SQS, Kinesis).

12. AWS Blu Age Runtime prend-il en charge les architectures informatiques autres que x86 ?

    Actuellement, AWS Blu Age Runtime ne prend en charge que les architectures informatiques et le calcul basés sur x86. AWS Blu Age Runtime ne prend pas en charge le calcul basé sur ARM et basé sur Graviton.

13. Comment les clients peuvent-ils rester informés des versions de AWS Blu Age Runtime, notamment en ce qui concerne les notifications relatives aux nouvelles versions et l'accès à l'historique des versions et aux notes de publication ?

    Les nouvelles versions de AWS Blu Age Runtime sont mises en ligne sur notre page de publication officielle. Nous vous recommandons de consulter cette page régulièrement, idéalement tous les 3 mois, pour connaître les dernières versions et mises à jour. En ce qui concerne l'accès à l'historique des versions et aux notes de publication, la disponibilité dépend de la date end-of-life (EOL) de chaque version majeure. Pour des informations détaillées sur les dates de fin de vie, la planification de la mise à niveau des versions et l'accès aux informations historiques, voir le cycle de vie de AWS Blu Age.

14. Quels sont les principaux composants de l'architecture de haut niveau de AWS Blu Age Runtime ?

    L'architecture AWS Blu Age Runtime comprend deux types de composants principaux. Les premières sont les bibliothèques Java (fichiers JAR) stockées dans un dossier partagé (accessible au chargeur de classes du serveur d'applications) qui fournissent la prise en charge des constructions et des instructions héritées. Ensuite, il y a les applications Web (fichiers war) contenant des applications basées sur Spring qui fournissent des cadres et des services aux programmes modernisés. Le moteur d'exécution inclut également : un registre des programmes qui rassemble tous les programmes pour les invocations et les appels interprogrammes et un registre des scripts qui collecte tous les scripts de tâches modernisés. Ces composants fonctionnent ensemble pour fournir un point d'entrée unifié basé sur REST et un cadre d'exécution pour les applications modernisées. Le Runtime et l'application modernisée sont déployés ensemble sur un serveur d'applications (Tomcat, par exemple).

15. Comment configurer le dossier partagé contenant les artefacts de AWS Blu Age Runtime ?

    Les artefacts AWS Blu Age Runtime (jar) doivent être rassemblés dans un dossier partagé, accessible au chargeur de classes du serveur d'applications. Pour un serveur Tomcat, la configuration est effectuée en modifiant le fichier de configuration normal nommé

catalina.properties. Par exemple, si vous avez créé le dossier partagé en tant que dossier nommé « shared », dans le dossier tomcat, vous devrez modifier l'entrée common.loader dans le fichier catalina.properties pour rendre le dossier partagé accessible au chargeur de classe tomcat, comme suit :

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/*.jar","${catalina.home}/
lib","${catalina.home}/lib/*.jar","${catalina.home}/shared","${catalina.home}/shared/
*.jar"
```

16. Comment AWS Blu Age Runtime gère-t-il l'apatridie et la gestion des sessions ?

    AWS Blu Age Runtime implémente l'apatridie et la gestion des sessions par le biais de multiples mécanismes. Pour les sessions HTTP, il utilise une identification basée sur les cookies avec un stockage de cache externe pour le contexte utilisateur. Les sessions peuvent être stockées dans différentes banques de données, notamment Amazon ElastiCache, le cluster Redis ou des cartes en mémoire. La conception de l'apatridie garantit que la plupart des états non transitoires sont stockés en externe dans une « source unique de vérité » commune, ce qui permet une haute disponibilité et une mise à l'échelle horizontale. Cette approche, combinée à l'équilibrage de charge et aux sessions partagées, permet de répartir les dialogues destinés aux utilisateurs sur plusieurs nœuds.

17. Quel rôle jouent les applications Web dans l'environnement AWS Blu Age Runtime ?

    Les applications Web de AWS Blu Age Runtime remplissent plusieurs fonctions clés. Ils fournissent des cadres d'exécution qui reproduisent les environnements existants et les moniteurs de transactions (tels que les lots JCL, CICS, IMS). Ils offrent des points d'entrée basés sur REST `gapwalk-application.war` pour déclencher et contrôler les transactions, les programmes et les lots. En outre, ils fournissent une émulation de programmes fournis par le système d'exploitation et de programmes « pilotes » spécialisés dont dépendent les applications existantes pour accéder à des services tels que IMS DB ou les boîtes de dialogue utilisateur via MFS.

18. Comment les programmes sont-ils enregistrés et gérés dans AWS Blu Age Runtime ?

    Les programmes de AWS Blu Age Runtime sont enregistrés via un ProgramRegistry système qui se remplit au démarrage du serveur. Chaque programme implémente l'interface du programme et est marqué comme un composant Spring. Les programmes sont enregistrés à l'aide de leurs identifiants, plusieurs entrées étant possibles si un programme possède plusieurs identifiants. Le processus d'enregistrement est automatique et enregistré dans les journaux Tomcat. ProgramRegistryCela permet à d'autres programmes et scripts de localiser et d'appeler des

programmes enregistrés, tout en préservant la modularité et l'interconnectivité du système modernisé.

19. Comment la configuration est-elle gérée dans les applications AWS Blu Age Runtime ?

    La configuration dans AWS Blu Age Runtime est gérée via des fichiers YAML à l'aide des fonctionnalités du framework Spring Boot. Deux fichiers de configuration principaux sont utilisés : application-main.yml pour la configuration du framework et pour les options spécifiques au client. `application-profile.yml` Le système suit la logique de priorité de Spring, permettant de remplacer la configuration par divers moyens. Une configuration supplémentaire peut être fournie via JNDI pour les bases de données et les paramètres de ligne de commande, offrant ainsi une flexibilité dans la gestion de la configuration. La configuration des enregistreurs est effectuée à l'aide des fichiers de configuration XML de journalisation.

20. Quel est le rôle des gestionnaires de secrets dans la configuration de AWS Blu Age Runtime ?

    Les gestionnaires de secrets de AWS Blu Age Runtime sécurisent les données de configuration sensibles telles que les informations d'identification des bases de données et les mots de passe du cache Redis. Ils permettent de stocker des données critiques en AWS secret et de les référencer dans des fichiers de configuration YAML. Le système prend en charge différents types de secrets, notamment les secrets de base de données qui remplissent automatiquement tous les champs pertinents et les secrets à mot de passe unique pour les ressources protégées par mot de passe. Cette approche améliore la sécurité en séparant les données sensibles de la configuration des applications.

21. Comment les développeurs peuvent-ils écrire leurs propres programmes compatibles avec AWS Blu Age Runtime ?

    Les développeurs peuvent créer des programmes compatibles avec AWS Blu Age Runtime en implémentant l'interface du programme et en suivant des modèles spécifiques. Le programme doit être déclaré en tant que composant Spring, implémenter les méthodes requises et être correctement enregistré dans le ProgramRegistry. Les développeurs doivent créer des classes de contexte et de configuration complémentaires, gérer les identificateurs de programme et garantir une intégration correcte avec le framework Spring. L'implémentation doit respecter les conventions de AWS Blu Age Runtime pour la structure et l'exécution du programme.

22. Comment AWS Blu Age Runtime gère-t-il les erreurs d'exécution de programmes ?

    AWS Blu Age Runtime gère les erreurs d'exécution de programmes par le biais de plusieurs mécanismes. Pour les tâches par lots, il capture l'état d'exécution, les codes de sortie et les informations d'erreur détaillées dans les détails de l'exécution des tâches. La gestion des erreurs

inclut des codes de sortie spécifiques (-1 pour les erreurs techniques, -2 pour les défaillances du programme de service) et une journalisation détaillée dans les journaux Tomcat. Le système peut être configuré pour annuler les transactions en cas d'exceptions d'exécution et fournit des options de notification d'erreur et de restauration. Les détails des erreurs sont accessibles via les points de terminaison REST à des fins de surveillance et de résolution des problèmes.

23. Quelles sont les fonctionnalités de surveillance de AWS Blu Age Runtime disponibles pour les tâches par lots ?

    AWS Blu Age Runtime fournit des fonctionnalités de surveillance pour les tâches par lots via différents points de terminaison. Il suit l'état d'exécution des tâches, les heures de début/fin, le mode d'exécution et les résultats détaillés. Le système propose des points de terminaison permettant de répertorier les scripts déclenchés, de récupérer les détails d'exécution des tâches et de surveiller les tâches en cours d'exécution. Les points de terminaison de Metrics fournissent des statistiques JVM, le nombre de sessions et des métriques détaillées d'exécution par lots. La plateforme prend également en charge la pagination et le filtrage temporel des données de surveillance.

24. Comment les statuts d'exécution des tâches AWS Blu Age Runtime sont-ils suivis et gérés ?

    Les statuts d'exécution des tâches sont suivis par le biais d'un système de statut complet qui inclut des états tels que DONE, TRIGGERED, RUNNING, KILLED et FAILED. Chaque exécution de tâche reçoit un identifiant unique pour le suivi et conserve des informations d'exécution détaillées, notamment l'heure de début, l'heure de fin, les informations sur l'appelant et les résultats de l'exécution. Le système fournit des points de terminaison REST pour demander l'état des tâches, gérer les tâches en cours et récupérer l'historique des exécutions. Les informations d'état sont conservées dans la mémoire du serveur et peuvent être supprimées en fonction de l'âge pour la gestion des ressources.

25. Comment AWS Blu Age Runtime gère-t-il les interactions avec le système externe ?

    Le moteur d'exécution gère les interactions avec le système externe par le biais de divers mécanismes, notamment les points de terminaison REST pour l'intégration des services, la prise en charge des files de messages (SQS, RabbitMQ, IBM MQ) et les options de connectivité aux bases de données. Il permet d'émuler les interactions entre les systèmes existants par le biais de composants spécialisés, prend en charge le protocole SSL/TLS pour des communications sécurisées et inclut des fonctionnalités de gestion de systèmes de fichiers externes. Le système prend également en charge l'intégration avec des fournisseurs d'authentification externes et peut être configuré pour interagir avec divers services tiers.

26. Comment l'authentification est-elle gérée dans AWS Blu Age Runtime ?

AWS Blu Age Runtime prend en charge plusieurs méthodes d'authentification, dont le principal mécanisme OAuth2 est le principal. Il peut s'intégrer à des fournisseurs d'identité tels qu'Amazon Cognito ou Keycloak. La configuration de l'authentification est gérée via le fichier de configuration principal appelé application-main.yml, dans lequel les paramètres de sécurité, les fournisseurs d'identité et les méthodes d'authentification peuvent être définis. Le système prend en charge des fonctionnalités telles que la protection XSS, CORS, CSRF, et peut être configuré à la fois pour la sécurité globale et pour la sécurité spécifique des terminaux. Pour le développement, un système d'authentification local avec les identifiants de super administrateur par défaut est également disponible.

27. Comment AWS Blu Age Runtime garantit-il une haute disponibilité ?

    AWS Blu Age Runtime garantit une haute disponibilité grâce à plusieurs mécanismes. Il implémente l'apatridie en stockant les états non transitoires dans un stockage partagé externe, ce qui permet à plusieurs instances d'applications de fonctionner ensemble. Le système prend en charge l'équilibrage de charge et les sessions partagées, ce qui permet de répartir les demandes sur plusieurs nœuds. Pour le stockage des données, il peut utiliser des bases de données hautement disponibles et des systèmes de mise en cache. L'architecture prend en charge le basculement automatique et peut être déployée sur plusieurs zones de disponibilité pour une fiabilité accrue.

28. Quel composant est utilisé pour reproduire les transactions distribuées CICS avec les applications AWS Blu Age ?

    Le AWS Blu Age Runtime fournit un point de terminaison dédié permettant d'invoquer les transactions JICS existantes dans le cadre d'une transaction globale (support XA). Le support de validation en deux phases sous-jacent repose sur le composant logiciel Atomikos.

29. Quel est le nom AWS Blu Age des classes utilisées pour définir le comportement spécifique d'un programme ?

    Chaque programme est lié à une classe de configuration dédiée qui permet de spécifier les comportements spécifiques au programme. Pour plus d'informations sur les conventions de dénomination et de localisation, consultez la structure AWS Blu Age de l'application modernisée

30. Quel encodage possède l'ordre de séquence de caractères suivant : espace, minuscules, majuscules, chiffres ?

    Jeux de caractères appartenant à la famille des variantes EBCDIC (tels que CP1 047, CP297 etc.).

31. Comment utilisez-vous AWS Blu Age Managed Runtime ?

Avec le AWS Management Console, le AWS CLI, ou le AWS APIs

32. Quelles sont les dimensions tarifaires de AWS Blu Age Runtime ?

    AWS Mainframe Modernization-core-hours (voir la tarification de la modernisation AWS du mainframe).

33. Quel est le mécanisme utilisé pour transmettre les données brutes via HTTP aux points de terminaison du programme ?

    Chaînes codées en Base64.

34. Comment un utilisateur lance-t-il un traitement par lots ?

    Utilisation d'un appel HTTP vers l'un des points de terminaison de traitement par lots dédiés (voir la page de documentation des points de terminaison par lots).

35. Quel point de terminaison AWS Blu Age Runtime est le principal point d'entrée de l'application frontale Web principale ?

    ```
    /transaction
    ```

36. Que signifie AWS Blu Age JICS ?

    Le AWS Blu Age JICS est le composant d'exécution utilisé pour soutenir la modernisation des ressources du CICS. Les définitions des ressources sont stockées dans un magasin de données dédié. Pour les administrer, utilisez l'API REST ou la console d'application JICS. Pour plus d'informations, voir Gérer la console d'applications JICS dans AWS Blu Age.

37. Quels sont les mécanismes de mise en cache de AWS Blu Age Runtime disponibles ?

    AWS Blu Age Runtime prend en charge plusieurs mécanismes de mise en cache, notamment Redis et. EhCache Redis est recommandé pour les environnements de production, car il fournit une mise en cache persistante partagée sur plusieurs nœuds. EhCache est disponible pour les déploiements autonomes avec mise en cache locale volatile intégrée. Le système prend en charge la mise en cache de divers composants, notamment les données Blusam, les informations de session, les ressources JICS et les files d'attente de stockage temporaires. La configuration du cache peut être personnalisée en fonction des différents cas d'utilisation et des exigences de performance.

38. Comment estimer le prix d'un déploiement d'un AWS Mainframe Modernization AWS Blu Age Runtime ?

AWS fournit des estimations aux clients en fonction de leurs besoins et de l'architecture cible.

39. Quel est le prix de la modernisation AWS du mainframe AWS Blu Age Runtime ?

    AWS La modernisation du mainframe propose deux modèles de tarification pour AWS Blu Age : une option Managed Runtime qui inclut le runtime, les ressources informatiques, le stockage interne et l'automatisation, et une option Runtime non managé qui couvre uniquement le runtime AWS Blu Age lui-même. Pour les AWS déploiements, les deux utilisent une structure pay-as-you-go tarifaire. Pour obtenir les informations tarifaires les plus complètes up-to-date et détaillées, il est recommandé de consulter la page officielle de tarification de la modernisation des mainframes AWS.

40. Et si nous devions déployer une application AWS refactorisée Blu Age sur une infrastructure non répertoriée dans le runtime pris en charge ?

    Si vous devez déployer une application AWS refactorisée Blu Age sur une infrastructure non répertoriée dans le runtime pris en charge, plusieurs options sont disponibles. Vérifiez d'abord si votre infrastructure est compatible avec les options de déploiement existantes telles qu'Amazon EKS Anywhere ou d'autres plateformes d'orchestration de conteneurs. Dans ce cas, vous pourrez peut-être utiliser le AWS Blu Age Runtime (non géré). Pour les infrastructures non compatibles, nous vous recommandons de consulter un spécialiste du AWS mainframe pour explorer des solutions personnalisées ou des adaptations potentielles. Vous pouvez également soumettre une demande de fonctionnalité du produit (PFR) pour un support d'infrastructure étendu. D'autres options de facturation peuvent être disponibles pour les déploiements non standard. Contactez votre AWS représentant pour discuter de vos besoins spécifiques et de la meilleure approche pour votre environnement.

41. Quelle est la licence du AWS Blu Age Runtime ? Est-ce open source ?

    AWS Blu Age Runtime n'est pas open source. Il s'agit d'une AWS adresse IP distribuée en tant que service cloud natif. Il existe deux options de déploiement :

    a. AWS Blu Age Managed, le runtime est déployé dans un service AWS géré dédié, tirant parti d'un environnement entièrement préconfiguré et prêt à être déployé, sans configuration ni administration.

    b. AWS Blu Age Non Managed, qui peut être déployé dans votre propre AWS architecture sur mesure basée sur Amazon EC2 ou Amazon ECS/AWS Fargate, que vous devez approvisionner et configurer vous-même. Les deux options entraînent des frais d'exécution, qui sont inclus dans les estimations de projet qui vous sont fournies. Comme il s'agit d'un service géré avec

Support accès, vous n'avez pas besoin du code source. Pour plus de détails sur les tarifs, consultez la page de tarification de la modernisation des AWS mainframes.

42. Comment sont gérées les modifications et les mises à niveau des frameworks et bibliothèques AWS Blu Age ?

    AWS Les frameworks et bibliothèques Blu Age sont mis à jour par le biais de processus réguliers de génération et de déploiement de code. Ces mises à jour sont gérées dans le cadre du cycle de vie de modernisation du AWS mainframe, qui inclut les mises à niveau des versions et le support de l'équipe AWS Blu Age ou de partenaires certifiés. Pour obtenir des informations détaillées sur le contrôle des versions, les processus de mise à niveau et les délais de support, reportez-vous à la documentation du cycle de vie de modernisation des AWS mainframes.

Données

1. Quelles sont les options de base de données disponibles pour les applications modernisées, en ce qui concerne la modernisation de l'ancienne base de données ?

   Les applications modernisées peuvent utiliser plusieurs options de base de données modernes, notamment : PostgreSQL, Amazon Aurora, RDS pour PostgreSQL, base de données Oracle, MS-SQL et IBM Db2. Ces options offrent la flexibilité nécessaire pour choisir le système de base de données le plus approprié en fonction des exigences spécifiques, tout en tirant parti des avantages des systèmes de gestion de base de données modernes et des fonctionnalités natives du cloud.

2. Quelle est la couverture de transformation d'IBM Db2 for ? z/OS vers Postgres DDL ?

   Transformation complète (y compris les contraintes de base de données).

3. AWS Blu Age prend-il en charge la génération de données de groupe (GDG) ?

   Oui, l'utilisation du GDG par lots est prise en charge, avec le support des générations relatives et absolues et des stratégies de nettoyage automatique.

4. AWS Blu Age prend-il en charge les ensembles de données concaténés ?

   Oui, l'utilisation d'ensembles de données concaténés par lots est prise en charge. Grâce à la concaténation, plusieurs ensembles de données peuvent être lus comme un seul ensemble de données. Veuillez noter que les ensembles de données Blusam ne peuvent pas faire partie d'une concaténation.

5. Quel est le processus appliqué aux requêtes SQL ?

Ajusté lors de la transformation du code, en fonction de la base de données cible.

6. Quelles options s'appliquent s'il existe plusieurs bases de données pour une application ?

   Configurez la base de données cible pour chaque requête et définissez toutes les bases de données de l'application et d'Apache Tomcat.

7. Est-ce que Blusam peut être désactivé ?

   Oui, dans le fichier de configuration principal, et aucune base de données n'est requise (pour plus d'informations, consultez la page de documentation de configuration de Blusam).

8. Quelle API AWS Blu Age est utilisée pour remplacer des bases de données telles que IMS DB ?

   L'API JHDB (Java Hierarchical DataBase).

9. Quel produit AWS Blu Age peut être utilisé pour migrer des données et des bases de données existantes vers un système de gestion de base de données relationnelle (RDBMS) moderne ?

   AWS Outil de modernisation de base de données Blu Age (Data Migrator).

10. Qu'est-ce que AWS Blu Age Data Simplifier et quel problème résout-il dans le cadre de la modernisation ?

    Data Simplifier est une bibliothèque essentielle de AWS Blu Age qui permet de relever le défi de gérer les anciens modèles d'accès à la mémoire en Java. Il fournit des structures prenant en charge l'accès à la mémoire de bas niveau, les types de données existants (tels que les données zonées, compressées, alphanumériques) et les données mixtes structured/raw memory access that are common in mainframe applications but not natively available in Java. The library exposes these features through familiar Java patterns like getters/setters et basées sur des classes APIs, les rendant accessibles aux développeurs Java tout en conservant les fonctionnalités existantes.

11. Comment AWS Blu Age gère-t-il les configurations de mémoire et les structures de données existantes ?

    AWS Blu Age gère les anciennes configurations de mémoire via l'interface Record, qui fournit une abstraction de tableaux d'octets de taille fixe. Pour les données structurées telles que les « éléments de données COBOL 01 », il utilise des RecordEntitysous-classes qui sont automatiquement générées lors de la modernisation. Ces classes conservent la structure hiérarchique des données existantes, chaque élément ayant une relation parent-enfant. Le système prend en charge à la fois l'accès à la mémoire brute et les modèles d'accès structurés, préservant ainsi la flexibilité des systèmes existants tout en fournissant une interface de programmation moderne.

12. Comment AWS Blu Age gère-t-il la modernisation des ensembles de données VSAM ?

    Le composant Blusam prend en charge la modernisation des ensembles de données VSAM, avec une API dédiée, des points de terminaison et une application Web d'administration (BAC : Blusam Administration Console). Blusam s'appuie sur une base de données relationnelle comme backend (PostgreSQL, utilisant RDS ou Aurora).

Transformation

1. Où puis-je trouver des informations sur le processus de transformation ?

   Consultez la documentation AWS Blu Insights.

2. Quels sont les noms des modules générés par AWS Blu Age ?

   Service, entités, Web et outils.

3. Pourquoi Java/Spring a-t-il été choisi comme l'une des technologies cibles de AWS Blu Age ?

   Java/Spring a été choisie comme technologie cible en raison de son adoption généralisée, de son vaste vivier de talents et de ses solides capacités d'entreprise. L'écosystème Java propose de vastes bibliothèques, frameworks et outils qui prennent en charge le développement d'applications modernes. Le framework Spring fournit des fonctionnalités de niveau entreprise, des fonctionnalités natives du cloud et suit les meilleures pratiques du secteur, ce qui le rend idéal pour les applications modernisées.

4. Quel est le nom du projet parent qui contient les modules générés par AWS Blu Age ?

   Le nom du projet parent est suffixé par « -pom » et peut être défini dans le centre de transformation à l'aide de la propriété Transform nommée project.

5. Comment AWS Blu Age gère-t-elle la modernisation des anciens planificateurs, s'ils sont fournis ?

   Les anciens actifs du planificateur ne sont pas modernisés par AWS Blu Age. Ils sont pris en compte lors de la phase d'évaluation, afin d'aider à identifier les éventuels artefacts manquants.

6. Quelles sont les conditions requises pour déboguer le code généré avec AWS Blu Age ?

   Tout environnement de développement intégré (IDE) prenant en charge Java, tel qu'Eclipse JetBrain, ou VisualCode.

Déploiement

1. Quels sont les environnements disponibles pour déployer l'application modernisée avec AWS Blu Age ?

   Windows Server, serveur Linux et conteneur Docker Linux.

2. Les applications refactorisées de AWS Blu Age peuvent-elles fonctionner sur n'importe quelle infrastructure ?

   Bien que les applications refactorisées AWS Blu Age ne soient conçues pour fonctionner sur aucune infrastructure, elles offrent une flexibilité significative dans les options de déploiement. Ces applications peuvent être déployées sur différentes plateformes informatiques, notamment les services gérés dans le cloud, le calcul sans serveur et l'infrastructure sur site. AWS Blu Age propose des options d'exécution gérées et non gérées, permettant aux entreprises de choisir entre une commodité entièrement gérée et un contrôle personnalisé en fonction de leurs besoins et exigences spécifiques. Cette flexibilité facilite le déplacement entre les infrastructures prises en charge, ce qui rend les applications refactorisées AWS Blu Age hautement adaptables aux différents environnements de déploiement. Pour plus de détails, consultez la documentation sur les options d'AWS Blu Age Runtime.

3. Quelles sont les configurations MQ prises en charge par AWS Blu Age ?

   SQS, IBM WebSphere MQ.

4. Sur quels serveurs d'applications un utilisateur peut-il déployer la logique d'application métier Java avec un environnement d'exécution non géré AWS Mainframe Modernization ?

   Apache Tomcat, version supérieure ou égale à 10.1.

5. Comment l'application refactorisée s'intègre-t-elle à d'autres applications telles qu'Amazon Services AWS Aurora ?

   L'application modernisée s'intègre Services AWS en prenant en charge la transformation vers des solutions de base de données natives dans le cloud telles qu'Amazon Aurora et RDS pour PostgreSQL. AWS Blu Age garantit l'intégration entre les applications modernisées et Services AWS permet aux entreprises d'utiliser les fonctionnalités du cloud. Cette intégration s'étend au stockage des données et aux services applicatifs au sein de l' AWS écosystème. Au-delà du stockage de base de données, AWS Blu Age Runtime s'intègre à divers outils, Services AWS notamment Amazon ElastiCache pour la mise en cache de Redis, AWS Secrets Manager pour la gestion de la configuration, et la modernisation du AWS mainframe pour le déploiement. Il

prend en charge Amazon EC2, Amazon EKS, ECS gérés par Fargate pour le déploiement de conteneurs. Le système peut utiliser AWS Identity and Access Management pour l'authentification Amazon Simple Storage Service pour le stockage et prend en charge l'intégration avec d'autres Services AWS via des connecteurs de configuration et de service.

6. Comment l'application refactorisée garantit-elle le respect des exigences d'évolutivité ?

   La solution garantit l'évolutivité en transformant les applications en architectures cloud natives capables d'utiliser l'infrastructure AWS élastique. Il met en œuvre des modèles de conception modernes et les meilleures pratiques qui permettent une mise à l'échelle horizontale et verticale. L'approche axée sur les services permet une mise à l'échelle indépendante des composants. Les applications modernisées peuvent tirer parti des fonctionnalités d'évolutivité inhérentes aux services cloud.

7. Que se passe-t-il une fois la refactorisation du code source terminée ?

   Après la refactorisation du code source, deux étapes principales se produisent. Tout d'abord, l'application refactorisée est créée. Ensuite, l'application est déployée et surveillée dans AWS Mainframe Modernization AWS Blu Age Runtime. Le déploiement peut être effectué soit dans un environnement AWS géré (AWS Mainframe Modernization Managed Runtime) dans lequel l'infrastructure est gérée de manière automatisée, soit dans votre environnement Compte AWS (AWS Mainframe Modernization AWS Blu Age, environnement d'exécution non géré) dans lequel les clients gèrent leur propre infrastructure. L'option non gérée peut être déployée sur différentes plateformes, notamment Amazon EC2, ECS sur EC2 ou sur Fargate, EKS sur. EC2

8. Comment puis-je déployer et exécuter une application modernisée avec AWS Blu Age sur une AMI Amazon Linux personnalisée, sans utiliser le service géré AWS Mainframe Modernization (M2) ?

   Cela peut être réalisé en déployant l'application à l'aide de AWS Blu Age Runtime (non géré) sur Amazon EC2. Le processus consiste à créer une application Java/Spring dépendante de la bibliothèque AWS Blu Age Runtime et à la déployer sur une AMI Amazon Linux personnalisée. Pour obtenir des instructions détaillées sur cette approche, consultez Configurer AWS Blu Age Runtime (non géré) sur Amazon EC2.

9. Une Amazon Machine Image (AMI) est-elle disponible ? Une image Docker est-elle disponible ?

   - AMI : Non, les clients ayant besoin de personnaliser et de configurer leur environnement comme ils le souhaitent, aucune AMI n'est disponible. Les clients peuvent récupérer les artefacts AWS Blu Age et configurer leur instance en fonction de leurs besoins.
   - Image Docker : Non, aucune image docker n'est disponible, mais la page AWS Configurer Blu Age Runtime sur un conteneur explique comment créer et déployer votre propre image

docker basée sur les fichiers binaires de Blu Age Runtime, AWS sur un système de gestion de conteneurs adapté.

10. Le client peut-il empaqueter et exécuter une application AWS Blu Age en tant que conteneur Docker ?

    Cela n'est pas possible pour M2 Managed Runtime, mais c'est le cas pour un environnement défini par le client basé sur une AMI Amazon Linux et pour les fournisseurs sur site ou d'autres fournisseurs de cloud.

11. Comment puis-je connaître l'ARN de la ressource de la politique SQS requise pour exécuter AWS Blu Age en mode non géré si je souhaite la réduire ?

    Pour déterminer l'ARN de la ressource de politique SQS spécifique pour exécuter AWS Blu Age de manière non gérée avec une politique délimitée, veuillez consulter l'équipe de mise en œuvre ou le responsable de compte technique (TAM). Ils peuvent fournir des conseils spécifiques au compte. Pour obtenir des informations générales sur les politiques SQS, reportez-vous à la documentation relative aux politiques AWS SQS.

12. Comment fonctionne la planification des tâches avec le traitement par lots ?

    Il est intégré à la branche Control-M/Stone ou à tout autre planificateur distribué.

Replate-forme d'applications avec Rocket Software (anciennement Micro Focus)

Ce guide décrit le end-to-end processus de replate-forme des applications mainframe à l'aide des solutions de modernisation des AWS mainframes sur. AWS Il décrit toutes les tâches et inclut des informations sur la configuration et le fonctionnement du moteur d'exécution AWS Mainframe Modernization sur Amazon, EC2 depuis la configuration initiale et l'analyse jusqu'à la création, au test et au déploiement de vos applications modernisées sur. AWS Il couvre également des sujets avancés tels que l'utilisation de structures de données existantes, l'utilisation de modèles et de projets prédéfinis, et la configuration de l'automatisation pour les sessions de streaming.

Rubriques

- Configurer Rocket Software (anciennement Micro Focus) (sur Amazon EC2)
- Configurer l'automatisation des sessions de streaming pour Rocket Enterprise Analyzer (anciennement Micro Focus) et Rocket Enterprise Developer
- Afficher les ensembles de données sous forme de tables et de colonnes dans Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)
- Modifiez des ensembles de données à l'aide des outils de fichiers de données Rocket Software (anciennement Micro Focus) dans Enterprise Developer
- Tutoriels pour Rocket Software (anciennement Micro Focus)
- Utilitaires de traitement par lots disponibles dans AWS Mainframe Modernization

Configurer Rocket Software (anciennement Micro Focus) (sur Amazon EC2)

AWS Mainframe Modernization fournit plusieurs Amazon Machine Images (AMIs) qui incluent des produits sous licence Rocket Software (anciennement Micro Focus). Ils vous AMIs permettent de configurer rapidement des instances Amazon Elastic Compute Cloud (Amazon EC2) pour prendre en charge les environnements Rocket Software que vous contrôlez et gérez. Cette rubrique décrit les étapes nécessaires pour y accéder et les lancer AMIs. Leur utilisation AMIs est entièrement facultative et il n'est pas nécessaire de les utiliser pour suivre les didacticiels de ce guide de l'utilisateur.

Rubriques

- Conditions préalables à la configuration de Rocket Software (anciennement Micro Focus) (sur Amazon EC2)
- Création du point de terminaison Amazon VPC pour Amazon S3
- Demander la mise à jour de la liste d'autorisation pour le compte
- Créez le AWS Identity and Access Management rôle
- Accordez à License Manager les autorisations requises
- Abonnez-vous aux Amazon Machine Images
- Lancer une instance de AWS Mainframe Modernization Rocket Software (anciennement Micro Focus)
- Sous-réseau ou VPC sans accès à Internet

Conditions préalables à la configuration de Rocket Software (anciennement Micro Focus) (sur Amazon EC2)

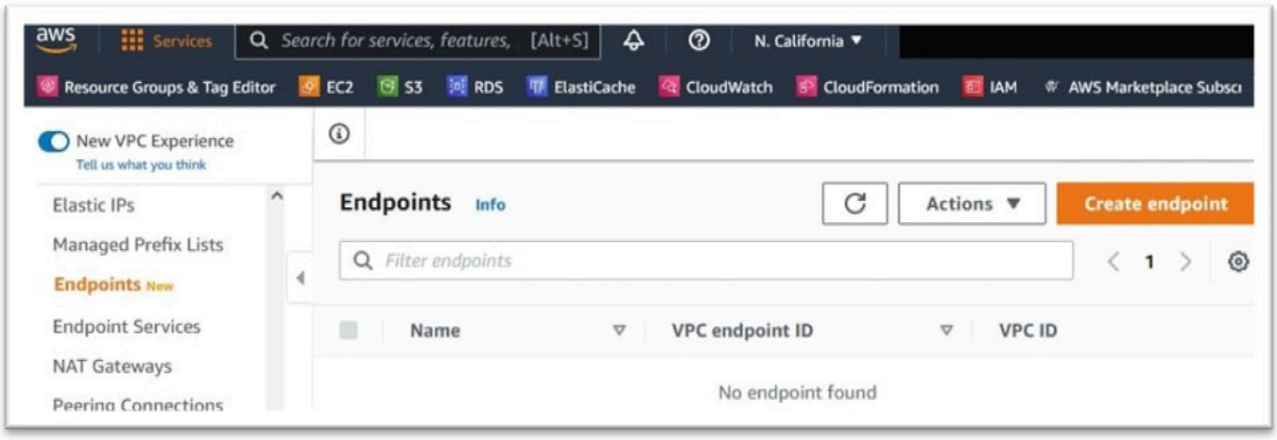
Lorsque vous configurez Rocket Software (sur Amazon EC2), assurez-vous de remplir les conditions préalables suivantes.

- Accès administrateur au compte sur lequel les EC2 instances Amazon seront créées.
- Identifiez l' Région AWS endroit où les EC2 instances Amazon seront créées et vérifiez que le AWS Mainframe Modernization service est disponible. Consultez la section AWS Services par région. Assurez-vous de choisir une région dans laquelle le service est disponible.
- Identifiez l'Amazon Virtual Private Cloud (Amazon VPC) dans lequel les EC2 instances Amazon seront créées.

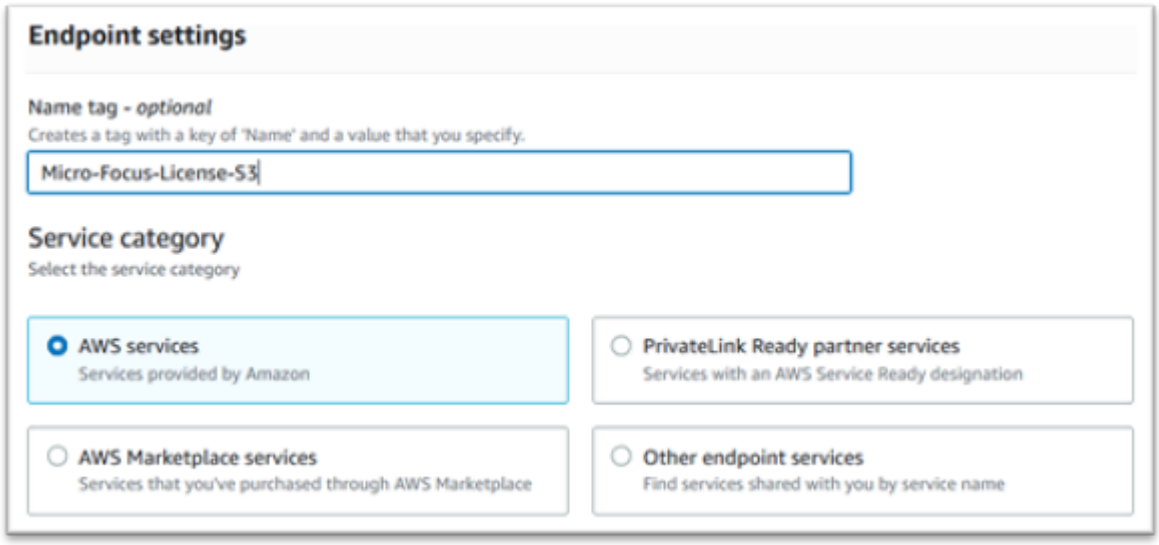
Création du point de terminaison Amazon VPC pour Amazon S3

Dans cette section, vous allez créer un point de terminaison Amazon VPC à utiliser par Amazon S3. La configuration de ce point de terminaison vous sera utile ultérieurement lors de la configuration de l'accès Internet pour VPC.

1. Accédez à Amazon VPC dans le. AWS Management Console
2. Dans le volet de navigation, choisissez Points de terminaison.
3. Choisissez Créer un point de terminaison.

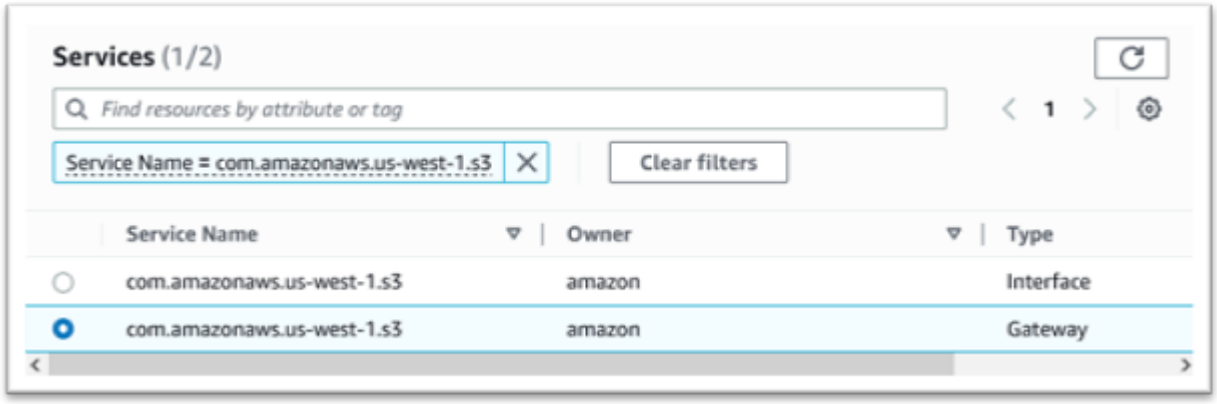


4. Entrez un tag nominatif significatif, par exemple : « Micro-Focus-License-S3 ».
5. Choisissez AWS Services comme catégorie de service.

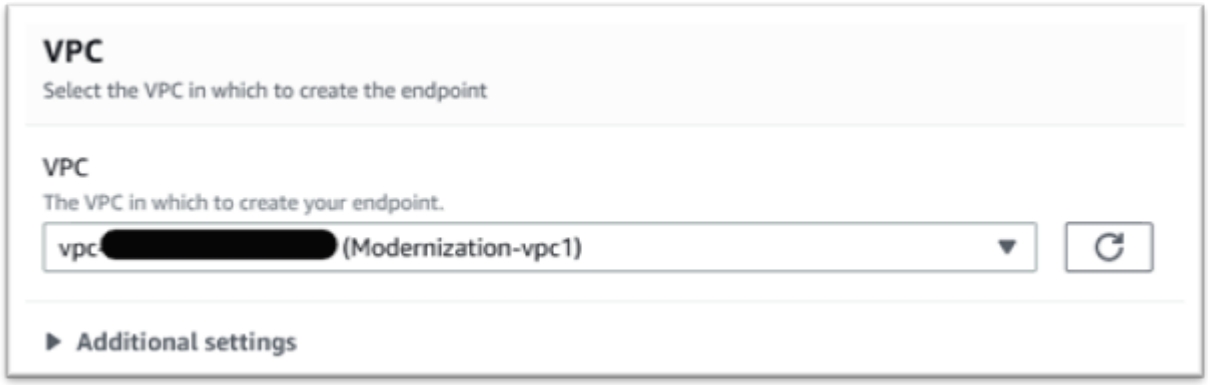


6. Sous Services, recherchez le service Amazon S3 Gateway : com.amazonaws. [région] .s3.

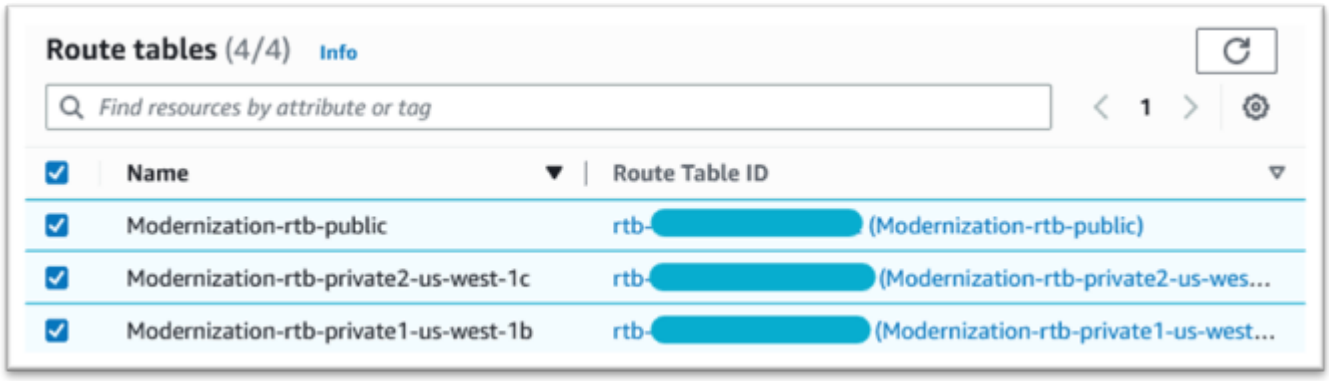
   Car `us-west-1` ce serait :`com.amazonaws.us-west-1.s3`.
7. Choisissez le service Gateway.



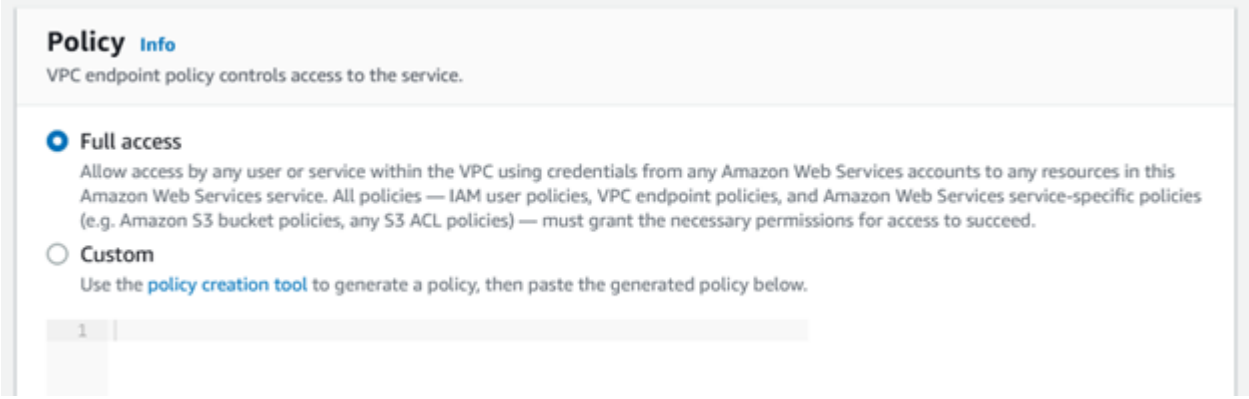
8. Pour VPC, choisissez le VPC que vous utiliserez.



9. Choisissez toutes les tables de routage pour le VPC.



10. Sous Politique, sélectionnez Accès complet.



> **Note**
>
> Si vous décidez de créer une politique personnalisée, assurez-vous qu'elle a accès au compartiment Amazon S3`s3://aws-supernova-marketplace-<region>-prod`.

11. Choisissez Créer un point de terminaison.

Demander la mise à jour de la liste d'autorisation pour le compte

Contactez votre AWS représentant pour que votre compte soit autorisé pour le AWS Mainframe Modernization AMIs. Merci de fournir les informations suivantes :

- La Compte AWS pièce d'identité.
- L' Région AWS endroit où le point de terminaison Amazon VPC a été créé.
- L'identifiant du point de terminaison Amazon VPC Amazon S3 créé dans. Création du point de terminaison Amazon VPC pour Amazon S3 Il s'agit de l'`vpce-xxxxxxxxxxxxxxxxx`identifiant du fichier com.amazonaws. Point de terminaison de la passerelle [region] .s3.
- Le nombre de licences requises pour toutes les EC2 instances Amazon de l'AMI Rocket Software Enterprise Suite.

  Une licence est requise par cœur de processeur (par 2 V CPUs pour la plupart des EC2 instances Amazon).

  Pour plus d'informations, voir Optimisation des options du processeur.

  Le numéro demandé peut être ajusté à l'avenir par AWS.

> **Note**
>
> Contactez votre AWS représentant ou AWS Support qui ouvrira le ticket d'assistance pour la demande de liste d'autorisation en votre nom. Vous ne pouvez pas le demander directement et le traitement de la demande peut prendre plusieurs jours.

Créez le AWS Identity and Access Management rôle

Créez une AWS Identity and Access Management politique et un rôle à utiliser par les EC2 instances AWS Mainframe Modernization Amazon. La création du rôle via la console IAM créera un profil d'instance associé du même nom. L'attribution de ce profil d'instance aux EC2 instances Amazon permet d'attribuer des licences logicielles Rocket. Pour plus d'informations sur les profils d'instance, consultez Utiliser un rôle IAM pour accorder des autorisations aux applications exécutées sur des EC2 instances Amazon.

Créer une politique IAM

Une politique IAM est d'abord créée, puis attachée au rôle.

1. Naviguez vers AWS Identity and Access Management dans le AWS Management Console.
2. Choisissez Politiques, puis Créer une politique.

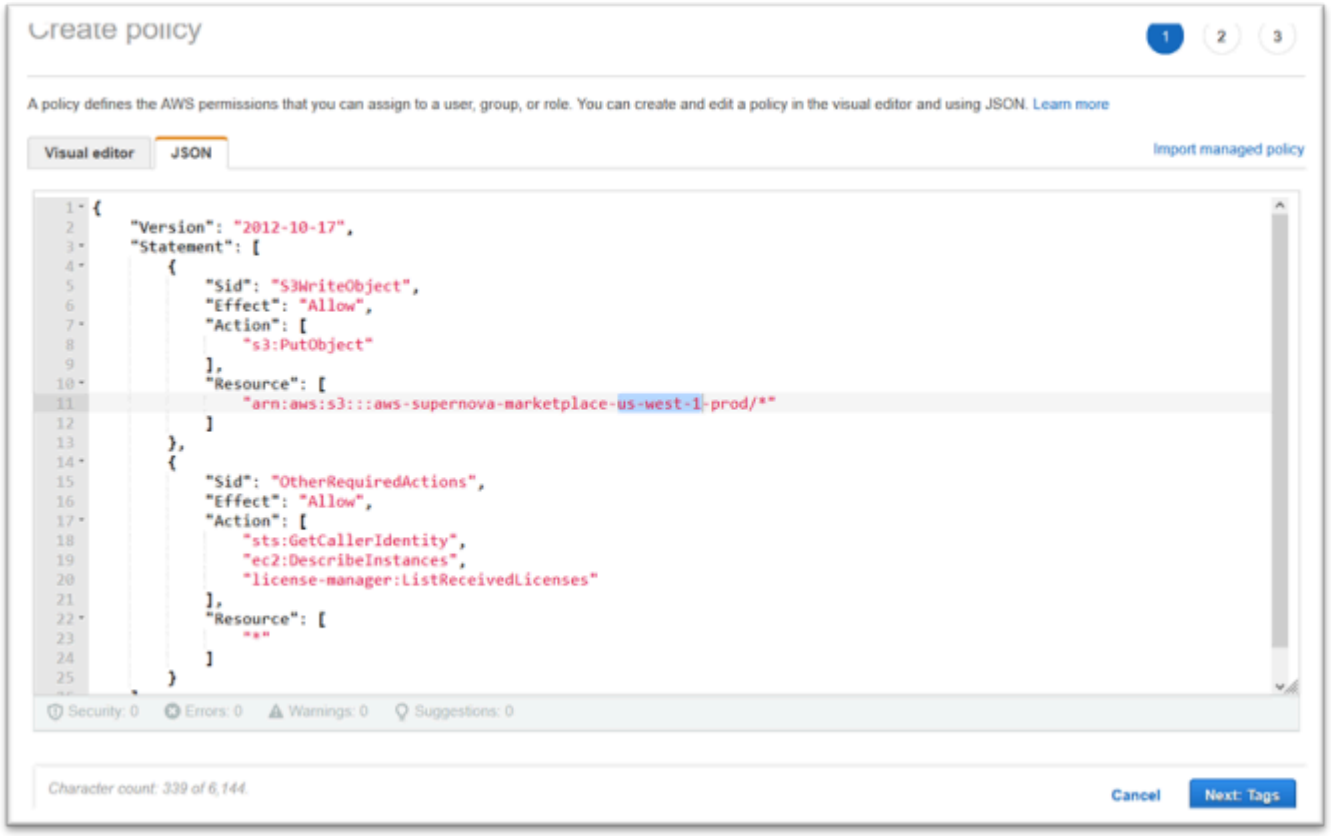
3. Choisissez l'onglet JSON.

4. Remplacez `us-west-1` le JSON suivant par celui Région AWS où le point de terminaison Amazon S3 a été défini, puis copiez-collez le JSON dans l'éditeur de politiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3WriteObject",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-supernova-marketplace-us-west-1-prod/*"
            ]
        },
        {
            "Sid": "OtherRequiredActions",
            "Effect": "Allow",
            "Action": [
                "sts:GetCallerIdentity",
                "ec2:DescribeInstances",
                "license-manager:ListReceivedLicenses"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
```

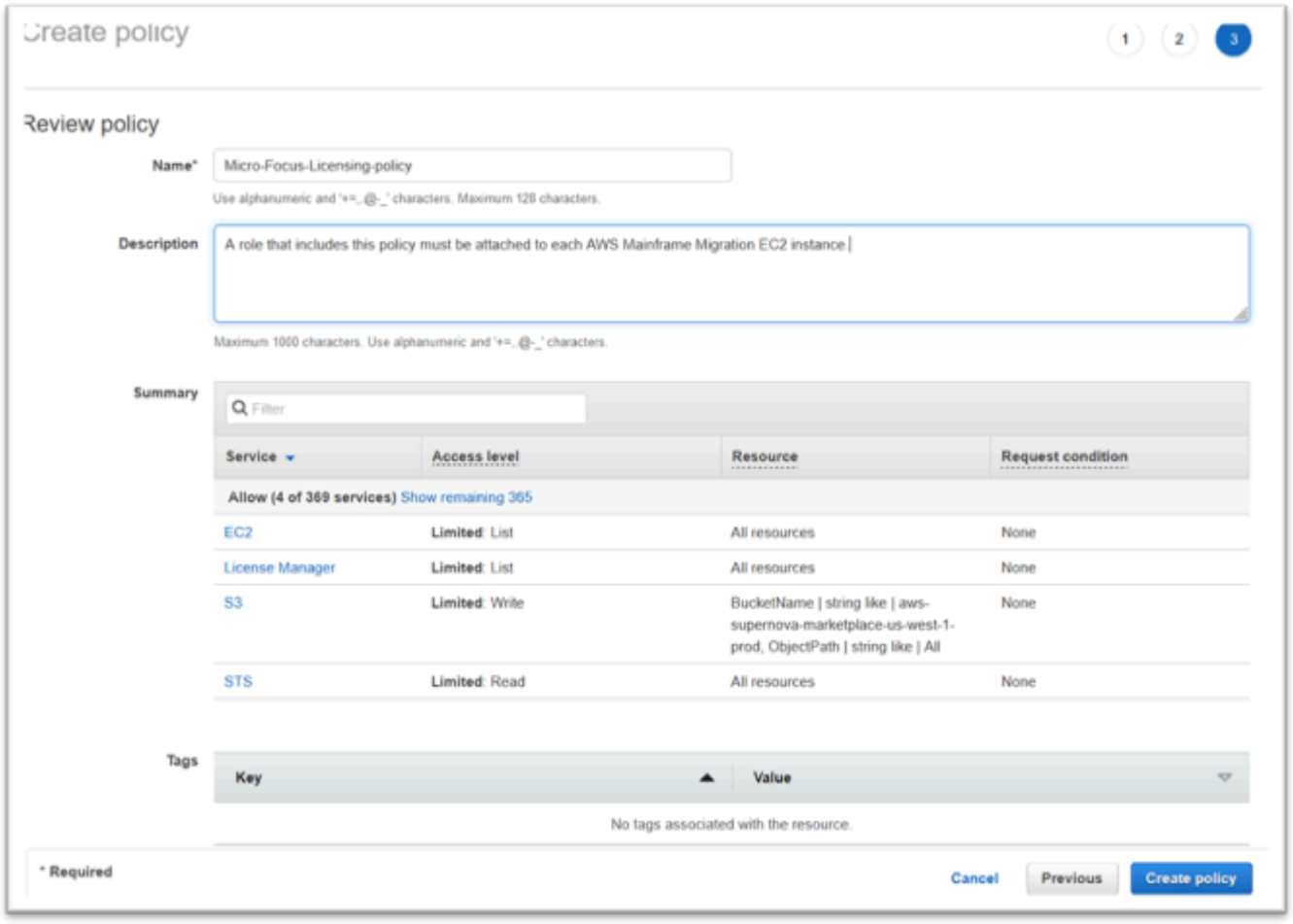
```
}
```

> **Note**
>
> Les actions situées sous le Sid `OtherRequiredActions` ne prennent pas en charge les autorisations au niveau des ressources et doivent être spécifiées `*` dans l'élément ressource.



5. Choisissez Suivant : Balises.

6. Entrez éventuellement des balises, puis choisissez Next : Review.
7. Entrez un nom pour la politique, par exemple « Micro-Focus-Licensing-Policy ». Entrez éventuellement une description, par exemple « Un rôle incluant cette politique doit être attaché à chaque EC2 instance AWS Mainframe Modernization Amazon ».

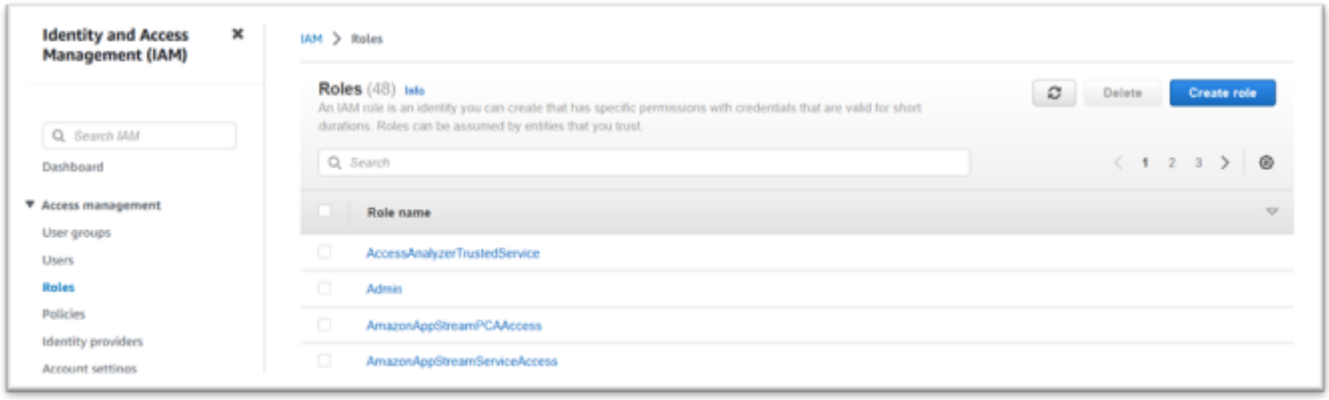


8. Choisissez Create Policy (Créer une politique).

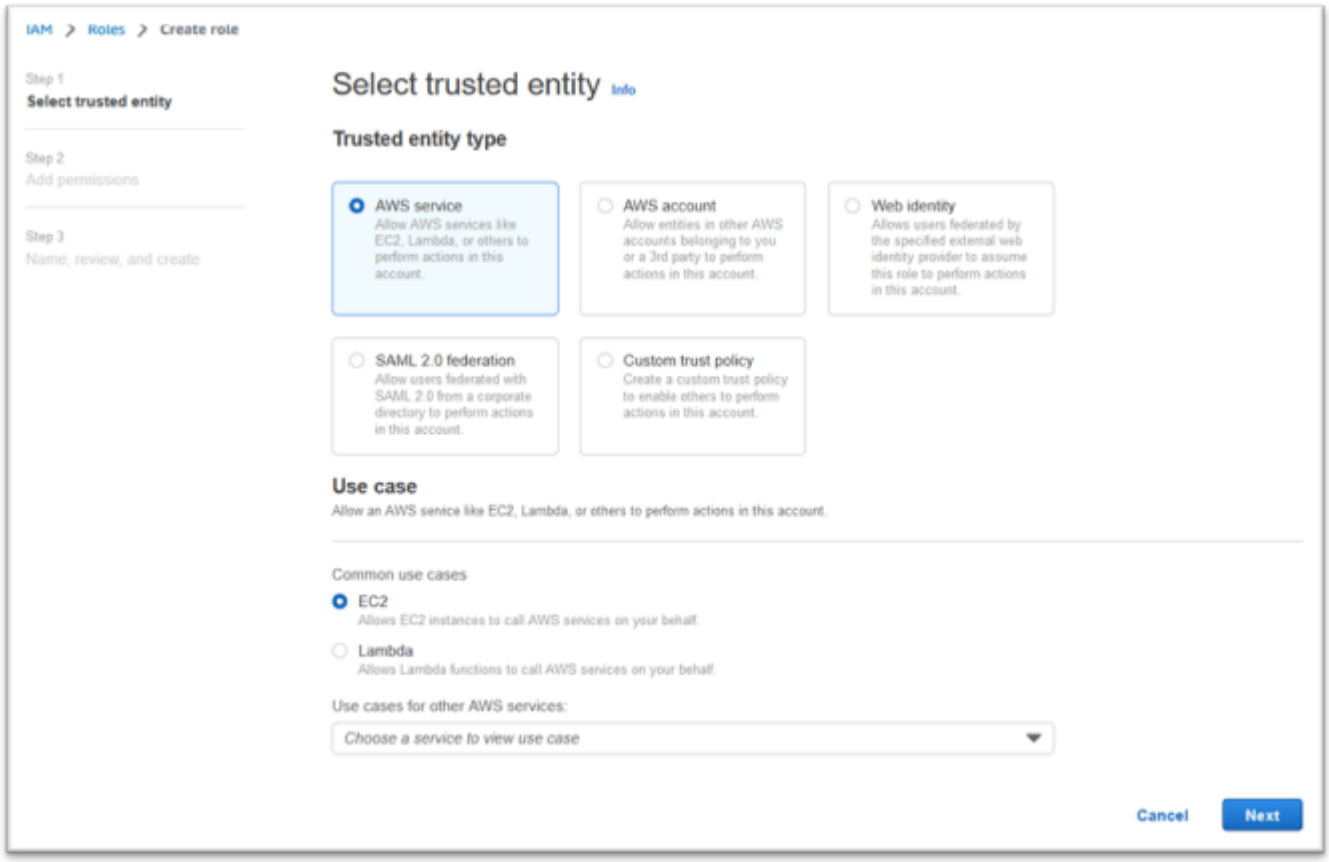
Création du rôle IAM

Après avoir créé une stratégie IAM, vous créez un rôle IAM et vous l'associez à la stratégie.

1. Accédez à IAM dans le. AWS Management Console
2. Choisissez Rôles, puis Créer un rôle.

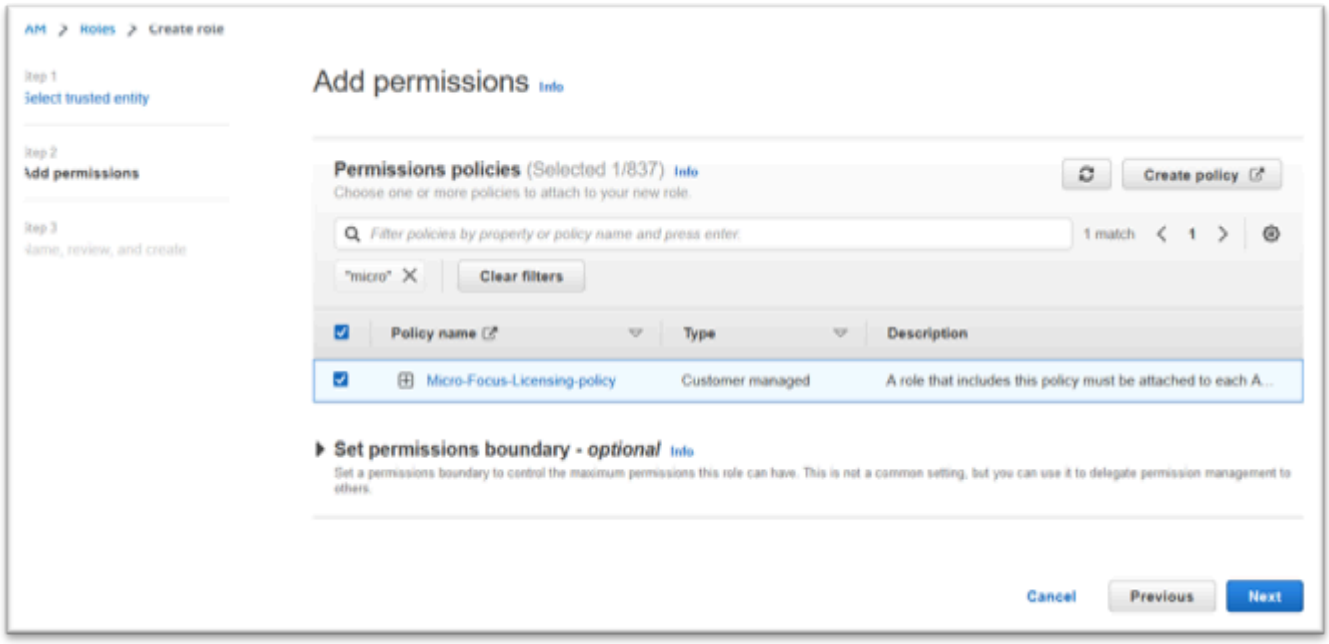


3. Laissez le type d'entité de confiance comme AWS service et choisissez le cas d'utilisation EC2courant.

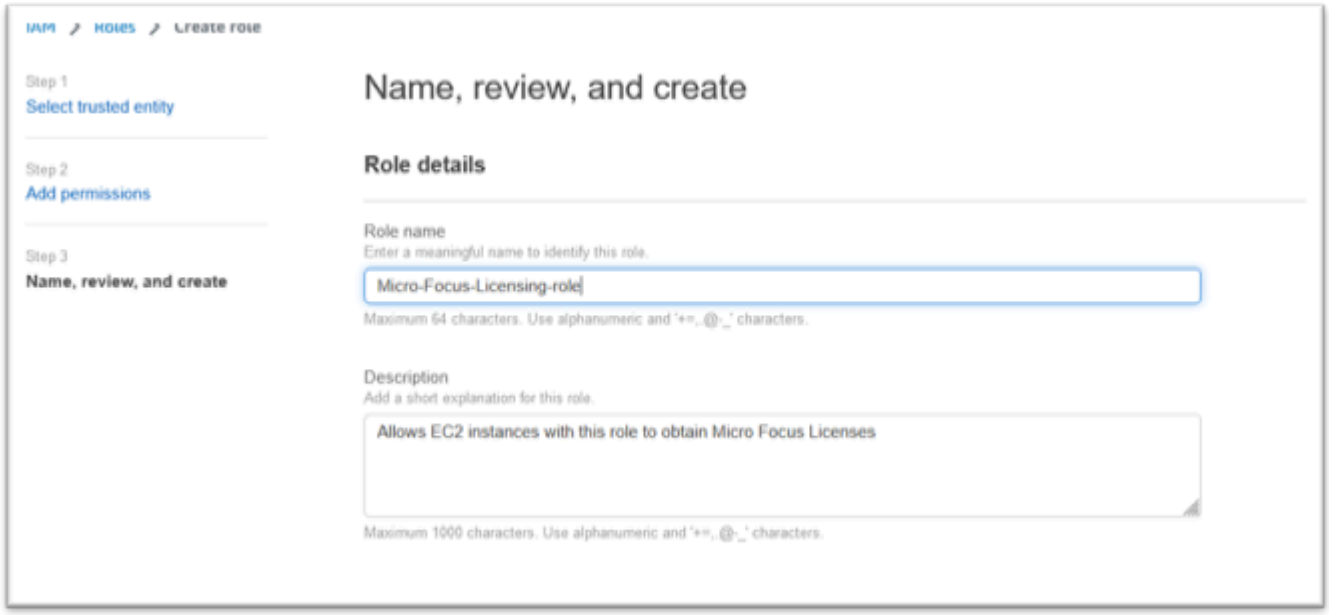


4. Choisissez Next (Suivant).

5. Entrez « Micro » dans le filtre et appuyez sur Entrée pour appliquer le filtre.
6. Choisissez la politique qui vient d'être créée, par exemple la « Micro-Focus-Licensing-Policy ».
7. Choisissez Next (Suivant).



8. Entrez le nom du rôle, par exemple « Micro-Focus-Licensing-Role ».
9. Remplacez la description par la vôtre, par exemple « Autorise les EC2 instances Amazon dotées de ce rôle à obtenir des licences Micro Focus ».

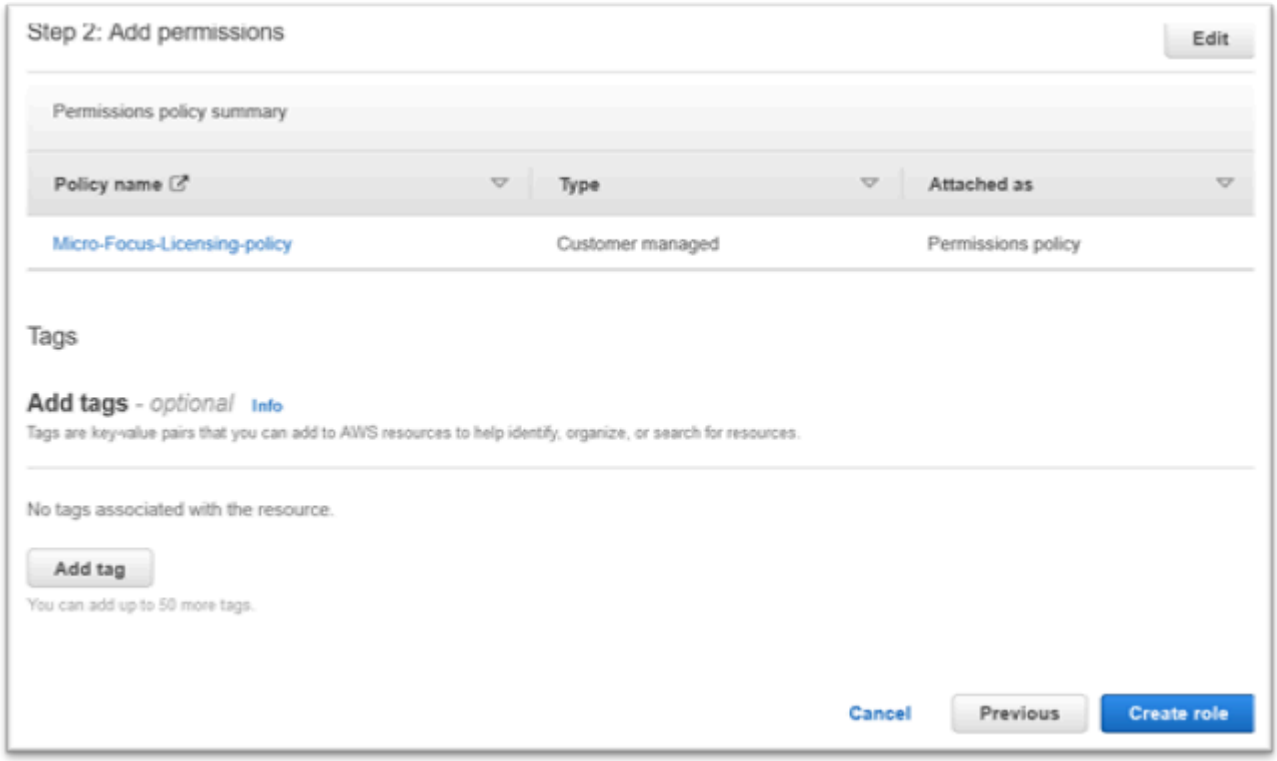


10. À l'étape 1 : Sélectionnez les entités de confiance, examinez le JSON et confirmez qu'il contient les valeurs suivantes :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
        }
    ]
}
```

> **Note**
>
> L'ordre de l'effet, de l'action et du principal n'est pas significatif.

11. Vérifiez que l'étape 2 : Ajouter des autorisations indique votre politique de licence.



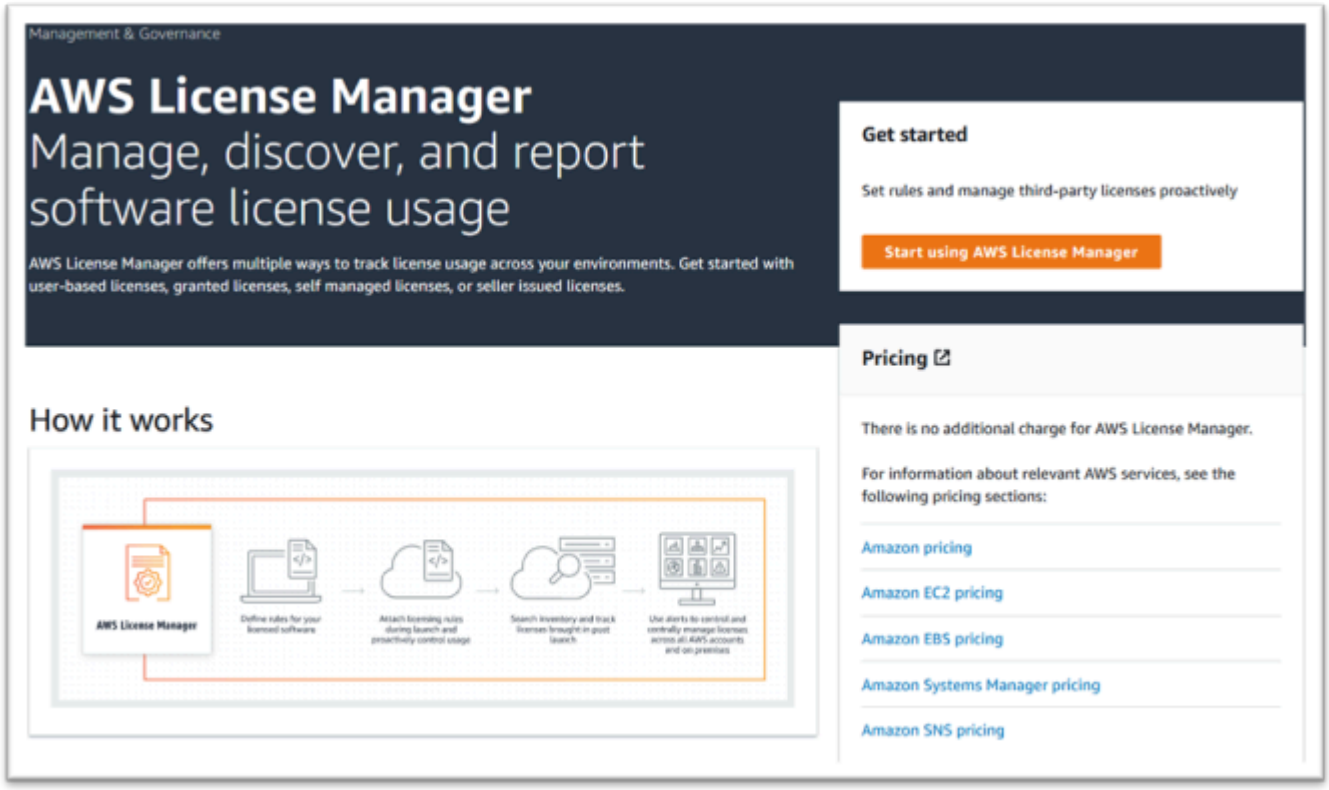
12. Sélectionnez Create role (Créer un rôle).

Une fois la demande de liste d'autorisation terminée, passez aux étapes suivantes.

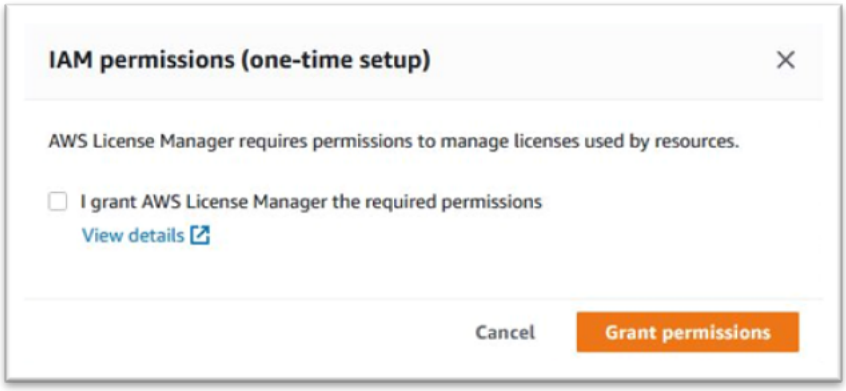
Accordez à License Manager les autorisations requises

Vous devez accorder des autorisations AWS License Manager pour configurer le moteur d'exécution Rocket Software (sur Amazon EC2).

1. Naviguez vers AWS License Manager dans le AWS Management Console.



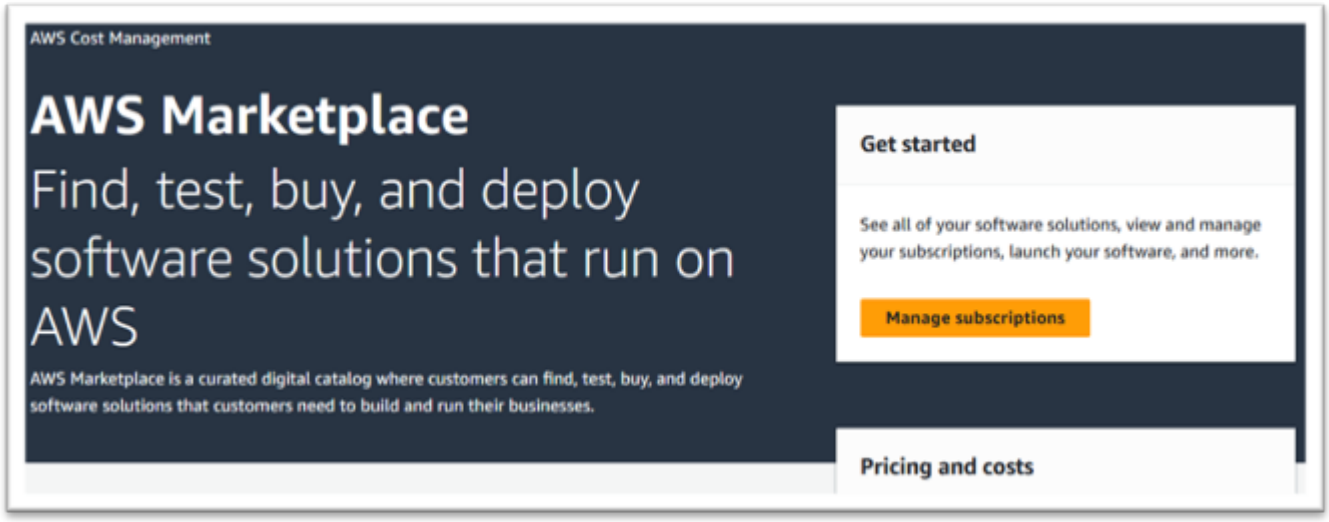
2. Choisissez Start using AWS License Manager.
3. Si la fenêtre contextuelle suivante s'affiche, affichez les détails, cochez la case et appuyez sur Accorder les autorisations.



Abonnez-vous aux Amazon Machine Images

Une fois que vous êtes abonné à un AWS Marketplace produit, vous pouvez lancer une instance depuis l'AMI du produit. Vous pouvez également gérer votre abonnement AMIs lors de la configuration du moteur d'exécution Rocket Software (anciennement Micro Focus) (sur Amazon EC2).

1. Accédez à AWS Marketplace Abonnements dans le AWS Management Console.
2. Sélectionnez Manage subscriptions (Gérer les abonnements).



3. Copiez et collez l'un des liens suivants dans la barre d'adresse du navigateur.

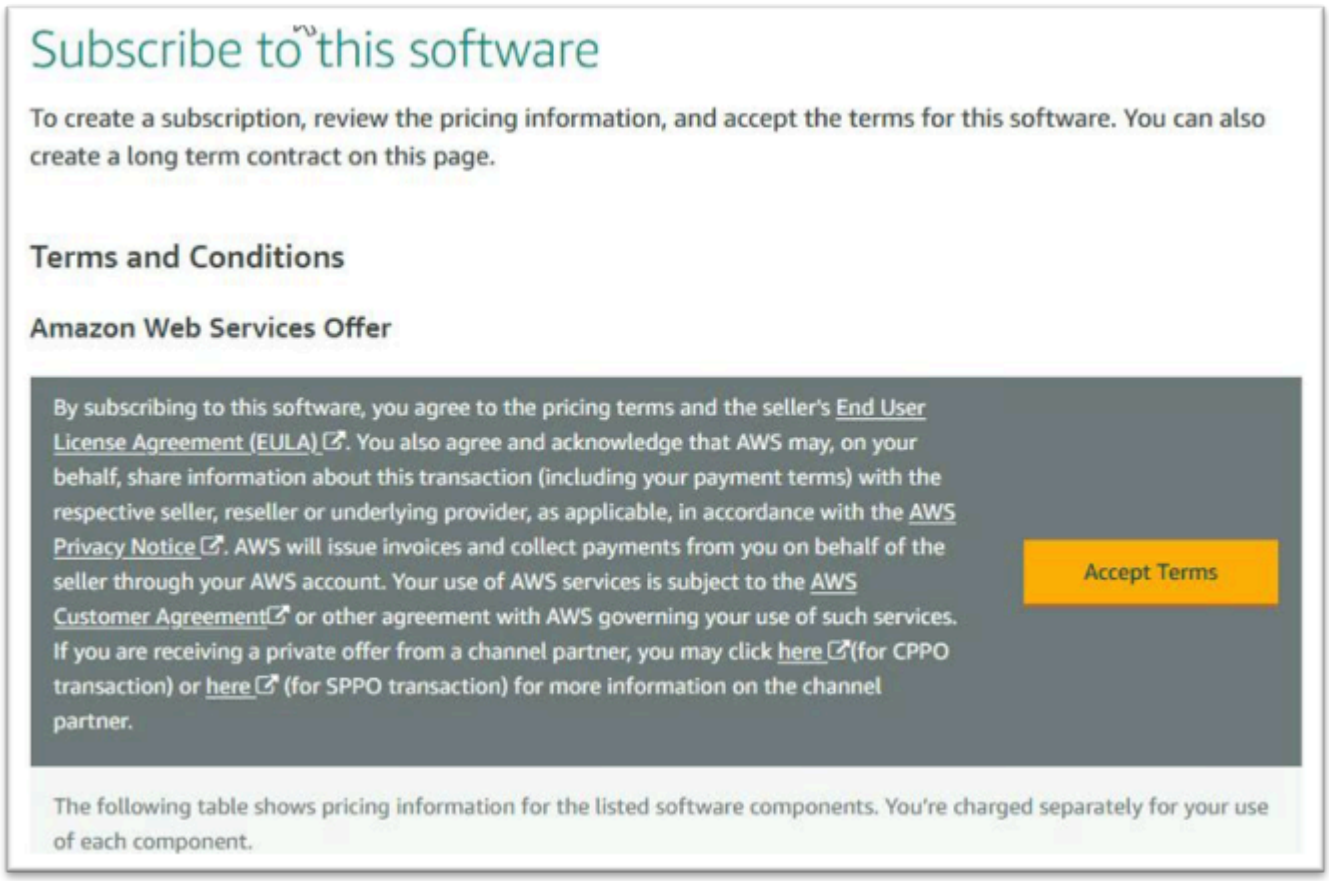
> **Note**
>
> 1. Choisissez un lien uniquement pour l'un des produits que vous êtes autorisé à utiliser.
> 2. Assurez-vous que votre compte est autorisé en suivant la Demander la mise à jour de la liste d'autorisation pour le compte page pour utiliser ces liens.

- Serveur d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-g5emev63l7blc
- Serveur d'entreprise pour Windows : https://aws.amazon.com/marketplace/pp/prodview-lwybsiyikbhc2
- Développeur d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-77qmpr42yzxwk
- Développeur d'entreprise avec Visual Studio 2022 : https://aws.amazon.com/marketplace/pp/prodview-m4l3lqiszo6cm
- Analyseur d'entreprise : https://aws.amazon.com/marketplace/ pp/prodview-tttheylcmcihm
- Outils de création d'entreprise pour Windows : https://aws.amazon.com/marketplace/pp/prodview-2rw35bbt6uozi
- Procédures stockées d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-zoeyqnsdsj6ha

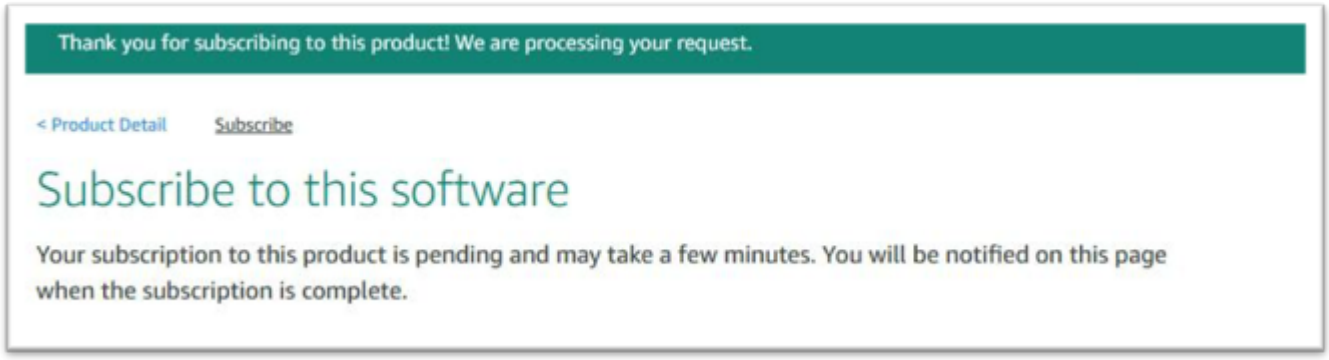
- Procédures stockées d'entreprise avec SQL Server 2019 : https://aws.amazon.com/marketplace/pp/prodview-ynfklquwubnz4

4. Choisissez Continue to Subscribe (Continuer pour s'abonner).

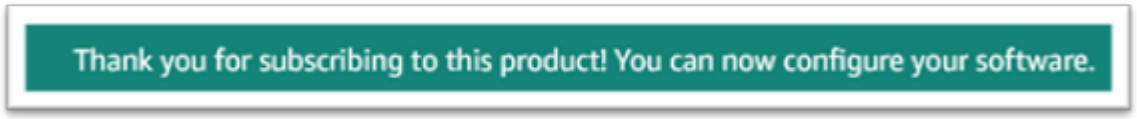
5. Si les conditions générales sont acceptables, choisissez Accepter les conditions.



6. Le traitement de l'abonnement peut prendre quelques minutes.

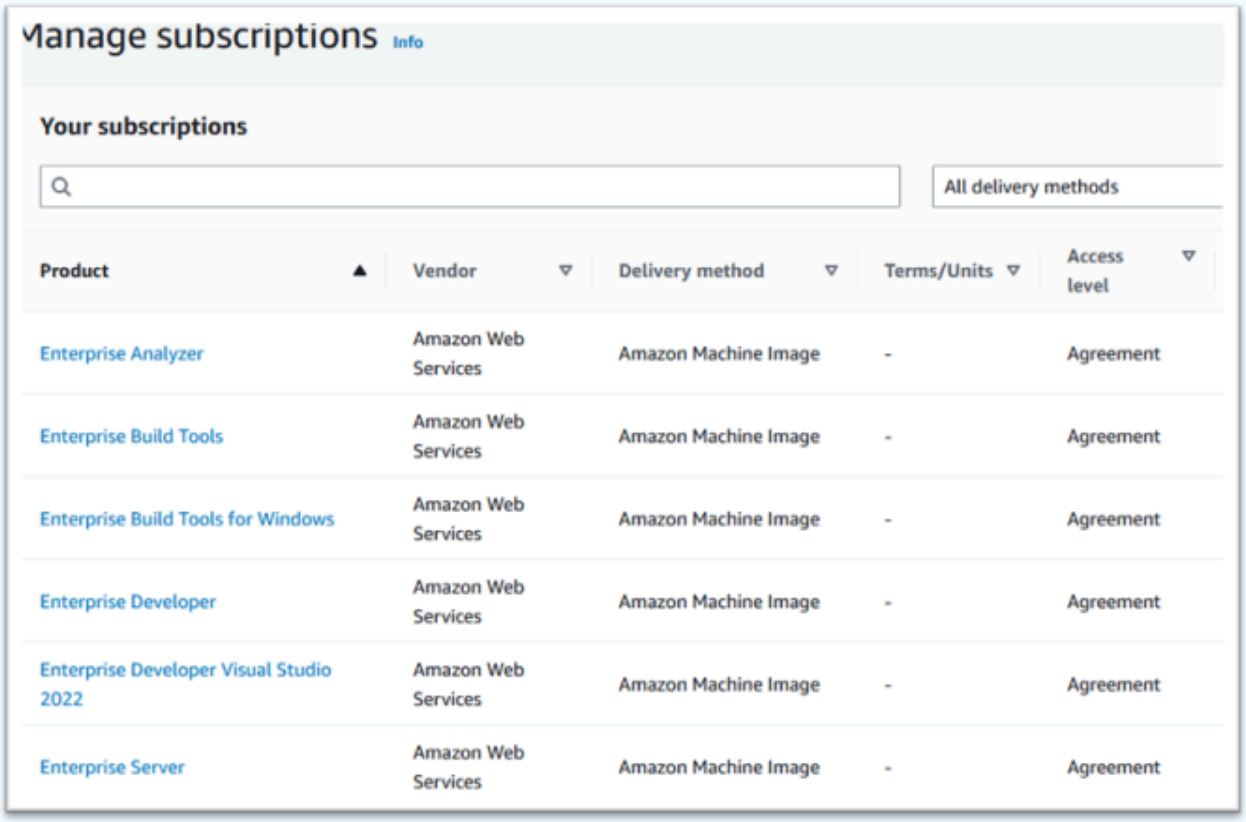


7. Une fois le message de remerciement affiché, copiez et collez le lien suivant de l'étape 3 pour continuer à ajouter des abonnements.



8. Arrêtez lorsque la section Gérer les abonnements affiche tous vos abonnés AMIs.

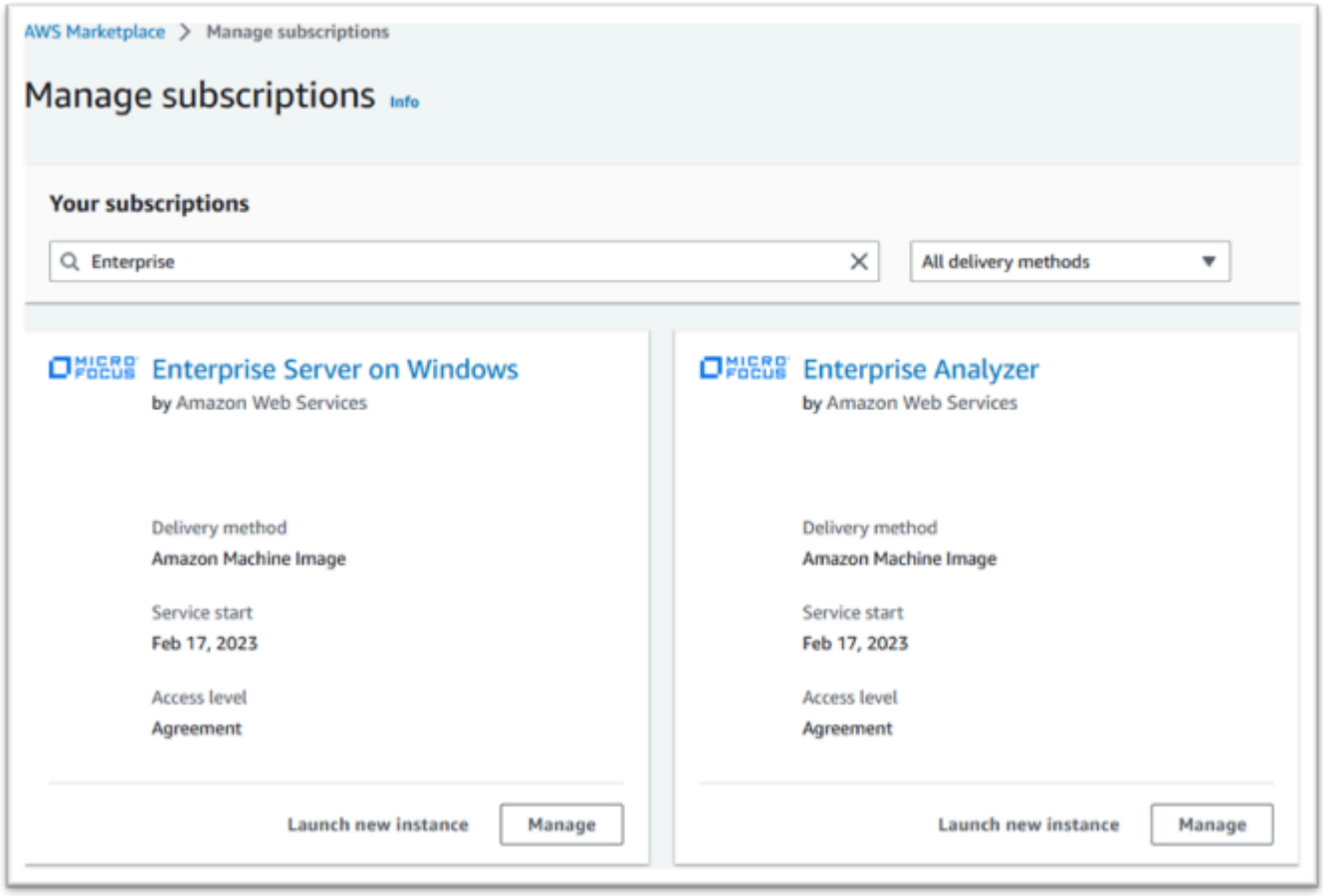
> **Note**
>
> Les préférences du panneau (icône en forme de roue dentée) sont définies pour afficher la vue sous forme de tableau.



Lancer une instance de AWS Mainframe Modernization Rocket Software (anciennement Micro Focus)

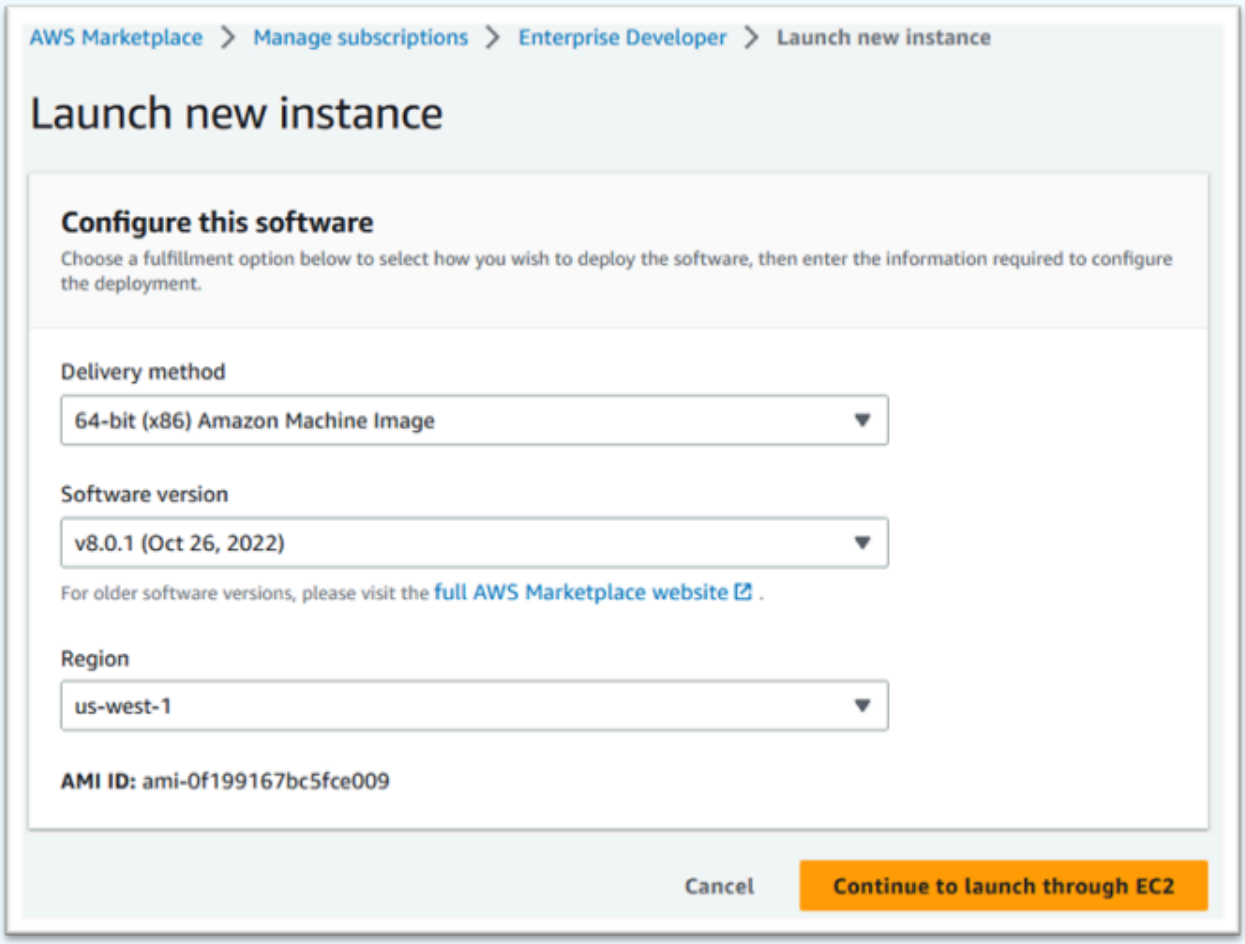
Après avoir créé des points de terminaison, une politique IAM, un rôle IAM et un abonnement à AMIs, vous êtes prêt à lancer une instance AWS Mainframe Modernization Rocket Software (Micro Focus) dans le. AWS Management Console

1. Accédez à AWS Marketplace Abonnements dans le AWS Management Console.
2. Localisez l'AMI à lancer et choisissez Launch New Instance.



3. Dans la boîte de dialogue de lancement d'une nouvelle instance, assurez-vous que la région autorisée est sélectionnée.
4. Appuyez sur Continuer pour lancer EC2.

> **Note**
>
> L'exemple suivant montre le lancement d'une AMI Enterprise Developer, mais le processus est le même pour toutes AWS Mainframe Modernization AMIs.

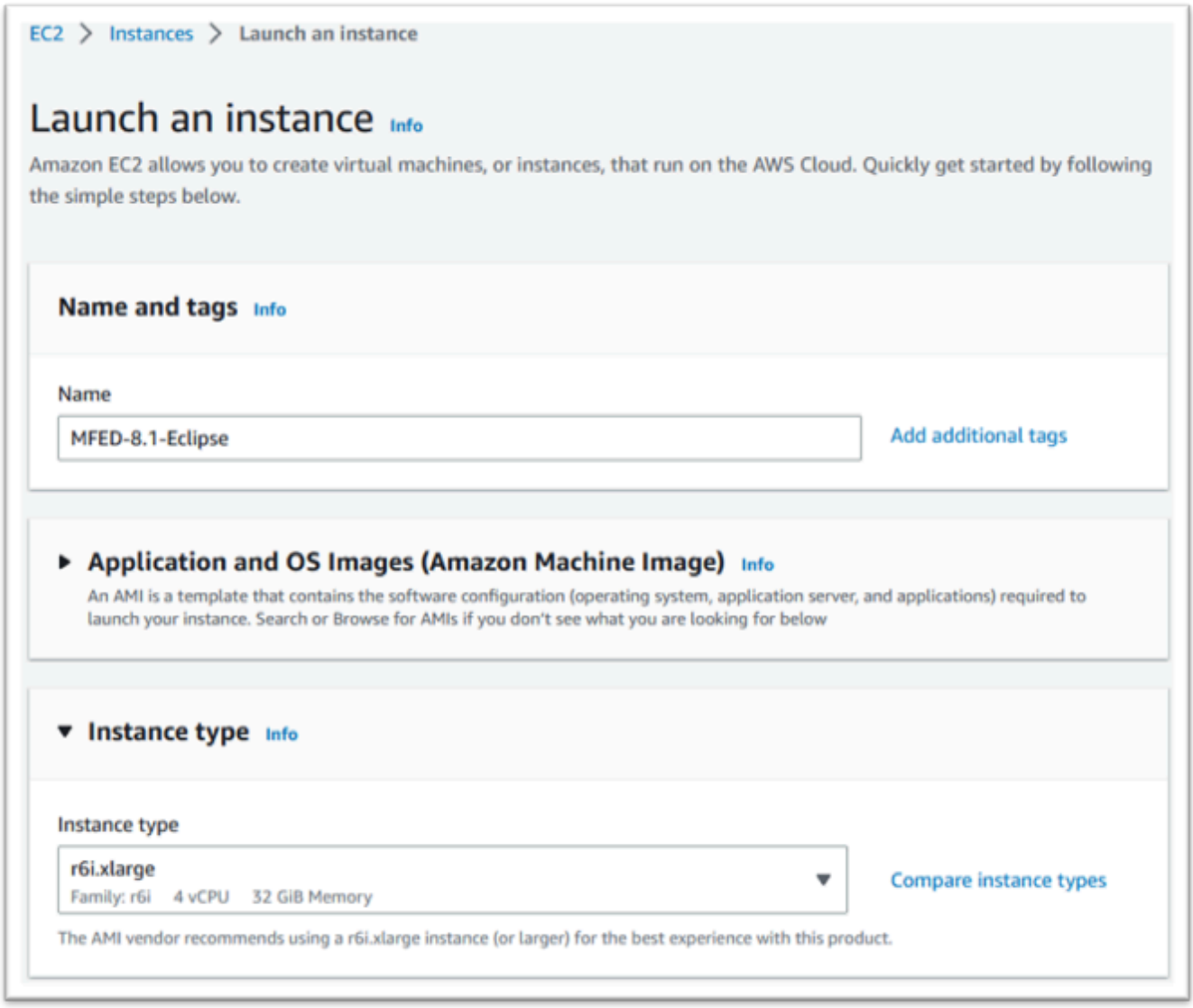


5. Entrez un nom pour le serveur.
6. Choisissez un type d'instance.

   Le type d'instance sélectionné doit être déterminé par les exigences de performance et de coût du projet. Les points de départ suggérés sont les suivants :

   - Pour Enterprise Analyzer, un r6i.xlarge
   - Pour les développeurs d'entreprise, un r6i.large
   - Pour une instance autonome d'Enterprise Server, un r6i.xlarge
   - Pour le cluster de disponibilité des performances logicielles (PAC) Rocket avec scale-out, un r6i.large

> **Note**
>
> La section Images de l'application et du système d'exploitation a été réduite pour la capture d'écran.



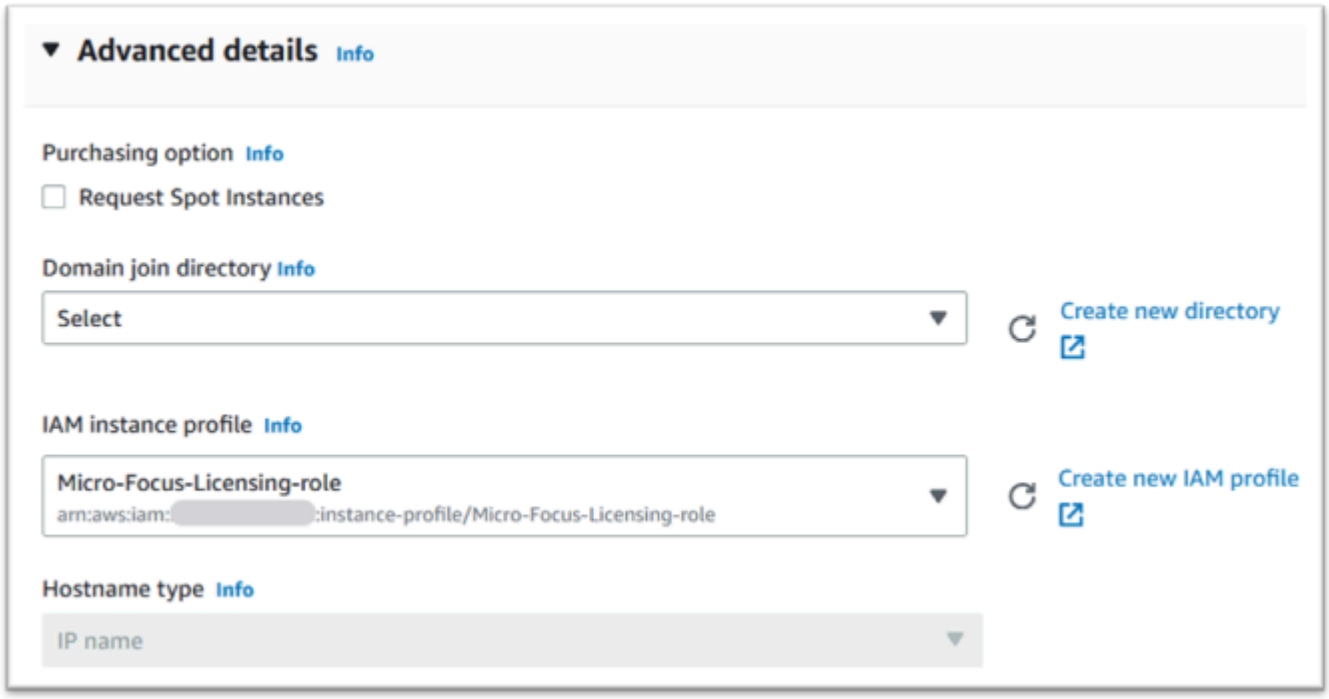
7. Choisissez ou créez (et enregistrez) une paire de clés (non illustrée).

   Pour plus d'informations sur les paires de clés pour les instances Linux, consultez les sections paires de EC2 clés Amazon et instances Linux.

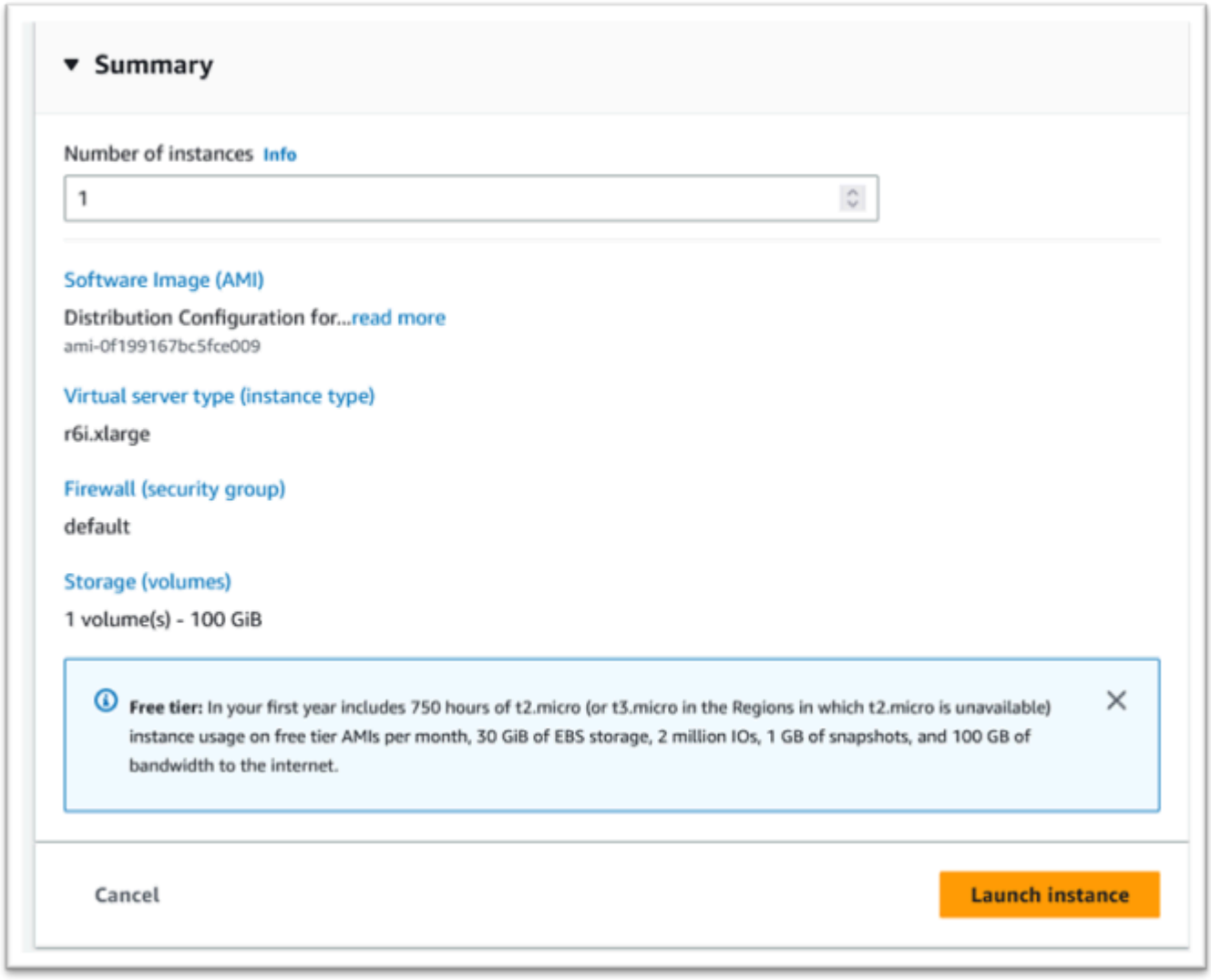
Pour plus d'informations sur les paires de clés pour les instances Windows, consultez les sections paires de EC2 clés Amazon et instances Windows.

8. Modifiez les paramètres réseau et choisissez le VPC autorisé et le sous-réseau approprié.
9. Choisissez ou créez un groupe de sécurité. S'il s'agit d'une EC2 instance de serveur d'entreprise, il est courant d'autoriser le trafic TCP vers les ports 86 et 10086 pour administrer la configuration du logiciel Rocket.
10. Configurez éventuellement le stockage pour l' EC2 instance Amazon.
11. Important - Développez les détails avancés et sous Profil d'instance IAM, choisissez le rôle de licence créé précédemment, par exemple « Micro-Focus-Licensing-Role ».

> **Note**
>
> Si cette étape est manquée, une fois l'instance créée, vous pouvez modifier le rôle IAM à partir de l'option Sécurité du menu Action de l' EC2 instance.

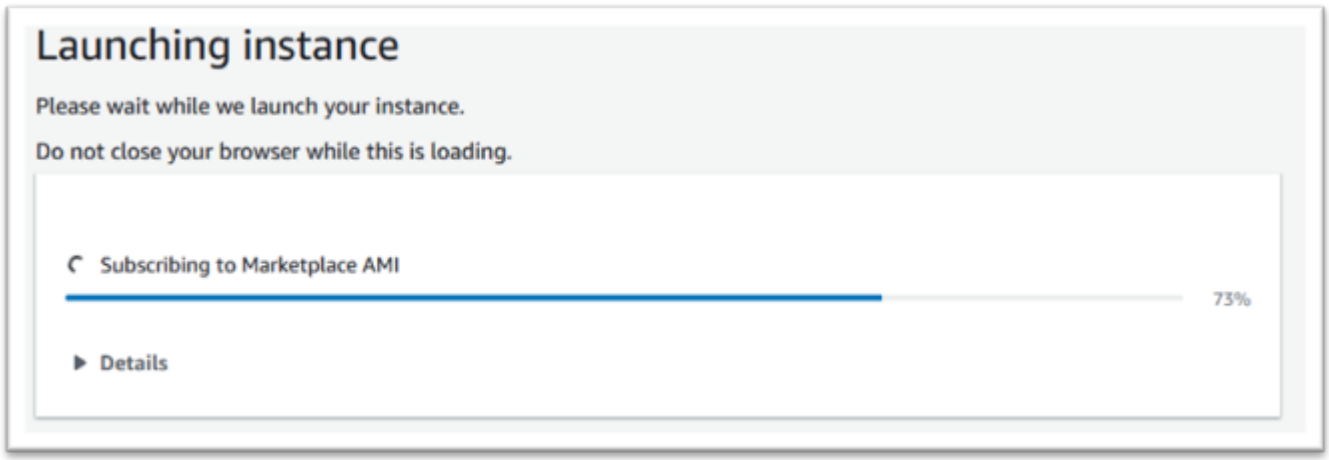


12. Consultez le résumé et appuyez sur Launch Instance.

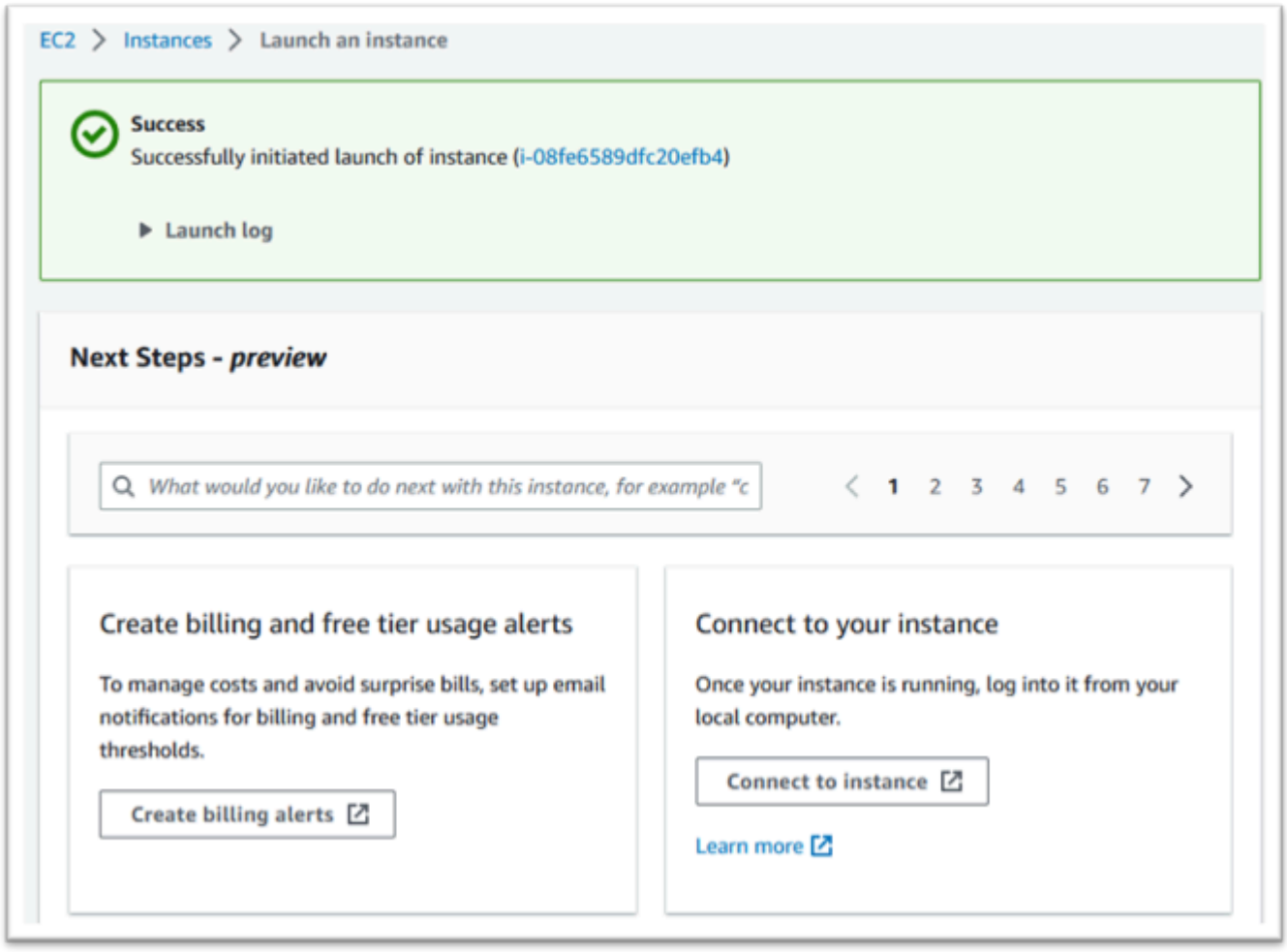


13. Le lancement de l'instance échouera si un type de serveur virtuel non valide est sélectionné.

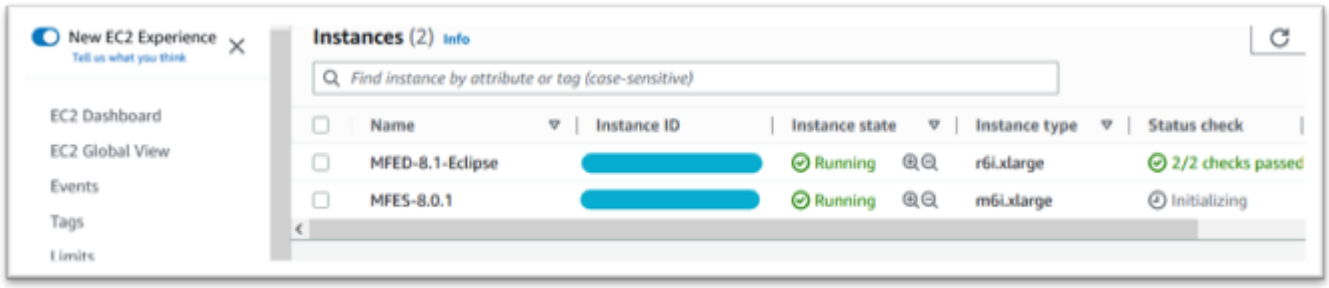
    Dans ce cas, choisissez Modifier la configuration de l'instance et modifiez le type d'instance.



14. Une fois le message « Success » affiché, choisissez Connect to instance pour obtenir les détails de connexion.



15. Vous pouvez également accéder EC2au AWS Management Console.
16. Choisissez Instances pour voir le statut de la nouvelle instance.



Sous-réseau ou VPC sans accès à Internet

Apportez ces modifications supplémentaires si le sous-réseau ou le VPC ne dispose pas d'un accès Internet sortant.

Le gestionnaire de licence doit accéder aux services AWS suivants :

- com.amazonaws. *region*.s3
- com.amazonaws. *region*.ec2
- com.amazonaws. *region*.gestionnaire de licences
- com.amazonaws. *region*.sts

Les étapes précédentes ont défini le com.amazonaws. *region*service .s3 en tant que point de terminaison de passerelle. Ce point de terminaison a besoin d'une entrée de table de routage pour tous les sous-réseaux sans accès à Internet.

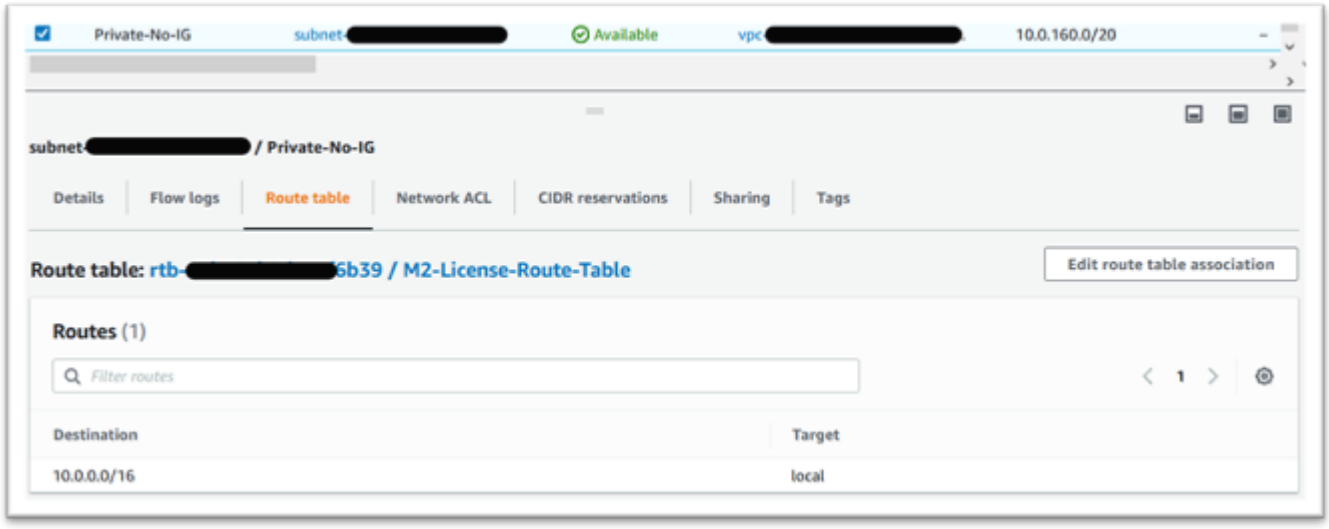
Les trois services supplémentaires seront définis comme des points de terminaison d'interface.

Rubriques

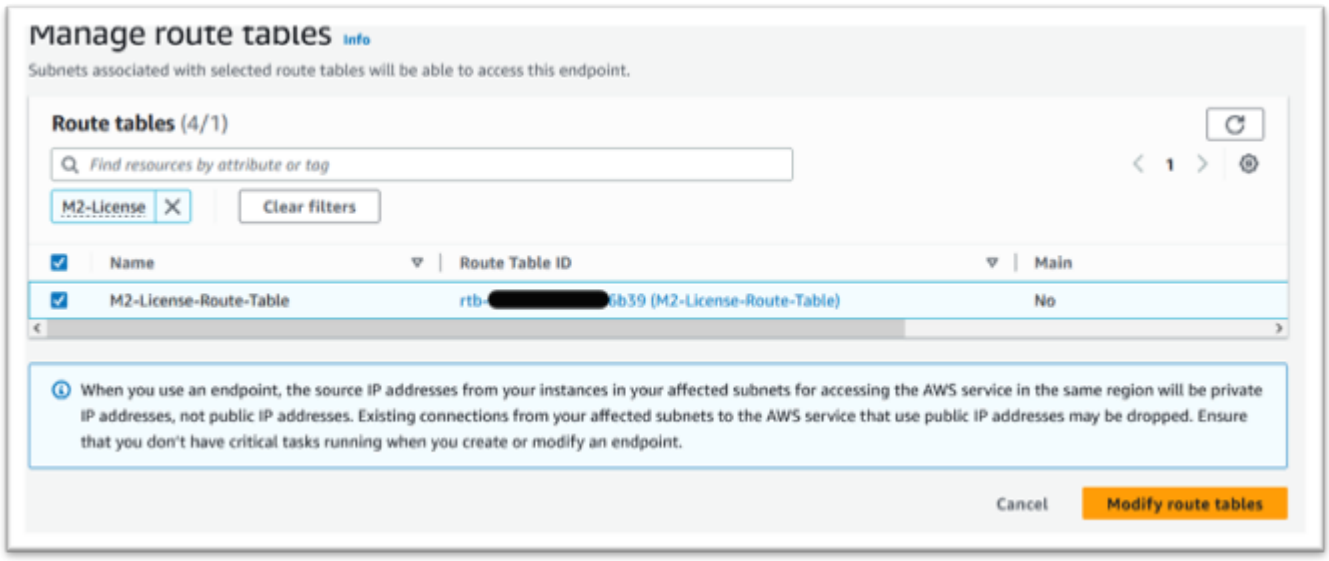
- Ajoutez l'entrée de table de routage pour le point de terminaison Amazon S3
- Définissez le groupe de sécurité requis
- Création des points de terminaison du service

Ajoutez l'entrée de table de routage pour le point de terminaison Amazon S3

1. Accédez à VPC dans le AWS Management Console et choisissez Subnets.
2. Choisissez le sous-réseau dans lequel les EC2 instances Amazon seront créées et cliquez sur l'onglet Route Table.
3. Notez les quelques derniers chiffres de l'identifiant de la table de routage. Par exemple, le 6b39 dans l'image ci-dessous.



4. Choisissez Endpoints dans le volet de navigation.
5. Choisissez le point de terminaison créé précédemment, puis gérez les tables de routage, soit dans l'onglet Tables de routage du point de terminaison, soit dans le menu déroulant Actions.
6. Choisissez la table de routage à l'aide des chiffres identifiés précédemment et appuyez sur Modifier les tables de routage.

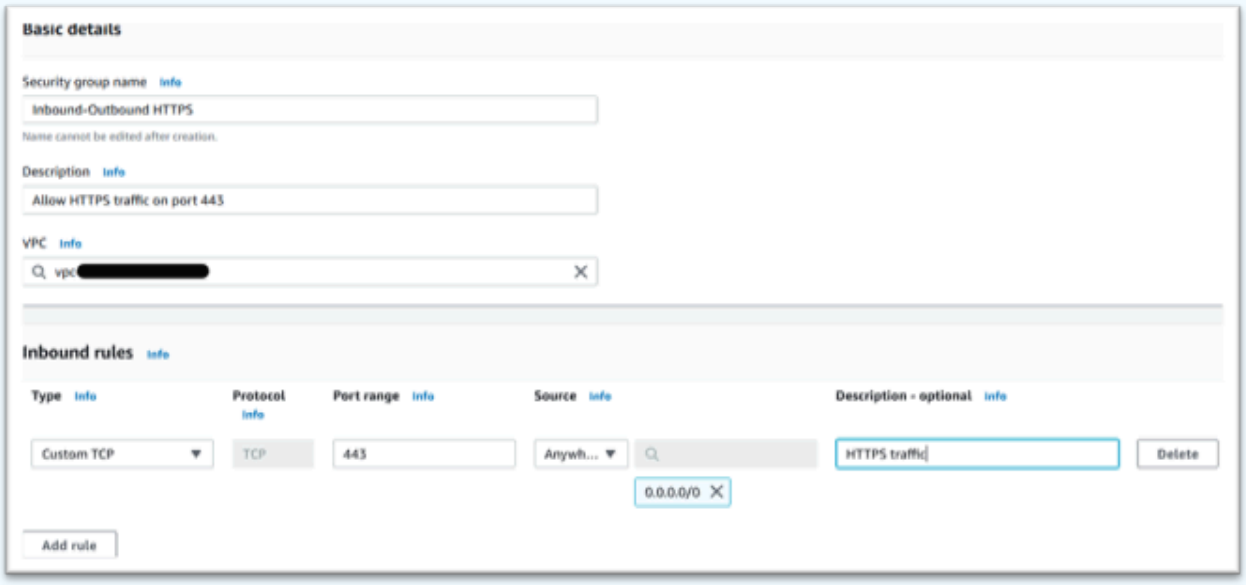


Définissez le groupe de sécurité requis

Les services Amazon EC2 et License Manager communiquent via HTTPS via le port 443. AWS STS Cette communication est bidirectionnelle et nécessite des règles entrantes et sortantes pour permettre à l'instance de communiquer avec les services.

1. Accédez à Amazon VPC dans le. AWS Management Console
2. Localisez les groupes de sécurité dans la barre de navigation et choisissez Créer un groupe de sécurité.
3. Entrez le nom et la description du groupe de sécurité, par exemple « Inbound-Outbound HTTPS ».
4. Appuyez sur le X dans la zone de sélection du VPC pour supprimer le VPC par défaut, puis choisissez le VPC qui contient le point de terminaison S3.
5. Ajoutez une règle entrante qui autorise le trafic TCP sur le port 443 depuis n'importe où.

> **Note**
>
> Les règles entrantes (et sortantes) peuvent être restreintes davantage en limitant la source. Pour plus d'informations, consultez la section Contrôlez le trafic vers vos AWS ressources à l'aide de groupes de sécurité dans le guide de l'utilisateur Amazon VPC.



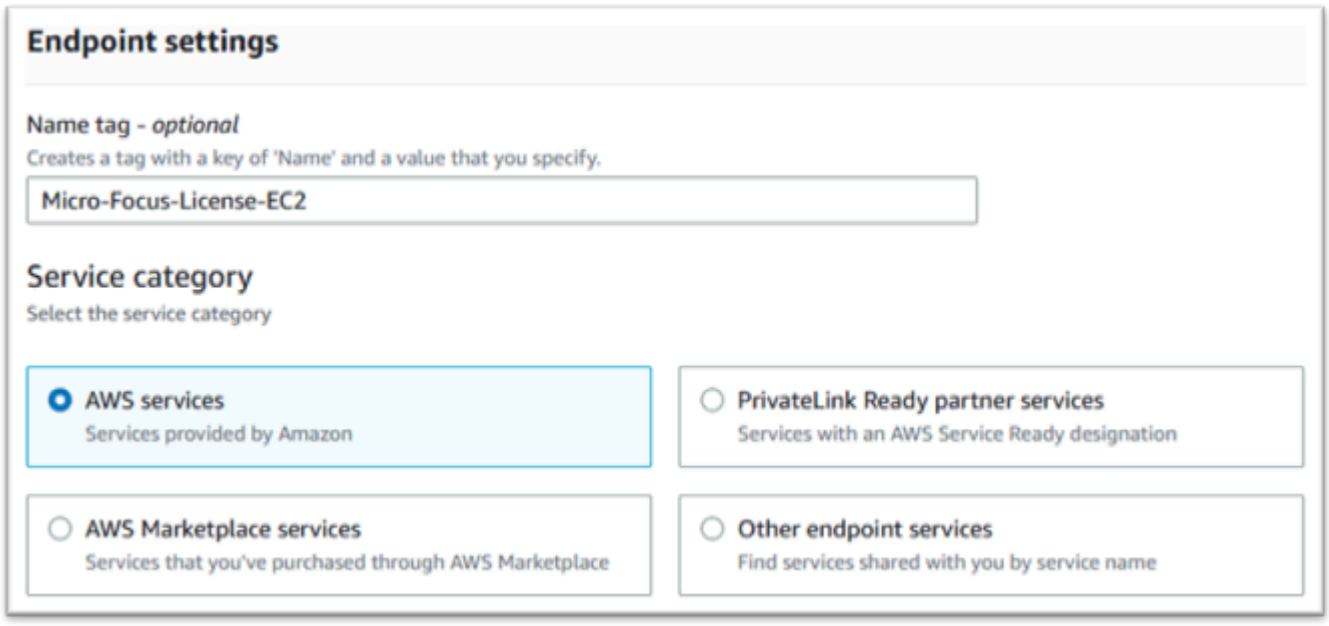
6. Appuyez sur Créer un groupe de sécurité.

Création des points de terminaison du service

Répétez ce processus trois fois, une fois pour chaque service.

1. Accédez à Amazon VPC dans le AWS Management Console et choisissez Endpoints.
2. Appuyez sur Créer un point de terminaison.

3. Entrez un nom, par exemple « Micro-Focus-License- EC2 », « Micro-Focus-License-STS » ou « Micro-Focus-License-Manager ».
4. Choisissez la catégorie de service AWS Services.



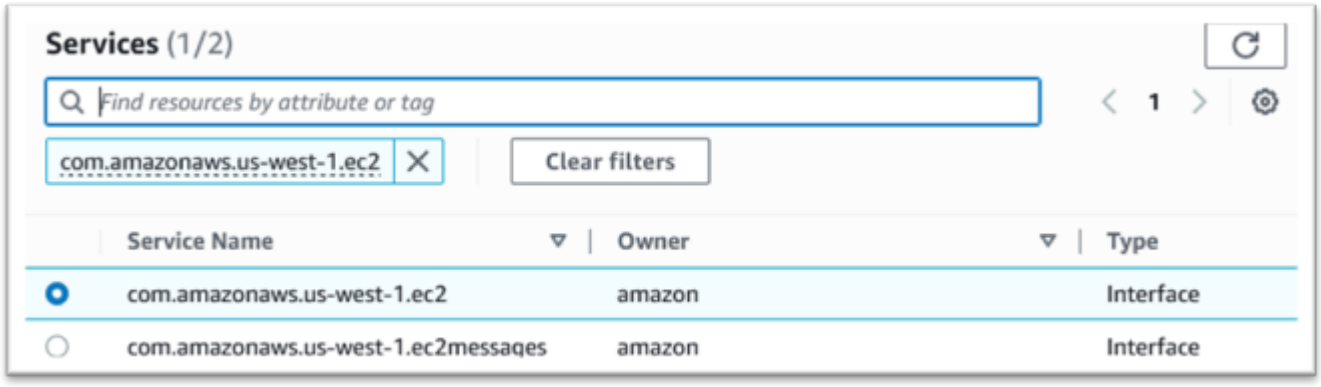
5. Sous Services, recherchez le service d'interface correspondant, qui est l'un des suivants :

   - « com.amazonaws. *region*.ec2 »
   - « com.amazonaws. *region*.sts »
   - « com.amazonaws. *region*.gestionnaire de licences »

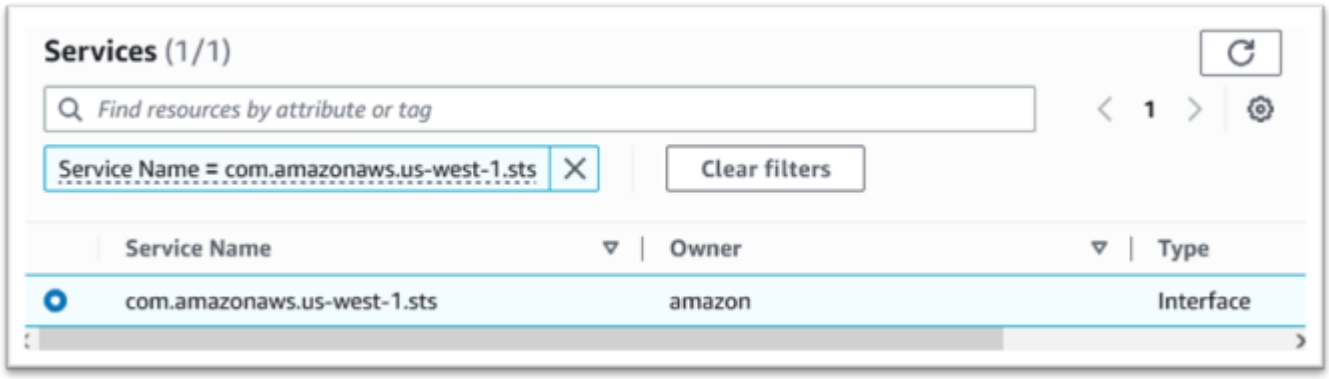
   Par exemple :

   - « com.amazonaws.us-west-1.ec2 »
   - « com.amazonaws.us-west-1.sts »
   - « com.amazonaws.us-west-1.license-manager »
6. Choisissez le service d'interface correspondant.

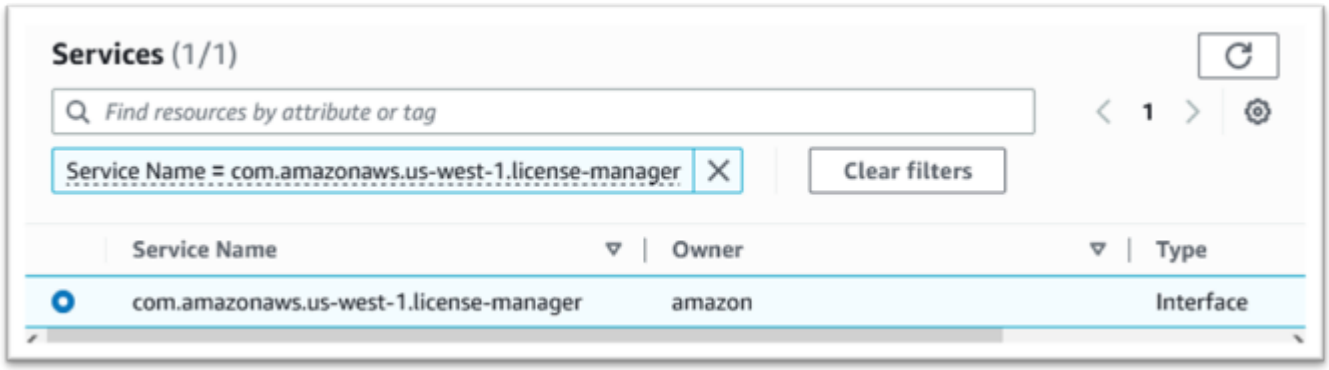
   com.amazonaws. ***region***.ec2 :



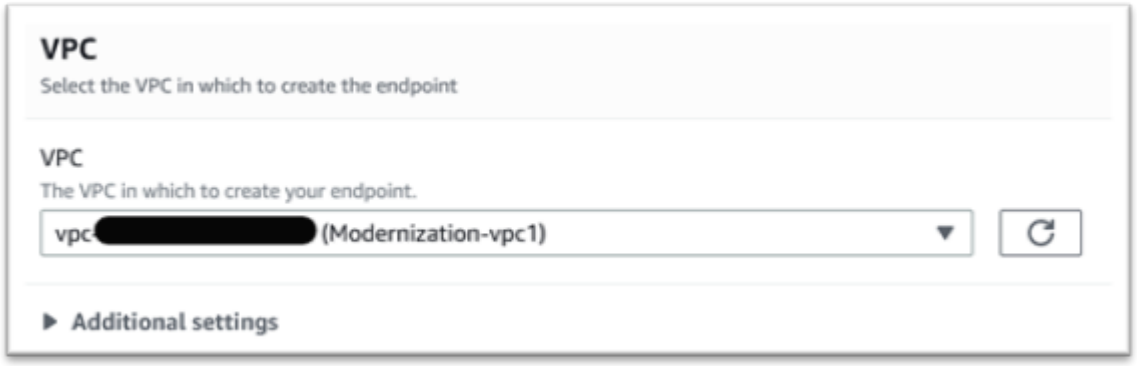
com.amazonaws. ***region***.sts :



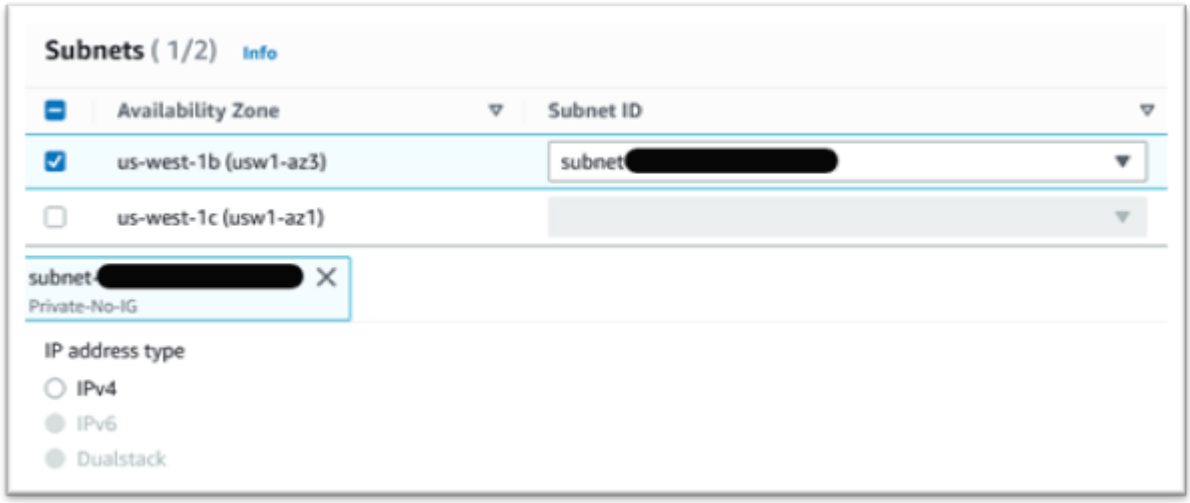
com.amazonaws. ***region***.gestionnaire de licences :



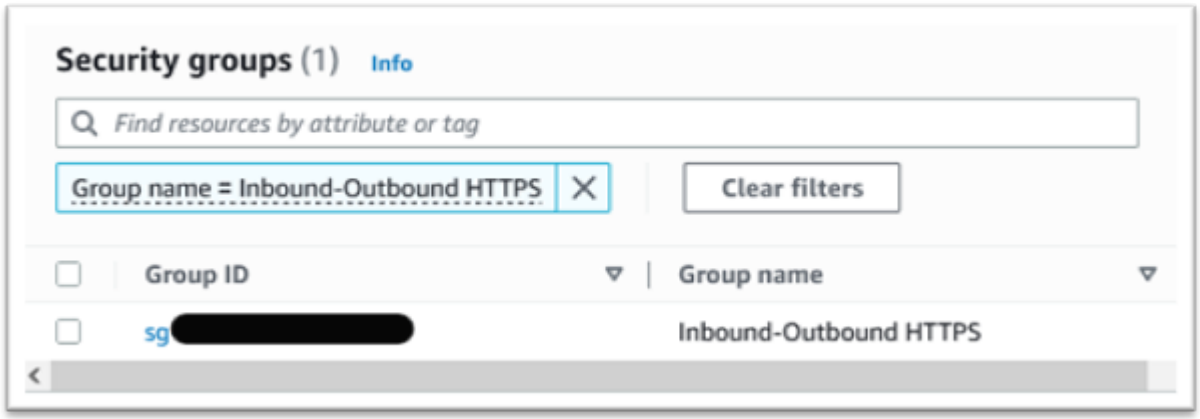
7. Pour VPC, choisissez le VPC pour l'instance.



8. Choisissez la zone de disponibilité et les sous-réseaux pour le VPC.



9. Choisissez le groupe de sécurité créé précédemment.



10. Sous Politique, sélectionnez Accès complet.

11. Choisissez Créer un point de terminaison.
12. Répétez cette procédure pour les autres interfaces.

Configurer l'automatisation des sessions de streaming pour Rocket Enterprise Analyzer (anciennement Micro Focus) et Rocket Enterprise Developer

Vous pouvez exécuter automatiquement un script au début et à la fin de la session pour permettre une automatisation spécifique au contexte de votre client. Pour plus d'informations sur cette fonctionnalité AppStream 2.0, consultez la section Utiliser des scripts de session pour gérer l'expérience de streaming de vos utilisateurs AppStream 2.0 dans le guide d'administration Amazon AppStream 2.0.

Cette fonctionnalité nécessite que vous disposiez au moins des versions suivantes des images Enterprise Analyzer et Enterprise Developer :

- `m2-enterprise-analyzer-v8.0.4.R1`
- `m2-enterprise-developer-v8.0.4.R1`

Rubriques

- Configurer l'automatisation au début de la session
- Configurer l'automatisation à la fin de la session

Configurer l'automatisation au début de la session

Si vous souhaitez exécuter un script d'automatisation lorsque les utilisateurs se connectent à la AppStream version 2.0, créez votre script et nommez-lem2-user-setup.cmd. Stockez le script dans le dossier d'accueil AppStream 2.0 de l'utilisateur. Les images AppStream 2.0 fournies par AWS Mainframe Modernization recherchent un script portant ce nom à cet emplacement et l'exécutent s'il existe.

> **Note**
>
> La durée du script ne peut pas dépasser la limite fixée par AppStream 2.0, qui est actuellement de 60 secondes. Pour plus d'informations, consultez la section Exécuter des scripts avant le début des sessions de streaming dans le guide d'administration Amazon AppStream 2.0.

Configurer l'automatisation à la fin de la session

Si vous souhaitez exécuter un script d'automatisation lorsque les utilisateurs se déconnectent de la AppStream version 2.0, créez votre script et nommez-lem2-user-teardown.cmd. Stockez le script dans le dossier d'accueil AppStream 2.0 de l'utilisateur. Les images AppStream 2.0 fournies par AWS Mainframe Modernization recherchent un script portant ce nom à cet emplacement et l'exécutent s'il existe.

> **Note**
>
> La durée du script ne peut pas dépasser la limite fixée par AppStream 2.0, qui est actuellement de 60 secondes. Pour plus d'informations, consultez la section Exécuter des scripts après la fin des sessions de streaming dans le guide d'administration Amazon AppStream 2.0.

Afficher les ensembles de données sous forme de tables et de colonnes dans Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)

Vous pouvez accéder aux ensembles de données du mainframe déployés dans AWS Mainframe Modernization à l'aide du runtime Rocket Software (anciennement Micro Focus). Vous pouvez afficher les ensembles de données migrés sous forme de tables et de colonnes à partir d'une instance de Rocket Enterprise Developer. L'affichage des ensembles de données de cette façon vous permet de :

- Effectuez `SQL SELECT` des opérations sur les fichiers de données migrés.
- Exposez les données en dehors de l'application mainframe migrée sans modifier l'application.
- Filtrez facilement les données et enregistrez-les au format CSV ou dans un autre format de fichier.

> **Note**
>
> Les étapes 1 et 2 sont des activités ponctuelles. Répétez les étapes 3 et 4 pour chaque ensemble de données afin de créer les vues de base de données.

Rubriques

- Prérequis
- Étape 1 : configurer la connexion ODBC à la banque de données Rocket Software (base de données Amazon RDS)
- Étape 2 : Création du fichier MFDBFH.cfg
- Étape 3 : Création d'un fichier de structure (STR) pour la mise en page de votre cahier
- Étape 4 : Création d'une vue de base de données à l'aide du fichier de structure (STR)
- Étape 5 : Afficher les ensembles de données de Rocket Software (anciennement Micro Focus) sous forme de tableaux et de colonnes

Prérequis

- Vous devez avoir accès à Rocket Enterprise Developer Desktop via la AppStream version 2.0.

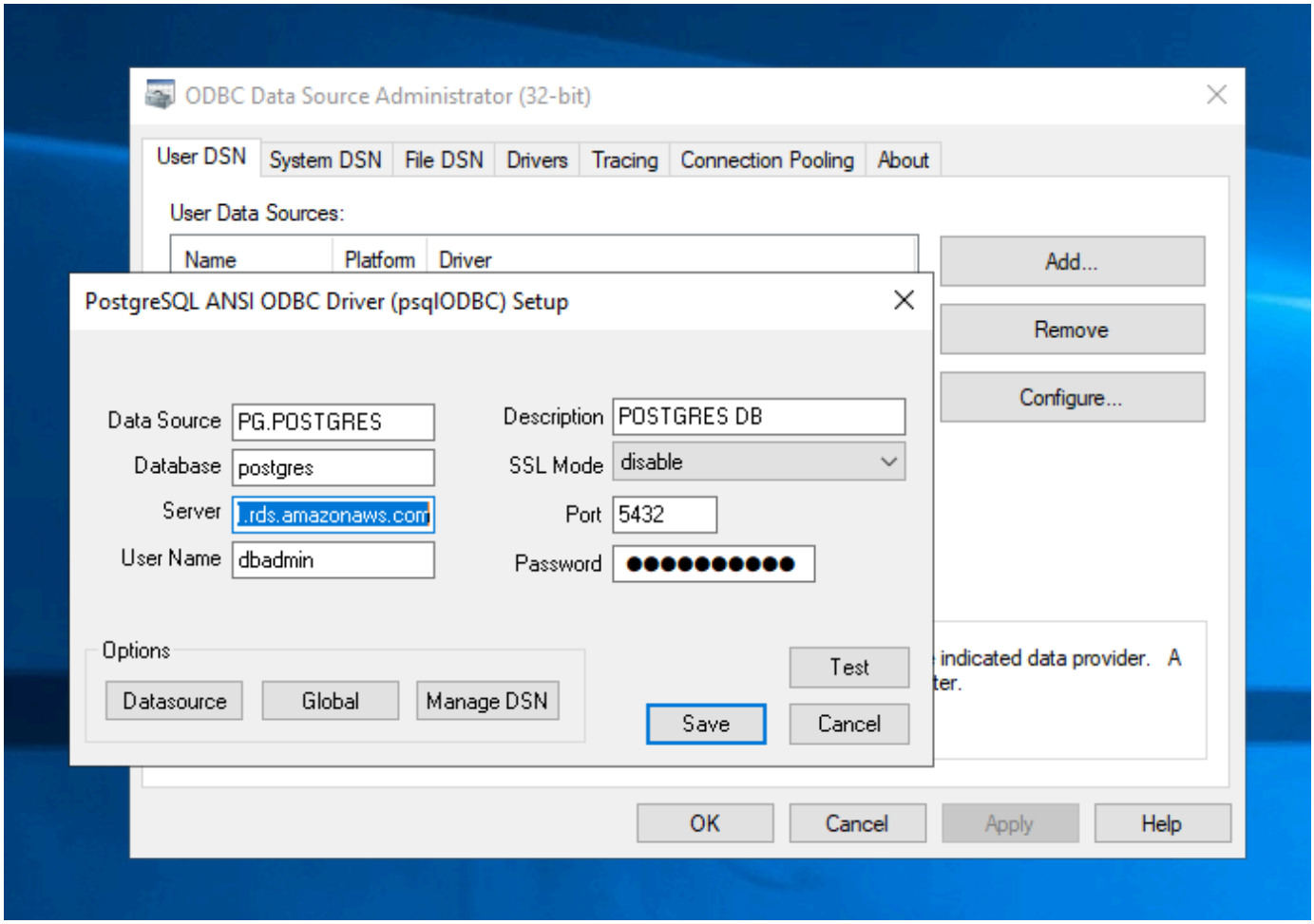
- Une application doit être déployée et exécutée dans le cadre de la modernisation du AWS mainframe à l'aide du moteur d'exécution Rocket Software.
- Vous stockez les données de votre application dans Aurora PostgreSQL Compatible Edition.

Étape 1 : configurer la connexion ODBC à la banque de données Rocket Software (base de données Amazon RDS)

Au cours de cette étape, vous configurez une connexion ODBC à la base de données qui contient les données que vous souhaitez afficher sous forme de tables et de colonnes. Il s'agit d'une étape unique.

1. Connectez-vous à Rocket Enterprise Developer Desktop à l'aide de l'URL de streaming AppStream 2.0.
2. Ouvrez l'administrateur de sources de données ODBC, choisissez User DSN, puis choisissez Ajouter.
3. Dans Create New Data Source, sélectionnez PostgreSQL ANSI, puis Finish.
4. Créez une source de données pour `PG.POSTGRES` en fournissant les informations de base de données nécessaires, comme suit :

```
Data Source : PG.POSTGRES
Database    : postgres
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
User Name   : user_name
Password    : user_password
```



5. Choisissez Test pour vous assurer que la connexion fonctionne. Le message devrait s'afficher en `Connection successful` cas de réussite du test.

   Si le test échoue, consultez les informations suivantes.

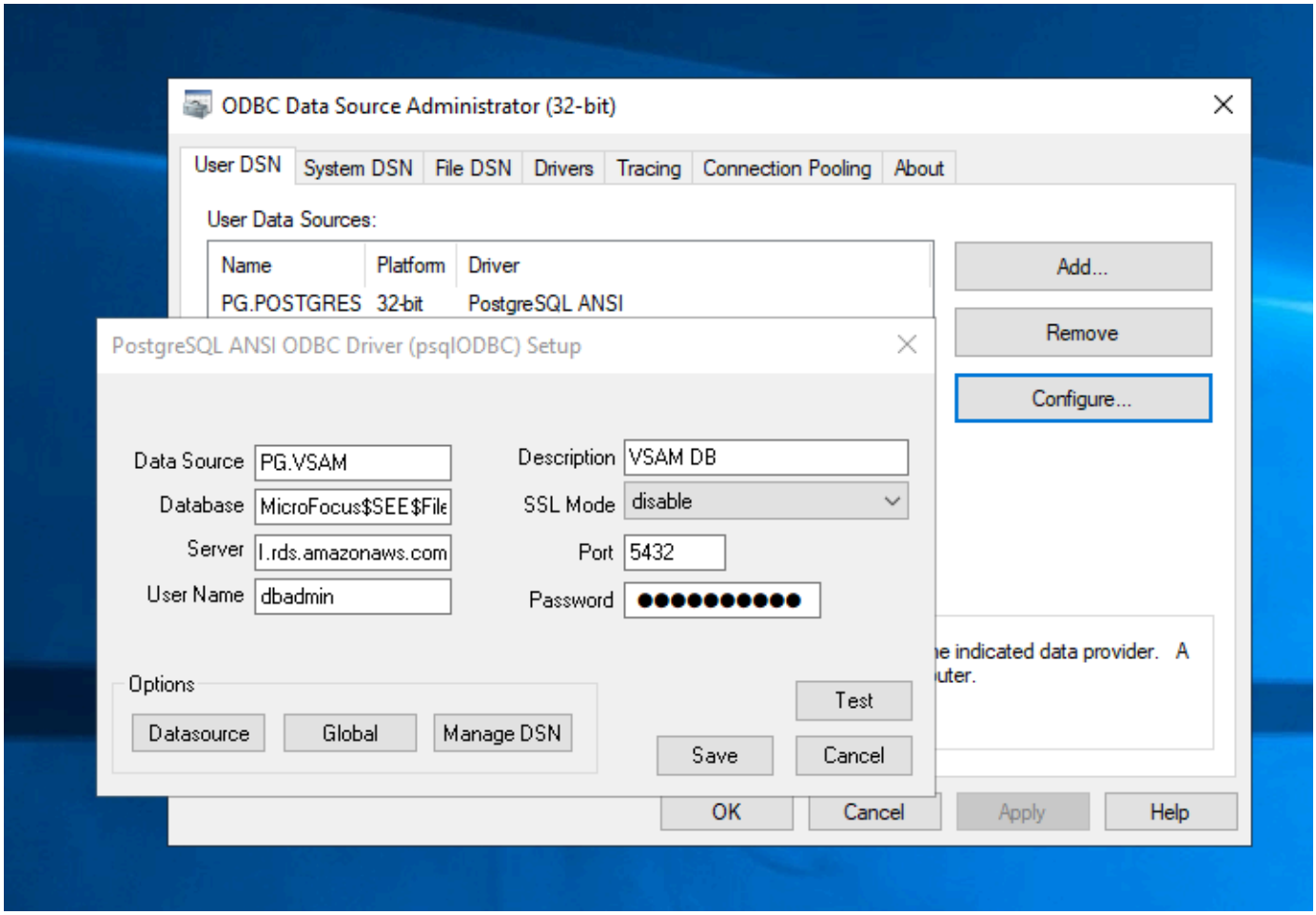
   - Résolution des problèmes pour Amazon RDS
   - Comment résoudre les problèmes lors de la connexion à mon instance de base de données Amazon RDS ?

6. Enregistrez la source de données.

7. Créez une source de données pour`PG.VSAM`, testez la connexion et enregistrez-la. Fournissez les informations de base de données suivantes :

```
Data Source : PG.VSAM
Database    : MicroFocus$SEE$Files$VSAM
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
```

```
User Name    : user_name
Password     : user_password
```



Étape 2 : Création du fichier MFDBFH.cfg

Au cours de cette étape, vous allez créer un fichier de configuration qui décrit le magasin de données Micro Focus. Il s'agit d'une étape de configuration unique.

1. Dans votre dossier principal, par exemple, dans`D:\PhotonUser\My Files\Home Folder \MFED\cfg\MFDBFH.cfg`, créez le fichier MFDBFH.cfg avec le contenu suivant.

```
<datastores>
       <server name="ESPACDatabase" type="postgresql" access="odbc">
        <dsn name="PG.POSTGRES" type="database" dbname="postgres"/>
        <dsn name="PG.VSAM" type="datastore" dsname="VSAM"/>
       </server>
```

```
      </datastores>
```

2. Vérifiez la configuration du MFDBFH en exécutant les commandes suivantes pour interroger la banque de données Micro Focus :

```
*##*
*## Test the connection by running the following commands*
*##*

set MFDBFH_CONFIG="D:\PhotonUser\My Files\Home Folder\MFED\cfg\MFDBFH.cfg"

dbfhdeploy list sql://ESPACDatabase/VSAM?folder=/DATA
```

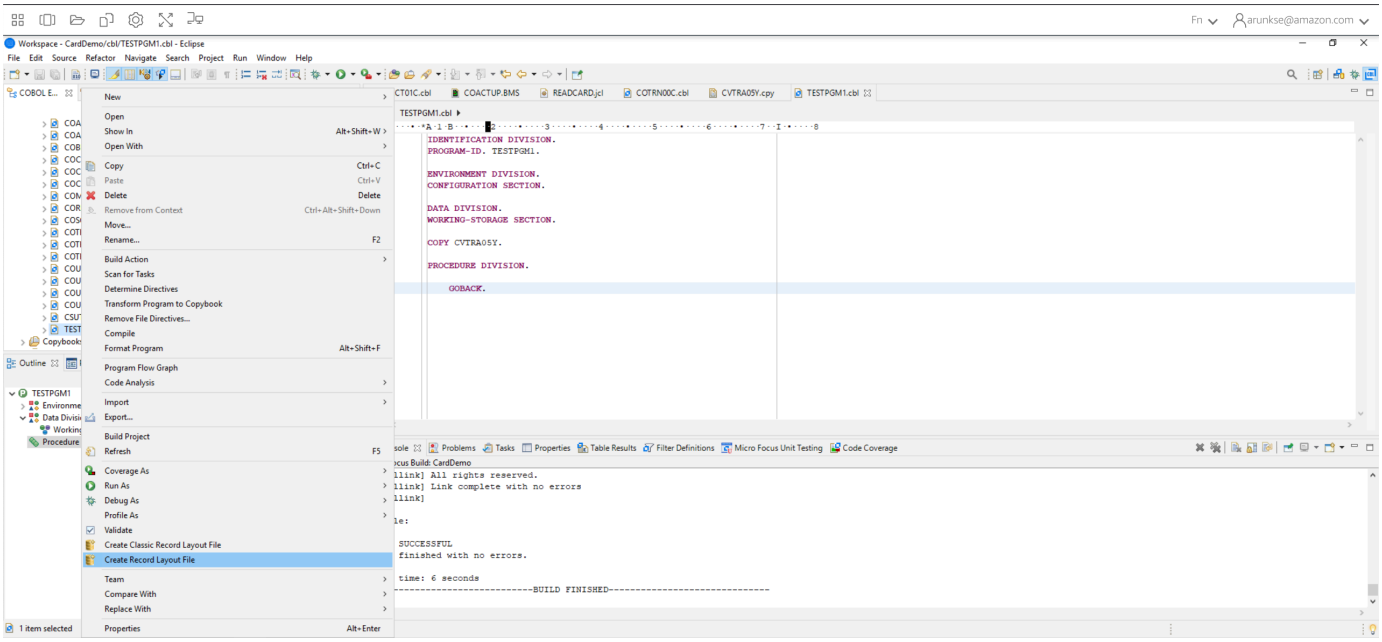
Étape 3 : Création d'un fichier de structure (STR) pour la mise en page de votre cahier

Au cours de cette étape, vous allez créer un fichier de structure pour la mise en page de votre cahier afin de pouvoir l'utiliser ultérieurement pour créer des vues de base de données à partir des ensembles de données.

1. Compilez le programme associé à votre cahier. Si aucun programme n'utilise le cahier, créez et compilez un programme simple comme celui-ci avec une instruction COPY pour votre cahier.

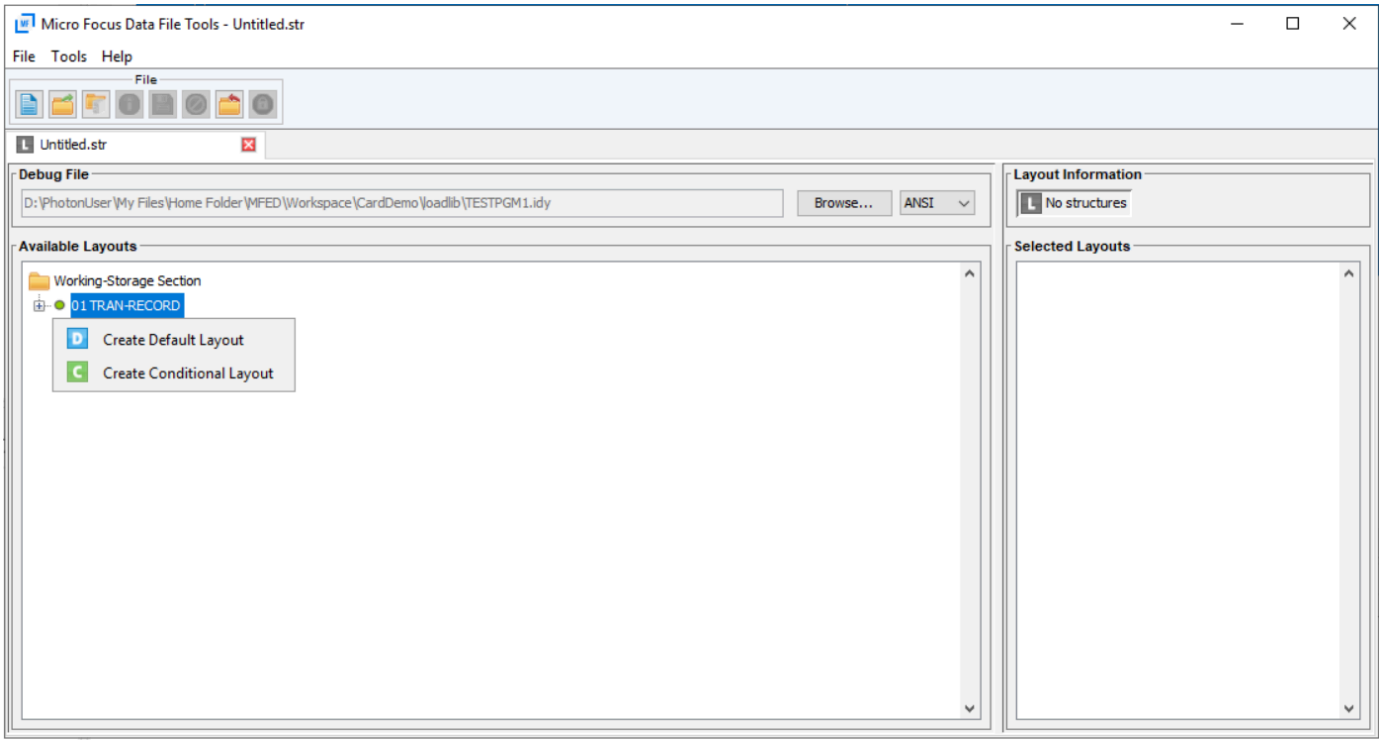
```
IDENTIFICATION DIVISION.
      PROGRAM-ID. TESTPGM1.

      ENVIRONMENT DIVISION.
      CONFIGURATION SECTION.

      DATA DIVISION.
      WORKING-STORAGE SECTION.

      COPY CVTRA05Y.

      PROCEDURE DIVISION.

      GOBACK.
```

2. Une fois la compilation réussie, cliquez avec le bouton droit sur le programme et choisissez Créer un fichier de mise en page d'enregistrement. Cela ouvrira les outils de fichiers de données Micro Focus à l'aide du fichier .idy généré lors de la compilation.

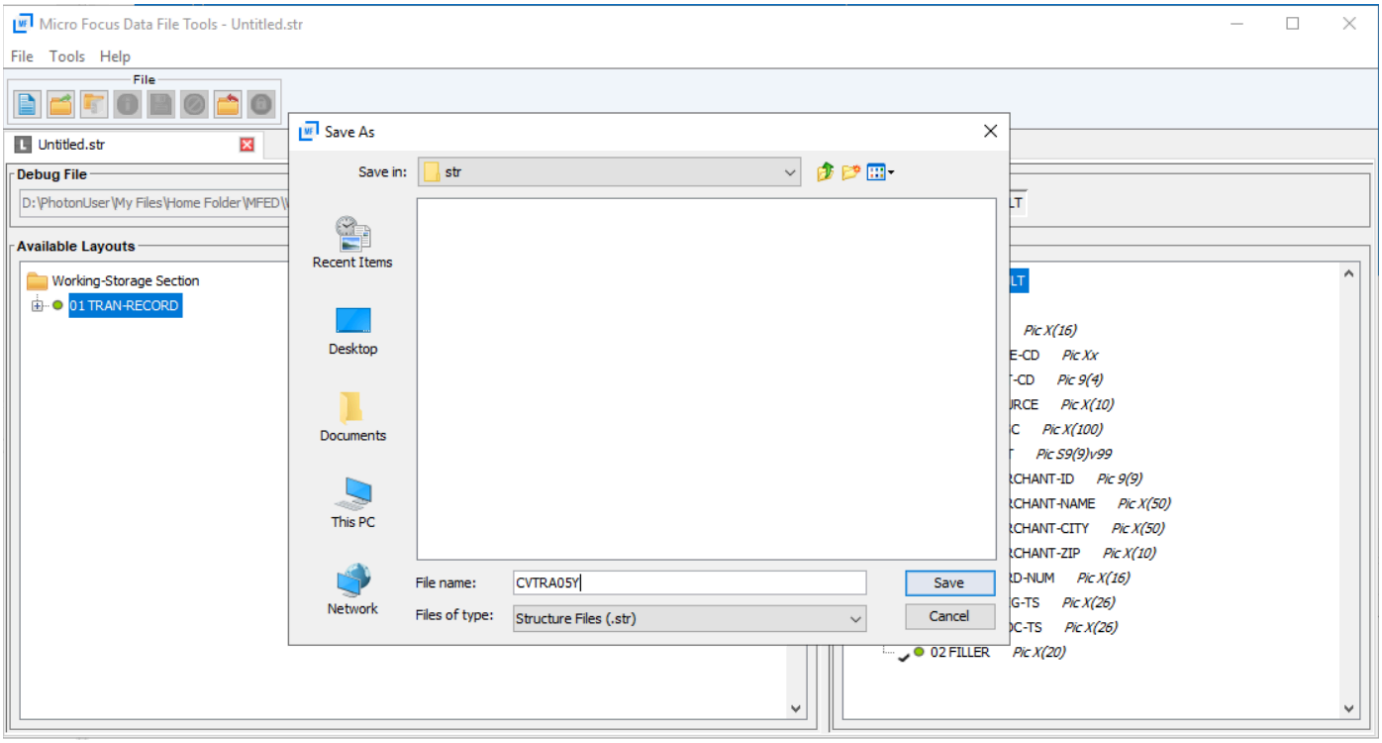


3. Cliquez avec le bouton droit sur la structure d'enregistrement et choisissez Créer une mise en page par défaut (structure unique) ou Créer une mise en page conditionnelle (structure multiple) en fonction de la mise en page.

   Pour plus d'informations, consultez la section Création de fichiers de structure et de mises en page dans la documentation Micro Focus.



4. Après avoir créé la mise en page, choisissez Fichier dans le menu, puis cliquez sur Enregistrer sous. Parcourez et enregistrez le fichier dans votre dossier personnel avec le même nom de fichier que votre bloc-notes. Vous pouvez choisir de créer un dossier appelé `str` et d'y enregistrer tous vos fichiers de structure.



Étape 4 : Création d'une vue de base de données à l'aide du fichier de structure (STR)

Au cours de cette étape, vous utilisez le fichier de structure créé précédemment pour créer une vue de base de données pour un ensemble de données.

- Utilisez la `dbfhview` commande pour créer une vue de base de données pour un ensemble de données qui se trouve déjà dans la banque de données Micro Focus, comme illustré dans l'exemple suivant.

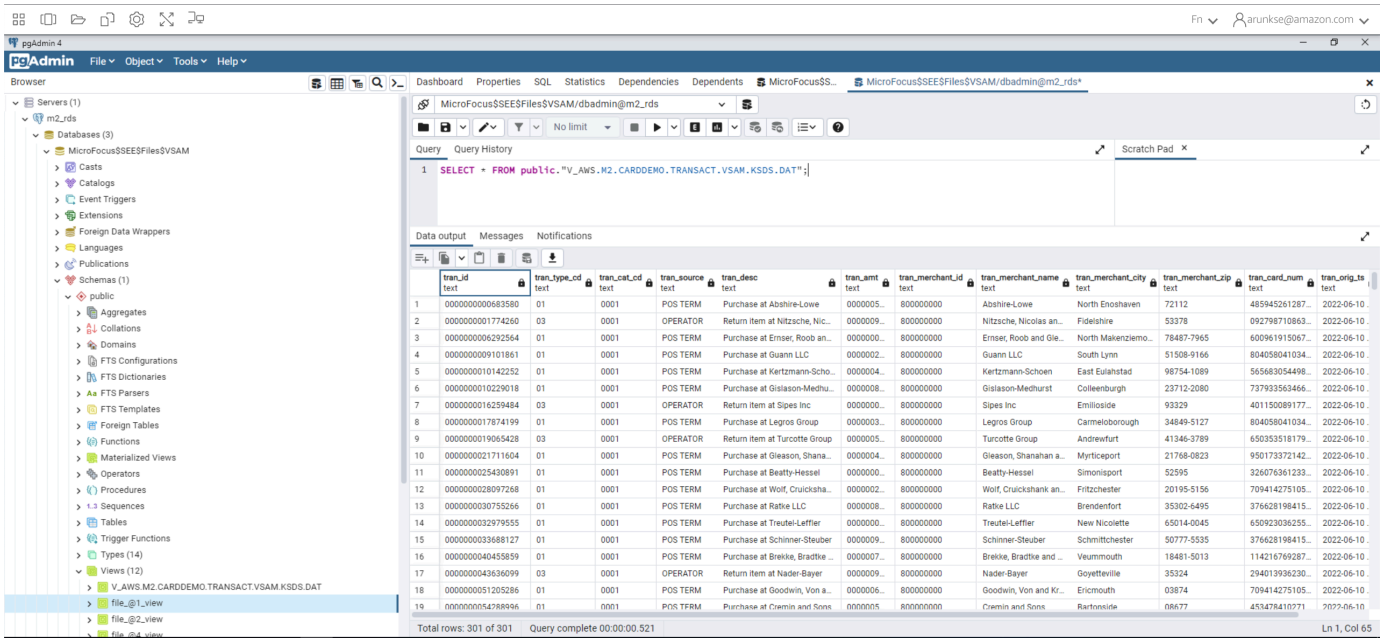
```
##
      ## The below command creates database view for VSAM file
 AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS
      ## using the STR file CVTRA05Y.str
      ##

      dbfhview -create -struct:"D:\PhotonUser\My Files\Home Folder\MFED\str
\CVTRA05Y.str" -name:V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT -file:sql://
ESPACDatabase/VSAM/AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT?folder=/DATA

      ##
      ## Output:
      ##

      Micro Focus Database File Handler - View Generation Tool Version 8.0.00
      Copyright (C) 1984-2022 Micro Focus. All rights reserved.

      VGN0017I Using structure definition 'TRAN-RECORD-DEFAULT'
      VGN0022I View 'V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT' installed in
 datastore 'sql://espacdatabase/VSAM'
      VGN0002I The operation completed successfully
```

Étape 5 : Afficher les ensembles de données de Rocket Software (anciennement Micro Focus) sous forme de tableaux et de colonnes

Au cours de cette étape, connectez-vous à la base de données `pgAdmin` afin de pouvoir exécuter des requêtes pour afficher les ensembles de données tels que les tables et les colonnes.

- Connectez-vous à la base de données à `MicroFocus$SEE$Files$VSAM` l'aide de pgAdmin et interrogez la vue de base de données que vous avez créée à l'étape 4.

```
SELECT * FROM public."V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT";
```



Modifiez des ensembles de données à l'aide des outils de fichiers de données Rocket Software (anciennement Micro Focus) dans Enterprise Developer

Vous pouvez consulter et modifier des ensembles de données dans AWS Mainframe Modernization à l'aide du moteur d'exécution Rocket Software pour tous les ensembles de données migrés.
Les étapes décrites dans ce document vous guideront tout au long du processus d'accès aux ensembles de données à l'aide des outils de fichiers de données.
Cela vous permet de visualiser et de modifier les ensembles de données migrés selon vos besoins.

Rubriques

- Prérequis
- Outils de fichiers de données Launch Rocket Software (anciennement Micro Focus)
- Modifier les ensembles de données VSAM stockés dans la base de données MFDBFH
- Modifier les ensembles de données non VSAM stockés dans la base de données MFDBFH

- Modifier les ensembles de données VSAM et non-VSAM stockés dans le système de fichiers (EFS/FSx)

Prérequis

Avant de commencer, vous devez avoir déployé une application avec les ensembles de données importé dans le cadre du service AWS Mainframe Modernization à l'aide du moteur Rocket Software.

Pour continuer à modifier les ensembles de données, vous devez effectuer l'étape 1, étape 2, et (éventuellement) Étape 3 depuis la the section called "Afficher les ensembles de données sous forme de tables dans Enterprise Developer" page pour configurer la connexion ODBC, et la banque de données Micro Focus (c'est-à-dire`MFDBFH`).

> **Important**
>
> Ce guide part du principe que vous utilisez Amazon Aurora Postgres comme banque de données Micro Focus () `MFDBFH` pour stocker les données de votre application.

Outils de fichiers de données Launch Rocket Software (anciennement Micro Focus)

Une fois les conditions requises remplies, vous lancez les outils de fichiers de données Micro Focus en configurant la variable d'`MFDBFH_CONFIG`environnement pour accéder aux ensembles de données stockés dans la base de données (`MFDBFH`).

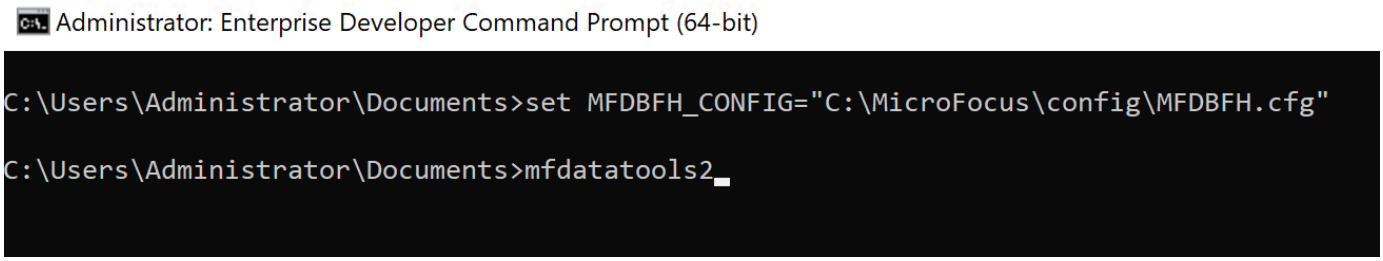
Pour ce faire,

1. Connectez-vous au poste de travail Micro Focus Enterprise Developer et lancez l'invite de commande Enterprise Developer (64 bits) depuis le menu Démarrer.
2. Définissez la variable d'`MFDBFH_CONFIG`environnement avec le chemin complet de votre `MFDBCH.cfg` fichier.

   ```
   set MFDBFH_CONFIG="C:\MicroFocus\config\MFDBFH.cfg"
   ```

3. Lancez Micro Focus Data File Tools depuis la ligne de commande Enterprise Developer à l'aide de la commande suivante.

```
mfdatatools2
```



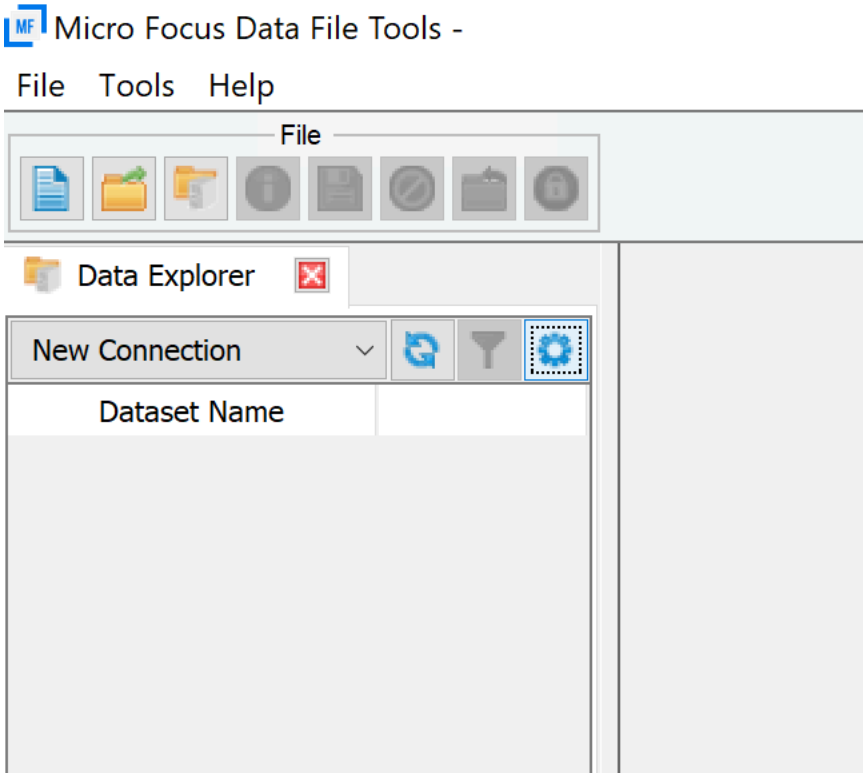
Cela ouvre les outils de fichiers de données Micro Focus dans une fenêtre séparée.

Modifier les ensembles de données VSAM stockés dans la base de données MFDBFH

Lorsque vous lancez les outils de fichiers de données Micro Focus, vous ouvrez un ensemble de données VSAM qui est stocké dans la banque de données Micro Focus.

Pour ce faire,

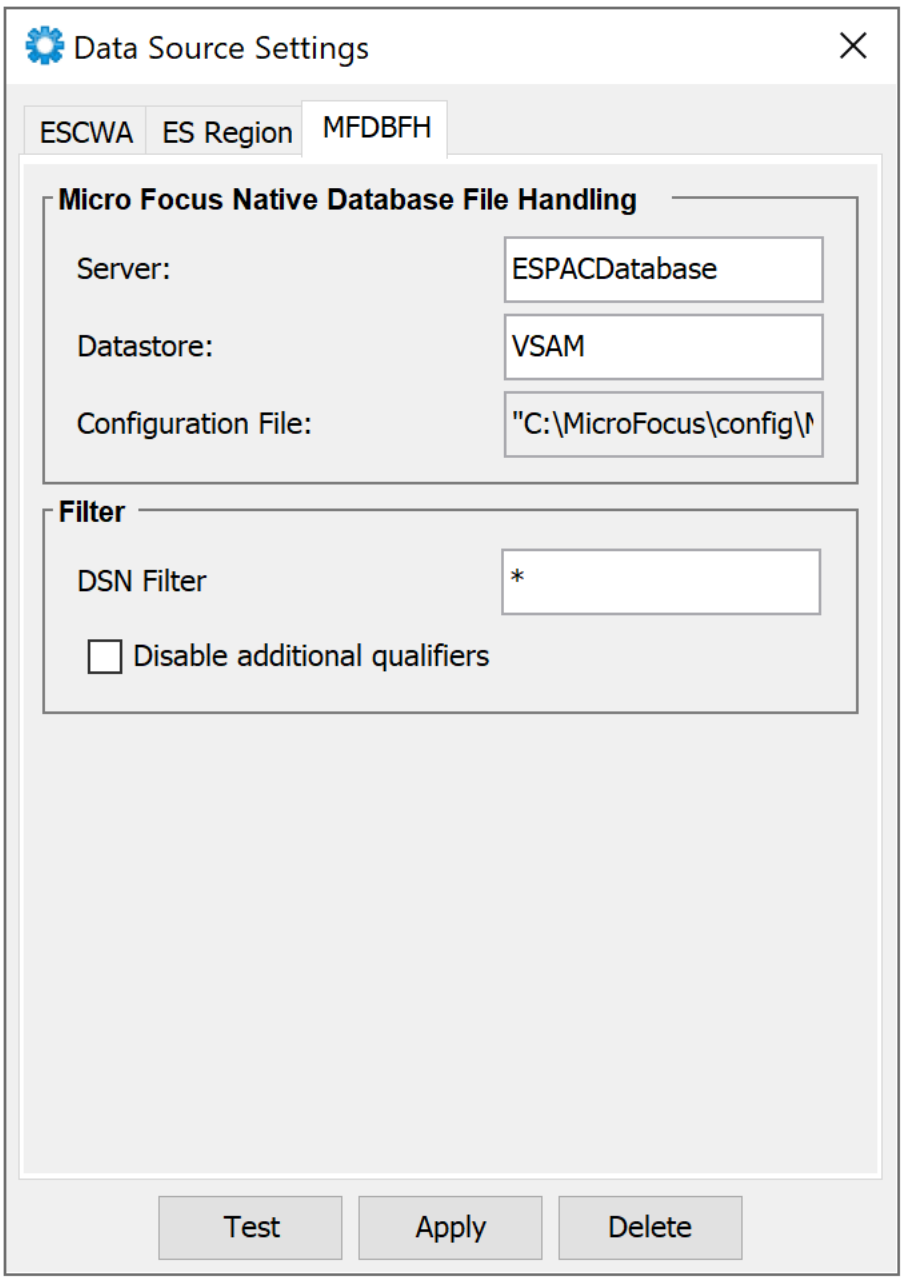
1. Dans le menu Fichier de la fenêtre Outils de fichiers de données Micro Focus, choisissez Data Explorer.
2. Dans la section Explorateur de données, choisissez Paramètres (icône en forme de roue dentée) pour configurer une nouvelle connexion. Cela ouvre une fenêtre de paramètres de source de données.



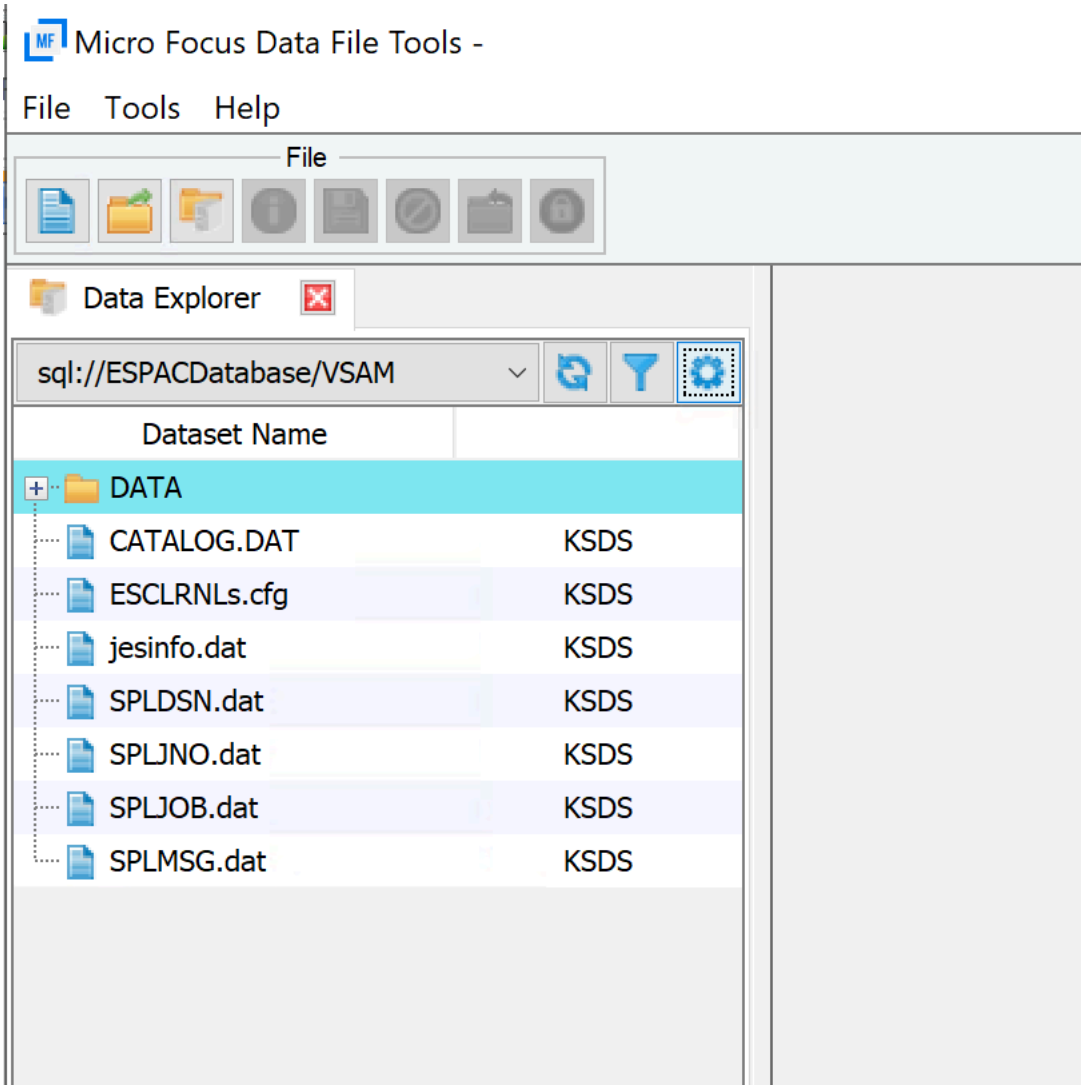
3. Dans la fenêtre Paramètres de la source de données, choisissez l'onglet MFDBFH et entrez les valeurs suivantes :

   - Serveur : `ESPACDatabase`
   - Banque de données : `VSAM`

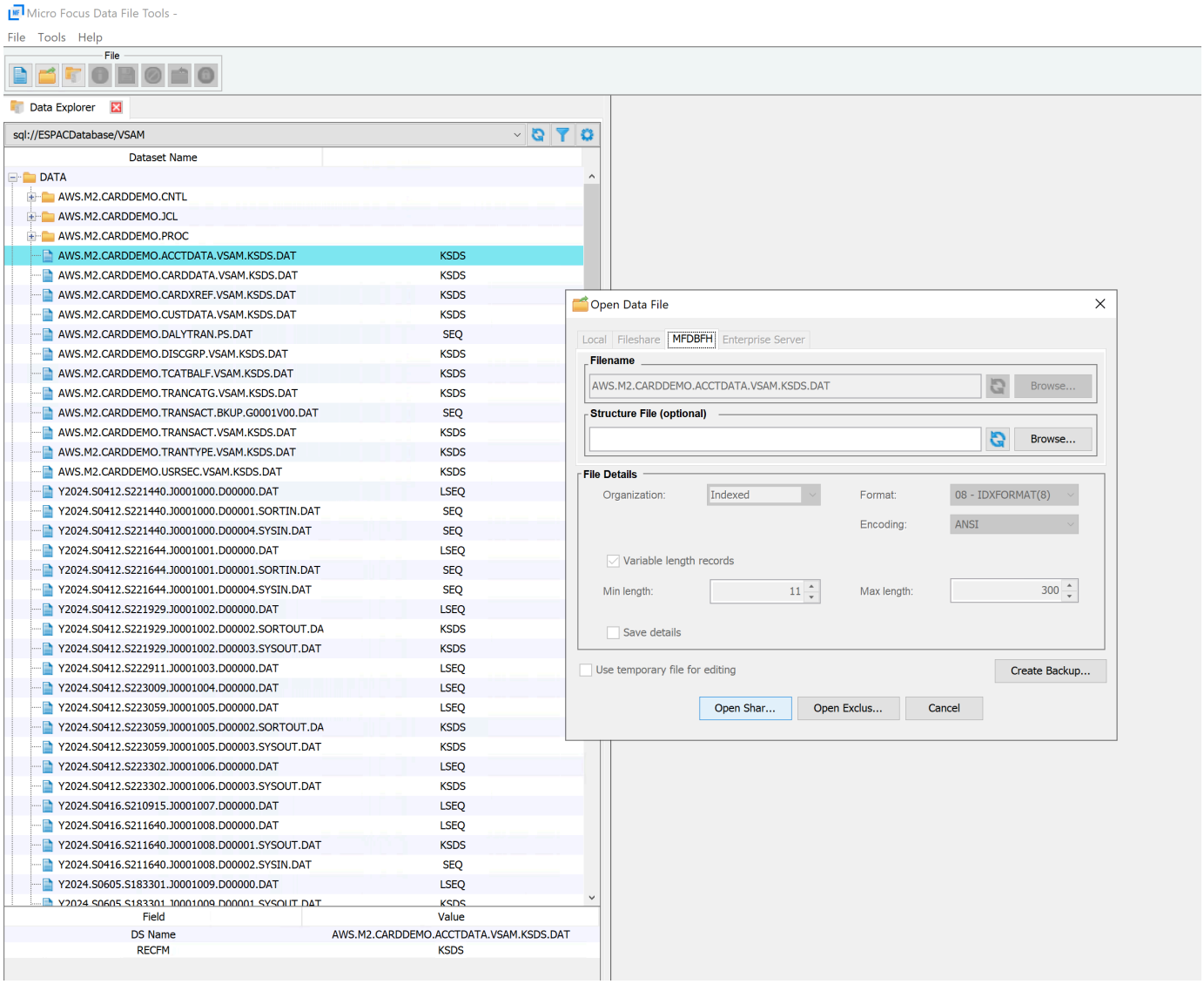
   Choisissez Appliquer pour enregistrer la configuration.



L'explorateur de données affiche désormais tous les ensembles de données stockés dans`MFDBFH`.



4. Développez le chemin relatif DATA et double-cliquez sur le jeu de données VSAM que vous souhaitez ouvrir.
5. Dans la fenêtre Open Data File, choisissez Open Shared ou Open Exclusive pour ouvrir le jeu de données.



Vous pouvez désormais consulter ou modifier l'ensemble de données ouvertes.

Modifier les ensembles de données non VSAM stockés dans la base de données MFDBFH

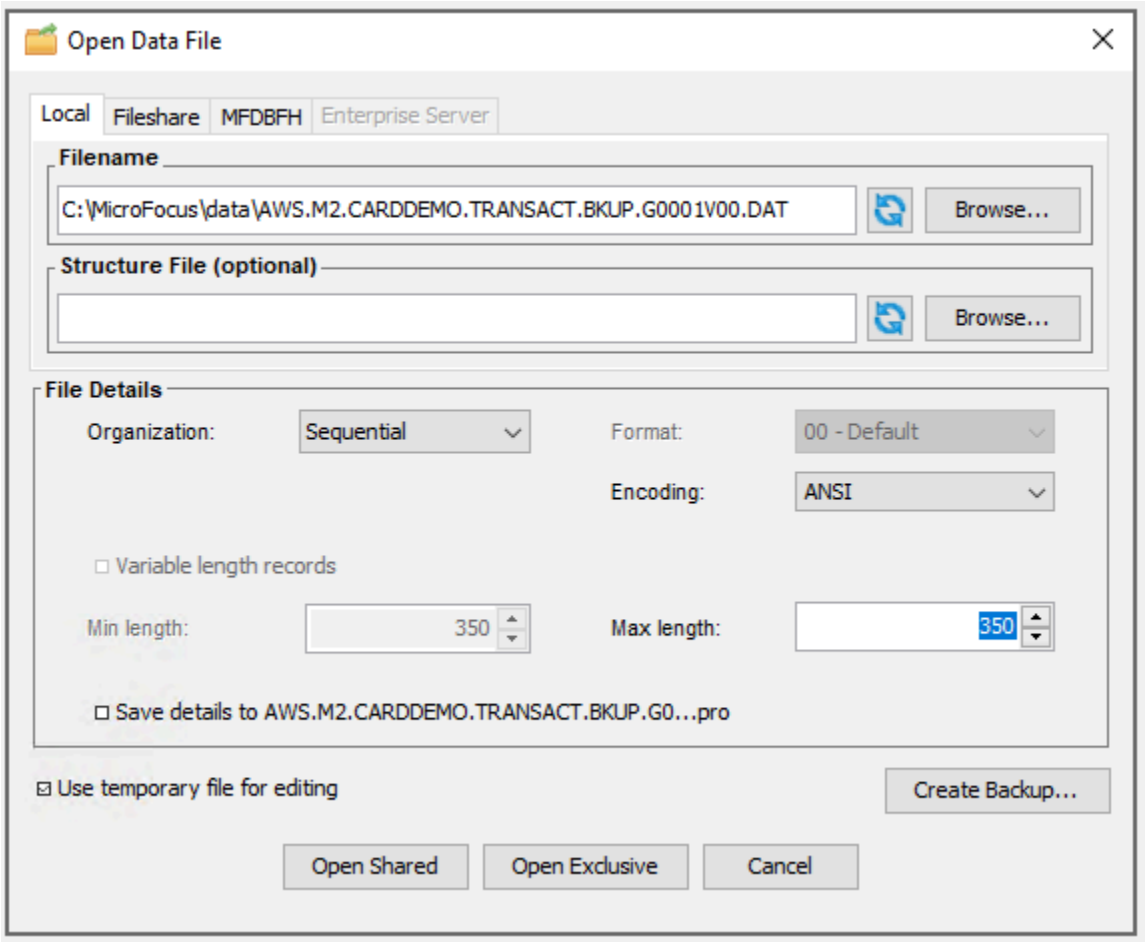
Si vous souhaitez modifier des ensembles de données non VSAM, vous ouvrez un ensemble de données non VSAM stocké dans la banque de données Micro Focus.

Pour ce faire,

1. À partir de l'invite de commande Enterprise Developer (64 bits), exécutez la `dbfhdeploy data extract` commande pour télécharger l'ensemble de données non VSAM sur votre système de fichiers local.

   > **Note**
   >
   > Avant d'exécuter cette commande, assurez-vous d'avoir défini la variable d'`MFDBFH_CONFIG`environnement avec le chemin complet de votre `MFDBFH.cfg` fichier.
   >
   > ```
   > dbfhdeploy data extract sql://ESPACDatabase/VSAM/
   > AWS.M2.CARDDEMO.TRANSACT.BKUP.G0001V00.DAT?folder=/DATA C:\MicroFocus\data
   > \AWS.M2.CARDDEMO.TRANSACT.BKUP.G0001V00.DAT
   > ```

2. Lancez Micro Focus Data File Tools depuis le menu Démarrer.
3. Dans le menu Fichier de Micro Focus Data File Tools, choisissez Ouvrir, puis Fichier de données.
4. Dans la fenêtre Open Data File, parcourez l'ensemble de données téléchargé dans votre système de fichiers local. Modifiez les détails du fichier selon vos besoins. Choisissez ensuite Open Shared ou Open Exclusive pour ouvrir le jeu de données.



Vous pouvez désormais consulter ou modifier l'ensemble de données ouvertes.

Les ensembles de données modifiés ou mis à jour peuvent être réimportés dans la banque de données Micro Focus en suivant les étapes the section called "Importer des ensembles de données pour les applications" ou en utilisant l'utilitaire de ligne de commande dbfhdeploy.

Modifier les ensembles de données VSAM et non-VSAM stockés dans le système de fichiers (EFS/FSx)

Vous pouvez également ouvrir un ensemble de données stocké dans un système de fichiers.

Pour ce faire,

1. Montez le EFS/FSx système de fichiers dans l' EC2 instance Enterprise Developer.
2. Utilisez les outils de fichiers de données Micro Focus pour parcourir et ouvrir les ensembles de données depuis le système de fichiers.

Tutoriels pour Rocket Software (anciennement Micro Focus)

Les didacticiels de cette section vous aident à commencer à configurer diverses tâches dans le moteur d'exécution Rocket Software pour le service de modernisation des AWS mainframes. Ces didacticiels concernent la configuration d'un exemple d'application, CI/CD pipelines, utilisation de modèles avec Rocket Enterprise Developer et configuration d'Enterprise Analyzer.

Rubriques

- Tutoriel : Configuration de la version Rocket Software (anciennement Micro Focus) pour l' BankDemo exemple d'application
- Tutoriel : Configuration d'un CI/CD pipeline à utiliser avec Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)
- Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer
- Tutoriel : Utiliser des modèles avec Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)
- Tutoriel : Configuration d'Enterprise Analyzer sur la version 2.0 AppStream
- Tutoriel : Configurer Rocket Enterprise Developer sur AppStream 2.0

Tutoriel : Configuration de la version Rocket Software (anciennement Micro Focus) pour l' BankDemo exemple d'application

AWS La modernisation du mainframe vous permet de configurer des builds (et des pipelines integration/continuous delivery (CI/CD (continus) pour vos applications migrées. Ces builds et pipelines utilisent AWS CodeBuild AWS CodeCommit, et AWS CodePipeline pour fournir ces fonctionnalités. CodeBuild est un service de génération entièrement géré qui compile votre code source, exécute des tests unitaires et produit des artefacts prêts à être déployés. CodeCommit est un service de contrôle de version qui vous permet de stocker et de gérer de manière privée des référentiels Git dans le cloud. AWS CodePipeline est un service de livraison continue qui vous permet de modéliser, de visualiser et d'automatiser les étapes nécessaires à la publication de votre logiciel.

Ce didacticiel explique AWS CodeBuild comment compiler l' BankDemo exemple de code source de l'application à partir d'Amazon S3, puis exporter le code compilé vers Amazon S3.

AWS CodeBuild est un service d'intégration continue entièrement géré qui compile le code source, exécute des tests et produit des progiciels prêts à être déployés. Vous pouvez utiliser des

CodeBuild environnements de génération préemballés ou créer des environnements de génération personnalisés utilisant vos propres outils de génération. Ce scénario de démonstration utilise la deuxième option. Il s'agit d'un environnement de CodeBuild construction qui utilise une image Docker préemballée.

> **Important**
>
> Avant de démarrer votre projet de modernisation du mainframe, nous vous recommandons de vous renseigner sur le AWS Migration Acceleration Program (MAP) pour mainframe ou de contacter des spécialistes du AWS mainframe pour connaître les étapes nécessaires à la modernisation d'une application mainframe.

Rubriques

- Prérequis
- Étape 1 : Partagez les actifs de construction avec le AWS compte
- Étape 2 : créer des compartiments Amazon S3
- Étape 3 : Création du fichier de spécifications de construction
- Étape 4 : télécharger les fichiers sources
- Étape 5 : créer des politiques IAM
- Étape 6 : Création d'un rôle IAM
- Étape 7 : associer les politiques IAM au rôle IAM
- Étape 8 : Création du CodeBuild projet
- Étape 9 : démarrer la construction
- Étape 10 : Télécharger les artefacts de sortie
- Nettoyage des ressources

Prérequis

Avant de commencer ce didacticiel, remplissez les conditions préalables suivantes.

- Téléchargez l'BankDemo exemple d'application et décompressez-le dans un dossier. Le dossier source contient les programmes COBOL et les copybooks, ainsi que les définitions. Il contient également un dossier JCL à titre de référence, bien que vous n'ayez pas besoin de compiler JCL. Le dossier contient également les méta-fichiers nécessaires à la compilation.

- Dans la console AWS Mainframe Modernization, sélectionnez Tools. Dans Analyse, développement et création d'actifs, choisissez Partager des actifs avec mon compte AWS.

Étape 1 : Partagez les actifs de construction avec le AWS compte

Au cours de cette étape, vous vous assurez de partager les actifs de construction avec votre AWS compte, en particulier dans la région où les actifs sont utilisés.

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le volet de navigation de gauche, sélectionnez Outils.
3. Dans Analyse, développement et création d'actifs, choisissez Partager des actifs avec mon AWS compte.

> **Important**
>
> Vous devez effectuer cette étape une fois dans chaque AWS région où vous avez l'intention d'effectuer des builds.

Étape 2 : créer des compartiments Amazon S3

Au cours de cette étape, vous allez créer deux compartiments Amazon S3. Le premier est un compartiment d'entrée pour contenir le code source, et l'autre est un compartiment de sortie pour contenir la sortie de compilation. Pour plus d'informations, consultez la section Création, configuration et utilisation des compartiments Amazon S3 dans le guide de l'utilisateur Amazon S3.

1. Pour créer le compartiment d'entrée, connectez-vous à la console Amazon S3 et choisissez Create bucket.
2. Dans Configuration générale, donnez un nom au compartiment et spécifiez l' Région AWS endroit où vous souhaitez le créer. Par exemple`codebuild-regionId-accountId-input-bucket`, le nom est « où se `regionId` trouve le Région AWS compartiment » et « où `accountId` est votre Compte AWS identifiant ».

> **Note**
>
> Si vous créez le bucket dans un pays différent Région AWS de celui de l'est des États-Unis (Virginie du Nord), spécifiez le `LocationConstraint` paramètre. Pour plus d'informations, consultez Create Bucket dans le manuel Amazon Simple Storage Service API Reference.

3. Conservez tous les autres paramètres et choisissez Create bucket.
4. Répétez les étapes 1 à 3 pour créer le compartiment de sortie. Par exemple`codebuild-regionId-accountId-output-bucket`, le nom est « où se `regionId` trouve le Région AWS compartiment » et « où `accountId` est votre Compte AWS identifiant ».

   Quels que soient les noms que vous choisissez pour ces compartiments, veillez à les utiliser tout au long de ce didacticiel.

Étape 3 : Création du fichier de spécifications de construction

Au cours de cette étape, vous créez un fichier de spécifications de construction,. Ce fichier fournit les commandes de construction et les paramètres associés, au format YAML, CodeBuild pour exécuter la génération. Pour plus d'informations, reportez-vous à la section Référence des spécifications de construction CodeBuild dans le Guide de AWS CodeBuild l'utilisateur.

1. Créez un fichier nommé `buildspec.yml` dans le répertoire que vous avez décompressé comme condition préalable.
2. Ajoutez le contenu suivant au fichier et enregistrez-le. Aucune modification n'est requise pour ce fichier.

```
version: 0.2
env:
  exported-variables:
    - CODEBUILD_BUILD_ID
    - CODEBUILD_BUILD_ARN
phases:
  install:
    runtime-versions:
      python: 3.7
  pre_build:
    commands:
```

```
      - echo Installing source dependencies...
      - ls -lR $CODEBUILD_SRC_DIR/source
  build:
    commands:
      - echo Build started on `date`
      - /start-build.sh -Dbasedir=$CODEBUILD_SRC_DIR/source -Dloaddir=
$CODEBUILD_SRC_DIR/target
  post_build:
    commands:
      - ls -lR $CODEBUILD_SRC_DIR/target
      - echo Build completed on `date`
artifacts:
  files:
    - $CODEBUILD_SRC_DIR/target/**
```

Voici`CODEBUILD_BUILD_ID`, `CODEBUILD_BUILD_ARN`$`CODEBUILD_SRC_DIR/source`, et $`CODEBUILD_SRC_DIR/target` les variables d'environnement sont-elles disponibles dans CodeBuild. Pour plus d'informations, consultez la section Variables d'environnement dans les environnements de génération.

À ce stade, votre répertoire devrait ressembler à ceci.

```
(root directory name)
    |-- build.xml
    |-- buildspec.yml
    |-- LICENSE.txt
    |-- source
         |... etc.
```

3. Compressez le contenu du dossier dans un fichier nomméBankDemo`.zip`.. Pour ce didacticiel, vous ne pouvez pas compresser le dossier. Compressez plutôt le contenu du dossier dans le fichierBankDemo`.zip`.

Étape 4 : télécharger les fichiers sources

Au cours de cette étape, vous chargez le code source de l' BankDemo exemple d'application dans votre compartiment d'entrée Amazon S3.

1. Connectez-vous à la console Amazon S3 et choisissez Buckets dans le volet de navigation de gauche. Choisissez ensuite le compartiment d'entrée que vous avez créé précédemment.

2. Sous Objets, choisissez Charger.
3. Dans la section Fichiers et dossiers, choisissez Ajouter des fichiers.
4. Accédez à votre `BankDemo.zip` fichier et sélectionnez-le.
5. Choisissez Charger.

Étape 5 : créer des politiques IAM

Au cours de cette étape, vous allez créer deux politiques IAM. Une politique autorise AWS Mainframe Modernization à accéder à l'image Docker qui contient les outils de compilation de Rocket Software et à l'utiliser. Cette politique n'est pas personnalisée pour les clients. L'autre politique autorise AWS Mainframe Modernization à interagir avec les compartiments d'entrée et de sortie, ainsi qu'avec les CloudWatch journaux Amazon qui CodeBuild en sont générés.

Pour en savoir plus sur la création d'une stratégie IAM, consultez la section Modification des politiques IAM dans le Guide de l'utilisateur IAM.

Pour créer une politique d'accès aux images Docker

1. Dans la console IAM, copiez le document de stratégie suivant et collez-le dans l'éditeur de stratégie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "arn:aws:ecr:*:673918848628:repository/m2-enterprise-build-
tools"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::aws-m2-repo-*-<region>-prod"
        }
    ]
}
```

2. Donnez un nom à la politique, par exemple,`m2CodeBuildPolicy`.

Pour créer une politique permettant à la modernisation du AWS mainframe d'interagir avec les buckets et les journaux

1. Dans la console IAM, copiez le document de stratégie suivant et collez-le dans l'éditeur de stratégie. Assurez-vous de mettre `regionId` à jour le Région AWS, et `accountId` votre Compte AWS.

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Action": [
                  "logs:CreateLogGroup",
                  "logs:CreateLogStream",
                  "logs:PutLogEvents"
              ],
              "Resource": [
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project",
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project:*"
              ],
              "Effect": "Allow"
          },
          {
              "Action": [
                  "s3:PutObject",
                  "s3:GetObject",
                  "s3:GetObjectVersion",
```

```
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::codebuild-regionId-accountId-input-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-input-bucket/*",
                "arn:aws:s3:::codebuild-regionId-accountId-output-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-output-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

2. Donnez un nom à la politique, par exemple,`BankdemoCodeBuildRolePolicy`.

Étape 6 : Création d'un rôle IAM

Au cours de cette étape, vous créez un nouveau rôle IAM qui permet d'interagir avec les AWS ressources CodeBuild à votre place, après avoir associé les politiques IAM que vous avez créées précédemment à ce nouveau rôle IAM.

Pour plus d'informations sur la création d'un rôle de service, consultez la section Création d'un rôle pour déléguer des autorisations à un AWS service dans le guide de l'utilisateur IAM,.

1. Connectez-vous à la console IAM et choisissez Rôles dans le volet de navigation de gauche.
2. Sélectionnez Create role (Créer un rôle).
3. Sous Type d'entité fiable, choisissez le service AWS.
4. Sous Cas d'utilisation pour d'autres services AWS CodeBuild, choisissez, puis choisissez CodeBuildà nouveau.
5. Choisissez Next (Suivant).
6. Sur la page Add permissions (Ajouter des autorisations), sélectionnez Next (Suivant). Vous attribuez une politique au rôle ultérieurement.
7. Sous Détails du rôle, saisissez un nom pour le rôle, par exemple,`BankdemoCodeBuildServiceRole`.
8. Sous Sélectionner les entités de confiance, vérifiez que le document de politique ressemble à ce qui suit :

```
{
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "codebuild.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        }
```

9. Sélectionnez Create role (Créer un rôle).

Étape 7 : associer les politiques IAM au rôle IAM

Au cours de cette étape, vous associez les deux politiques IAM que vous avez créées précédemment au rôle `BankdemoCodeBuildServiceRole` IAM.

1. Connectez-vous à la console IAM et choisissez Rôles dans le volet de navigation de gauche.
2. Dans Rôles, choisissez le rôle que vous avez créé précédemment, par exemple`BankdemoCodeBuildServiceRole`.
3. Dans Politiques d'autorisations, choisissez Ajouter des autorisations, puis Joindre des politiques.
4. Dans Autres politiques d'autorisation, choisissez les politiques que vous avez créées précédemment, par exemple, `m2CodeBuildPolicy` et`BankdemoCodeBuildRolePolicy`.
5. Choisissez Attach Policies (Attacher des politiques).

Étape 8 : Création du CodeBuild projet

Au cours de cette étape, vous créez le CodeBuild projet.

1. Connectez-vous à la CodeBuild console et choisissez Create build project.
2. Dans la section Configuration du projet, saisissez un nom pour le projet, par exemple,`codebuild-bankdemo-project`.

3. Dans la section Source, pour Source provider, choisissez Amazon S3, puis choisissez le bucket d'entrée que vous avez créé précédemment, par exemple,`codebuild-regionId-accountId-input-bucket`.
4. Dans le champ Clé d'objet S3 ou dossier S3, entrez le nom du fichier zip que vous avez chargé dans le compartiment S3. Dans ce cas, le nom du fichier est`bankdemo.zip`.
5. Dans la section Environnement, choisissez Image personnalisée.
6. Dans le champ Type d'environnement, sélectionnez Linux.
7. Sous Registre d'images, choisissez Autre registre.
8. Dans le champ URL du registre externe,

   - Pour Rocket Software v9 : Entrez`673918848628.dkr.ecr.us-west-1.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1`. Si vous utilisez une autre AWS région avec Rocket Software v9, vous pouvez également spécifier `673918848628.dkr.ecr.<m2-region>.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1` où se <m2-region>trouve une AWS région dans laquelle le service de modernisation du AWS mainframe est disponible (par exemple,`eu-west-3`).
   - Pour Rocket Software v8 : Enter `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:8.0.9.R1`
   - Pour Rocket Software v7 : Enter `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:7.0.R10`
9. Sous Rôle de service, choisissez Rôle de service existant, puis dans le champ ARN du rôle, choisissez le rôle de service que vous avez créé précédemment, par exemple,`BankdemoCodeBuildServiceRole`.
10. Dans la section Buildspec, choisissez Utiliser un fichier buildspec.
11. Dans la section Artifacts, sous Type, choisissez Amazon S3, puis choisissez votre compartiment de sortie, par exemple,`codebuild-regionId-accountId-output-bucket`.
12. Dans le champ Nom, entrez le nom d'un dossier dans le compartiment dans lequel vous souhaitez contenir les artefacts de sortie de génération, par exemple`bankdemo-output.zip`.
13. Sous Emballage des artefacts, sélectionnez Zip.
14. Choisissez Créer un projet de génération.

Étape 9 : démarrer la construction

Au cours de cette étape, vous lancez la construction.

1. Connectez-vous à la CodeBuild console.
2. Dans le volet de navigation de gauche, choisissez Créer des projets.
3. Choisissez le projet de construction que vous avez créé précédemment, par exemple`codebuild-bankdemo-project`.
4. Choisissez Démarrer la génération.

Cette commande lance le build. La compilation s'exécute de manière asynchrone. La sortie de la commande est un fichier JSON qui inclut l'identifiant de l'attribut. Cet attribut id est une référence à l'identifiant de CodeBuild construction de la construction que vous venez de démarrer. Vous pouvez consulter l'état du build dans la CodeBuild console. Vous pouvez également consulter les journaux détaillés de l'exécution de la compilation dans la console. Pour plus d'informations, voir Afficher les informations de construction détaillées dans le guide de AWS CodeBuild l'utilisateur.

Lorsque la phase en cours est TERMINÉE, cela signifie que votre compilation s'est terminée avec succès et que vos artefacts compilés sont prêts sur Amazon S3.

Étape 10 : Télécharger les artefacts de sortie

Au cours de cette étape, vous devez télécharger les artefacts de sortie depuis Amazon S3. L'outil de construction de Rocket Software peut créer plusieurs types d'exécutables différents. Dans ce didacticiel, il génère des objets partagés.

1. Connectez-vous à la console Amazon S3.
2. Dans la section Buckets role="bold">, choisissez le nom de votre bucket de sortie, par exemple,. `codebuild-regionId-accountId-output-bucket`
3. Choisissez Download role="bold">.
4. Décompressez le fichier téléchargé. Accédez au dossier cible pour voir les artefacts de construction. Il s'agit notamment des objets partagés `.so` Linux.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Supprimez les compartiments S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez Supprimer un compartiment dans le guide de l'utilisateur d'Amazon Simple Storage Service.

- Supprimez les politiques IAM que vous avez créées pour ce didacticiel. Pour plus d'informations, consultez la section Suppression des politiques IAM dans le guide de l'utilisateur IAM.
- Supprimez le rôle IAM que vous avez créé pour ce didacticiel. Pour plus d'informations, consultez Suppression de rôles ou de profils d'instance dans le guide de l'utilisateur IAM.
- Supprimez le CodeBuild projet que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un projet de construction CodeBuild dans le guide de AWS CodeBuild l'utilisateur.

Tutoriel : Configuration d'un CI/CD pipeline à utiliser avec Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)

Ce didacticiel vous explique comment importer, modifier, compiler et exécuter l' BankDemo exemple d'application dans Rocket Enterprise Developer, puis comment valider vos modifications pour déclencher un CI/CD oléoduc.

Table des matières

- Prérequis
- Création CI/CD infrastructure de base du pipeline
- Créez un AWS CodeCommit référentiel et CI/CD pipeline
  - Exemple de fichier déclencheur YAML config_git.yml
- Création d'Enterprise Developer AppStream 2.0
- Configuration et test pour les développeurs d'entreprise
  - Cloner le BankDemo CodeCommit référentiel dans Enterprise Developer
  - Création d'un projet COBOL sur BankDemo mainframe et création d'une application
  - Création d'un environnement BankDemo CICS et batch local pour les tests
  - Démarrez le serveur BANKDEMO depuis Enterprise Developer
  - Démarrez le terminal Rumba 3270
  - Exécuter une BankDemo transaction
  - Arrêtez le serveur BANKDEMO depuis Enterprise Developer
- Exercice 1 : Améliorer le calcul du prêt dans l'application BANKDEMO
  - Ajouter une règle d'analyse des prêts à Enterprise Developer Code Analysis
  - Étape 1 : Effectuer une analyse du code pour le calcul du prêt
  - Étape 2 : Modifier la carte CICS BMS et le programme COBOL et tester

- Étape 3 : Ajouter le calcul du montant total dans le programme COBOL
- Étape 4 : valider les modifications et exécuter le pipeline CI/CD
- Exercice 2 : Extraire le calcul du prêt dans BankDemo l'application
  - Étape 1 : Refactoriser la routine de calcul des prêts dans une section COBOL
  - Étape 2 : Extraire la routine de calcul des prêts dans un programme COBOL autonome
  - Étape 3 : valider les modifications et exécuter le CI/CD pipeline
- Nettoyage des ressources

Prérequis

Téléchargez les fichiers suivants.

- `basic-infra.yaml`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `pipeline.yaml`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `m2-code-sync-function.zip`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `config_git.yml`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `BANKDEMO-source.zip`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `BANKDEMO-exercise.zip`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).

L'objectif de chaque fichier est le suivant :

`basic-infra.yaml`

Ce AWS CloudFormation modèle crée l'infrastructure de base nécessaire pour CI/CD pipeline : VPC, compartiments Amazon S3, etc.

`pipeline.yaml`

Ce AWS CloudFormation modèle est utilisé par une fonction Lambda pour lancer la pile de pipelines. Assurez-vous que ce modèle se trouve dans un compartiment Amazon S3 accessible au public. Ajoutez le lien vers ce compartiment comme valeur par défaut pour le `PipelineTemplateURL` paramètre dans le `basic-infra.yaml` modèle.

`m2-code-sync-function.zip`

Cette fonction Lambda crée le CodeCommit référentiel, la structure de répertoire basée sur`config_git.yaml`, et lance la pile de pipelines à l'aide de. `pipeline.yaml` Assurez-vous que ce fichier zip est disponible dans un compartiment Amazon S3 accessible au public dans tous les pays Régions AWS où la modernisation des AWS mainframes est prise en charge. Nous vous recommandons de stocker le fichier dans un compartiment Région AWS et de le répliquer dans tous les compartiments. Régions AWS Utilisez une convention de dénomination pour le compartiment avec un suffixe identifiant le compartiment spécifique Région AWS (par exemple,`m2-cicd-deployment-source-eu-west-1`) et ajoutez le préfixe `m2-cicd-deployment-source` comme valeur par défaut pour le paramètre `DeploymentSourceBucket` et formez le compartiment complet en utilisant la fonction de AWS CloudFormation substitution `!Sub {DeploymentSourceBucket}-${AWS::Region}` tout en faisant référence à ce compartiment dans le `basic-infra.yaml` modèle de ressource. `SourceSyncLambdaFunction`

`config_git.yml`

CodeCommit définition de la structure du répertoire. Pour de plus amples informations, veuillez consulter Exemple de fichier déclencheur YAML config_git.yml.

`BANKDEMO-source.zip`.

BankDemo code source et fichier de configuration créés à partir du CodeCommit référentiel.

`BANKDEMO-exercise.zip`.

BankDemo source pour les exercices didacticiels créés à partir du CodeCommit référentiel.

Création CI/CD infrastructure de base du pipeline

Utilisez le AWS CloudFormation modèle `basic-infra.yaml` pour créer la pile d'infrastructure de base du pipeline CI/CD via la AWS CloudFormation console. Cette pile crée des compartiments Amazon S3 dans lesquels vous pouvez télécharger le code et les données de votre application, ainsi qu'une AWS Lambda fonction de support pour créer d'autres ressources nécessaires, telles qu'un AWS CodeCommit référentiel et un AWS CodePipeline pipeline.

> **Note**
>
> Pour lancer cette pile, vous avez besoin d'autorisations pour administrer IAM, Amazon S3, Lambda AWS CloudFormation et d'autorisations d'utilisation. AWS KMS

1. Connectez-vous à la AWS CloudFormation console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/cloudformation.
2. Créez une nouvelle pile avec l'une des options suivantes :
   - Sélectionnez Créer une pile. C'est la seule option disponible si une pile est en cours d'exécution.
   - Sur la page Stacks, choisissez Create Stack. Cette option n'est visible que si aucune pile n'est en cours d'exécution.
3. Sur la page Spécifier le modèle :
   - Dans Préparer le modèle, sélectionnez Le modèle est prêt.
   - Dans Spécifier le modèle, choisissez l'URL Amazon S3 comme source du modèle et entrez l'une des informations suivantes URLs en fonction de votre Région AWS.
     - `https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/basic-infra.yaml`
     - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/basic-infra.yaml`
   - Pour accepter vos paramètres, choisissez Next.

   La page Créer une pile s'ouvre.

Specify stack details

Stack name

Stack name

m2-cicd-pipeline-foundation

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Networking Configuration

Do you want to use an existing VPC in your account?

If you select 'Yes', then you must provide the VPC ID and the Subnet IDs.

No (Create one)

Which VPC ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID in a different AZ should be used for HA?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Enter the CIDR block that should be used for the new VPC

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

11.11.0.0/16

CIDR bits for creating subnets. Choose 5 for /27, 6 for /26, 7 for /25, 8 for /24 range

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

5

Deployment Configuration

Name of the S3 bucket which contains the source files for this stack deployment

Don't change unless you know what you are doing.

m2

Name of the source package file for the infrastructure Lambda function

Don't change unless you know what you are doing.

cicd/mf/m2-code-sync-function.zip

Full URL of the pipeline CloudFormation template file

Don't change unless you know what you are doing.

https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/pipeline.yaml

What name prefix to use for the new S3 buckets?

A name prefix for the S3 buckets that will be created by this stack.

mf

Cancel Previous Next

Effectuez les modifications suivantes :

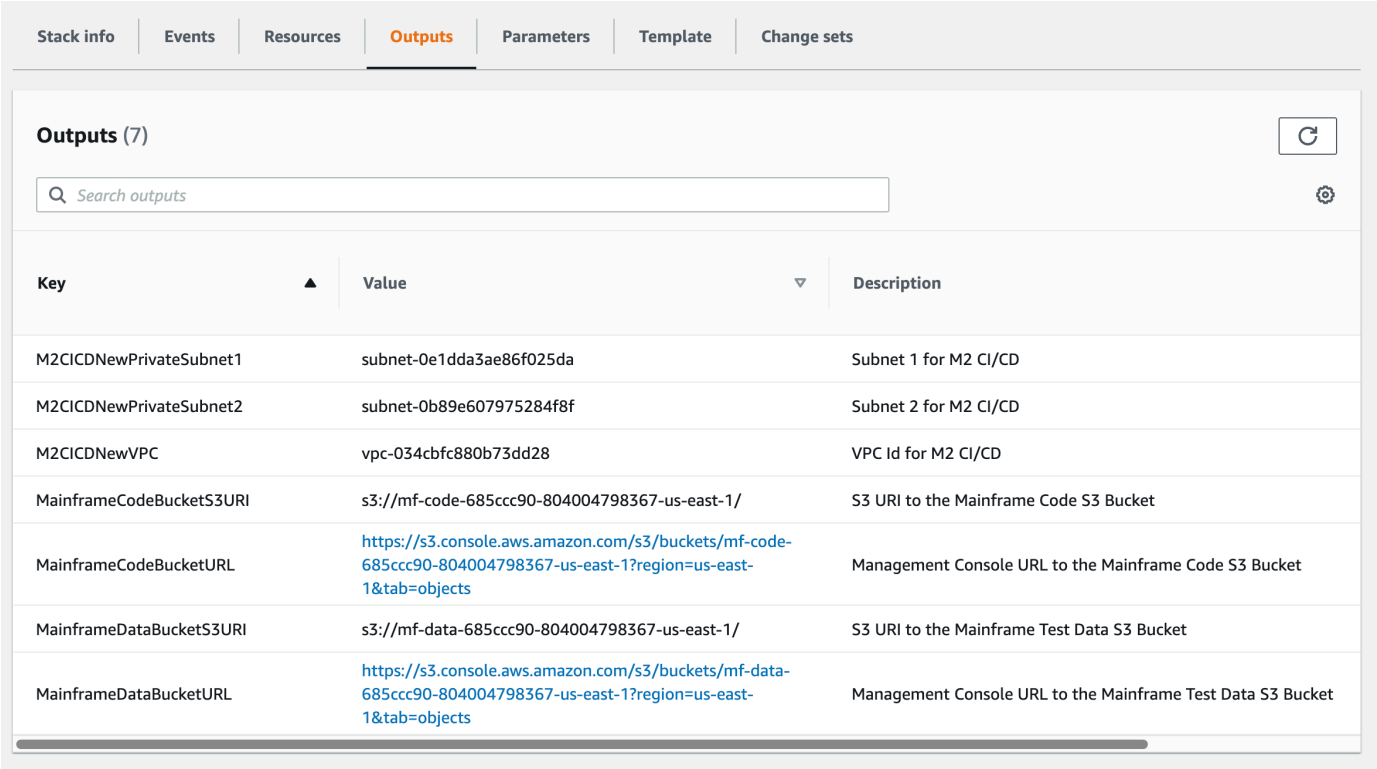
- Fournissez les valeurs appropriées pour le nom de la pile et les paramètres de configuration réseau.
- La plupart des paramètres des configurations de déploiement sont préremplis de manière appropriée, vous n'avez donc pas besoin de les modifier. Selon votre modèle Région AWS, remplacez le AWS CloudFormation modèle de pipeline par l'un des modèles Amazon S3 suivants URLs.
  - `https://m2-us-east-1.s3.amazonaws.com/cicd/mf/pipeline.yaml`
  - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/pipeline.yaml`
- Choisissez Next (Suivant).

> **Note**
>
> Ne modifiez pas les valeurs des paramètres par défaut, sauf si vous avez vous-même modifié le AWS CloudFormation modèle.

4. Dans Configurer les options de pile, choisissez Next.
5. Dans Fonctionnalités, choisissez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM pour autoriser la création AWS CloudFormation d'un rôle IAM en votre nom. Sélectionnez Créer la pile.

> **Note**
>
> Le provisionnement de cette pile peut prendre de 3 à 5 minutes.

6. Une fois la pile créée avec succès, accédez à la section Sorties de la pile nouvellement provisionnée. Vous y trouverez le compartiment Amazon S3 dans lequel vous devez télécharger le code de votre mainframe et les fichiers dépendants.



Créez un AWS CodeCommit référentiel et CI/CD pipeline

Au cours de cette étape, vous créez un CodeCommit référentiel et configurez un CI/CD pile de pipeline en appelant une fonction Lambda qui appelle AWS CloudFormation pour créer la pile de pipelines.

1. Téléchargez l'BankDemo exemple d'application sur votre ordinateur local.
2. Téléchargez `bankdemo.zip` depuis votre machine locale vers le compartiment Amazon S3 créé dansCréation CI/CD infrastructure de base du pipeline.
3. Téléchargement `config_git.yml`.
4. Modifiez le `config_git.yml` si nécessaire, comme suit :

   - Ajoutez votre propre nom de référentiel cible, votre branche cible et votre message de validation.

```
repository-config:
  target-repository: bankdemo-repo
  target-branch: main
  commit-message: Initial commit for bankdemo-repo main branch
```

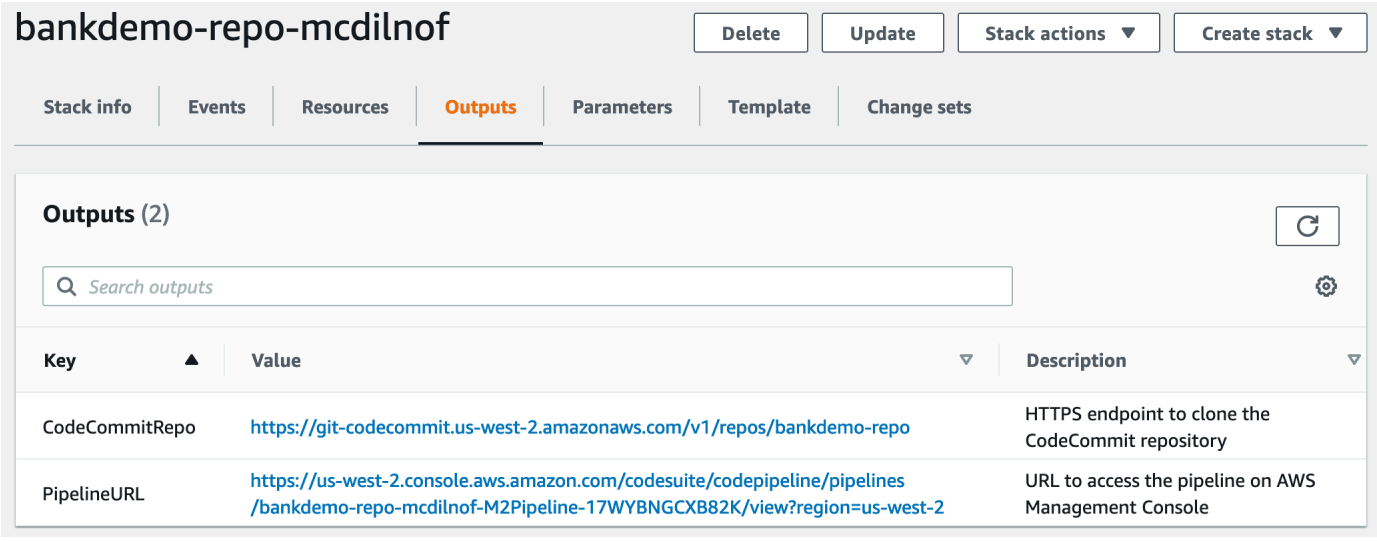
- Ajoutez l'adresse e-mail à laquelle vous souhaitez recevoir des notifications.

```
pipeline-config:
  # Send pipeline failure notifications to these email addresses
  alert-notifications:
    - myname@mycompany.com
  # Send notifications for manual approval before production deployment to these
 email addresses
  approval-notifications:
    - myname@mycompany.com
```

5. Téléchargez le `config_git.yml` fichier contenant la définition de la structure des dossiers du CodeCommit référentiel dans le compartiment Amazon S3 créé dansCréation CI/CD infrastructure de base du pipeline. Cela invoquera la fonction Lambda qui approvisionnera automatiquement le référentiel et le pipeline.

   Cela créera un CodeCommit référentiel avec le nom fourni dans le `config_git.yml` fichier `target-repository` défini ; par exemple,`bankdemo-repo`.

   La fonction Lambda créera également CI/CD les pipelines s'empilent AWS CloudFormation. La AWS CloudFormation pile aura le même préfixe que le `target-repository` nom fourni, suivi d'une chaîne aléatoire (par exemple`bankdemo-repo-`*`01234567`*. Vous trouverez l'URL du CodeCommit référentiel et l'URL permettant d'accéder au pipeline créé dans la console AWS de gestion.



6. Si la création CodeCommit du référentiel est terminée, CI/CD le pipeline sera déclenché immédiatement pour effectuer une opération complète CI/CD.

7. Une fois que le fichier a été transféré, il déclenche automatiquement le pipeline qui sera créé, déployé en phase de préparation, exécutera des tests et attendra l'approbation manuelle avant de le déployer dans l'environnement de production.

Exemple de fichier déclencheur YAML config_git.yml

```
repository-config:
  target-repository: bankdemo-repo
  target-branch: main
  commit-message: Initial commit for bankdemo-repo main branch
  directory-structure:
    - '/':
        files:
        - build.xml
        - '*.yaml'
        - '*.yml'
        - '*.xml'
        - 'LICENSE.txt'
        readme: |
          # Root Folder
          - 'build.xml' : Build configuration for the application
    - tests:
        files:
        - '*.py'
        readme: |
          # Test Folder
          - '*.py' : Test scripts
    - config:
        files:
        - 'BANKDEMO.csd'
        - 'BANKDEMO.json'
        - 'BANKDEMO_ED.json'
        - 'dfhdrdat'
        - 'ESPGSQLXA.dll'
        - 'ESPGSQLXA64.so'
        - 'ESPGSQLXA64_S.so'
        - 'EXTFH.cfg'
        - 'm2-2021-04-28.normal.json'
        - 'MFDBFH.cfg'
        - 'application-definition-template-config.json'
        readme: |
          # Config Folder
```

```
          This folder contains the application configuration files.
          - 'BANKDEMO.csd'     : CICS Resource definitions export file
          - 'BANKDEMO.json'    : Enterprise Server configuration
          - 'BANKDEMO_ED.json' : Enterprise Server configuration for ED
          - 'dfhdrdat'         : CICS resource definition file
          - 'ESPGSQLXA.dll'    : XA switch module Windows
          - 'ESPGSQLXA64.so'   : XA switch module Linux
          - 'ESPGSQLXA64_S.so' : XA switch module Linux
          - 'EXTFH.cfg'        : Micro Focus File Handler configuration
          - 'm2-2021-04-28.normal.json' : M2 request document
          - 'MFDBFH.cfg'       : Micro Focus Database File Handler
          - 'application-definition-template-config.json' : Application definition for
 M2
    - source:
        subdirs:
        - .settings:
            files:
              - '.bms.mfdirset'
              - '.cbl.mfdirset'
        - copybook:
            files:
              - '*.cpy'
              - '*.inc'
            readme: |
              # Copy folder
              This folder contains the source for COBOL copy books, PLI includes, ...
              - .cpy COBOL copybooks
              - .inc PLI includes
#        - ctlcards:
#            files:
#              - '*.ctl'
#              - 'KBNKSRT1.txt'
#            readme: |
#              # Control Card folder
#              This folder contains the source for Batch Control Cards
#              - .ctl Control Cards
        - ims:
            files:
              - '*.dbd'
              - '*.psb'
            readme: |
              # ims folder
              This folder contains the IMS DB source files with the extensions
              - .dbd for IMS DBD source
```

```
              - .psb for IMS PSB source
        - jcl:
            files:
              - '*.jcl'
              - '*.ctl'
              - 'KBNKSRT1.txt'
              - '*.prc'
            readme: |
              # jcl folder
              This folder contains the JCL source files with the extensions
              - .jcl
#        - proclib:
#            files:
#              - '*.prc'
#            readme: |
#              # proclib folder
#              This folder contains the JCL procedures referenced via PROCLIB
 statements in the JCL with extensions
#              - .prc
        - rdbms:
            files:
              - '*.sql'
            readme: |
              # rdbms folder
              This folder contains any DB2 related source files with extensions
              - .sql for any kind of SQL source
        - screens:
            files:
            - '*.bms'
            - '*.mfs'
            readme: |
              # screens folder
              This folder contains the screens source files with the extensions
              - .bms for CICS BMS screens
              - .mfs for IMS MFS screens
            subdirs:
            - .settings:
                files:
                    - '*.bms.mfdirset'
        - cobol:
            files:
              - '*.cbl'
              - '*.pli'
            readme: |
```

```
              # source folder
              This folder contains the program source files with the extensions
              - .cbl for COBOL source
              - .pli for PLI source
            subdirs:
            - .settings:
                files:
                   - '*.cbl.mfdirset'
    - tests:
        files:
        - 'test_script.py'
        readme: |
          # tests Folder
          This folder contains the application test scripts
pipeline-config:
  alert-notifications:
    - myname@mycompany.com
  approval-notifications:
    - myname@mycompany.com
```

Création d'Enterprise Developer AppStream 2.0

Pour configurer Rocket Enterprise Developer sur AppStream 2.0, consultezTutoriel : Configurer Rocket Enterprise Developer sur AppStream 2.0.

Pour connecter le CodeCommit référentiel à Enterprise Developer, utilisez le nom indiqué `target-repository` dansExemple de fichier déclencheur YAML config_git.yml.

Configuration et test pour les développeurs d'entreprise

Rubriques

- Cloner le BankDemo CodeCommit référentiel dans Enterprise Developer
- Création d'un projet COBOL sur BankDemo mainframe et création d'une application
- Création d'un environnement BankDemo CICS et batch local pour les tests
- Démarrez le serveur BANKDEMO depuis Enterprise Developer
- Démarrez le terminal Rumba 3270
- Exécuter une BankDemo transaction
- Arrêtez le serveur BANKDEMO depuis Enterprise Developer

Connectez-vous à l'instance Enterprise Developer AppStream 2.0 que vous avez créée dansCréation d'Enterprise Developer AppStream 2.0.

1. Démarrez Enterprise Developer à partir du démarrage de Windows. Choisissez Micro Focus Enterprise Developer, puis choisissez Enterprise Developer pour Eclipse. Si vous commencez pour la première fois, cela peut prendre un certain temps.
2. Dans le lanceur Eclipse, dans Workspace : entrez `C:\Users\<username>\workspace` puis choisissez Launch.

   > **Note**
   >
   > Assurez-vous de choisir le même emplacement après vous être reconnecté à l'instance AppStream 2.0. La sélection de l'espace de travail n'est pas permanente.

3. Dans Bienvenue, choisissez Open COBOL Perspective. Cela ne sera affiché que la première fois pour un nouvel espace de travail.

Cloner le BankDemo CodeCommit référentiel dans Enterprise Developer

1. Choisissez Fenêtre/Perspective /Perspective ouverte/Autre... /Git.
2. Choisissez Cloner un dépôt Git.
3. Dans Clone Git Repository, entrez les informations suivantes :

   - Dans Location URI, entrez l'URL HTTPS du CodeCommit référentiel.

     > **Note**
     >
     > Copiez l'URL de clonage HTTPS du CodeCommit référentiel dans la console AWS de gestion et collez-la ici. L'URI sera divisée en chemins d'hôte et de référentiel.

   - CodeCommit Référencez les informations d'identification de l'utilisateur dans Authentication User and Password et choisissez Stocker dans Secure Store.
4. Dans Sélection de branche, choisissez Branche principale, puis Suivant.
5. Dans Destination locale, dans Répertoire, entrez `C:\Users\<username>\workspace` et choisissez Terminer.

Le processus de clonage est terminé lorsqu'il `BANKDEMO [main]` est affiché dans la vue Git Repositories.

Création d'un projet COBOL sur BankDemo mainframe et création d'une application

1. Passez à la perspective COBOL.
2. Dans Project, désactivez la création automatique.
3. Dans Fichier, choisissez Nouveau, puis Mainframe COBOL Project.
4. Dans New Mainframe COBOL Project, entrez les informations suivantes :
   - Dans Nom du projet, entrez`BankDemo`.
   - Choisissez le modèle Micro Focus [64 bits].
   - Choisissez Finish (Terminer).
5. Dans COBOL Explorer, développez le nouveau BankDemo projet.

   > **Note**
   >
   > `[BANKDEMO main]`entre crochets indique que le projet est connecté au BankDemo CodeCommit dépôt local.

6. Si l'arborescence n'affiche pas les entrées relatives aux programmes COBOL, aux copybooks, aux sources BMS et aux fichiers JCL, choisissez Actualiser dans le menu contextuel du BankDemo projet.
7. BankDemo Dans le menu contextuel, choisissez Propriétés/Micro Focus/Paramètres du projet/COBOL :
   - Choisissez Jeu de caractères - ASCII.
   - Choisissez Appliquer, puis Fermer.
8. Si la génération de la source BMS et COBOL ne démarre pas immédiatement, vérifiez dans le menu Projet que l'option Créer automatiquement est activée.

   La sortie de compilation sera affichée dans la vue de la console et devrait être terminée au bout de quelques minutes avec des messages `BUILD SUCCESSFUL` et`Build finished with no errors`.

   L' BankDemo application doit maintenant être compilée et prête pour une exécution locale.

Création d'un environnement BankDemo CICS et batch local pour les tests

1. Dans COBOL Explorer, développezBANKDEMO / config.
2. Dans l'éditeur, ouvrezBANKDEMO_ED.json.
3. Recherchez la chaîne ED_Home= et modifiez le chemin pour pointer vers le projet Enterprise Developer, comme suit :D:\\<username>\\workspace\\BANKDEMO. Notez l'utilisation de barres obliques doubles (\\) dans la définition du chemin.
4. Enregistrez et fermez le fichier .
5. Choisissez Server Explorer.
6. Dans le menu contextuel par défaut, choisissez Ouvrir la page d'administration. La page d'administration de Micro Focus Enterprise Server s'ouvre dans le navigateur par défaut.
7. Pour les sessions AppStream 2.0 uniquement, apportez les modifications suivantes afin de préserver votre région Enterprise Server locale pour les tests locaux :

   - Dans Directory Server/Default, choisissez PROPERTIES/Configuration.
   - Remplacez l'emplacement du référentiel parD:\<username>\My Files\Home Folder \MFDS.

   > **Note**
   >
   > Vous devez effectuer les étapes 5 à 8 après chaque nouvelle connexion à une instance AppStream 2.0.

8. Dans Directory Server/Default, choisissez Importer, puis effectuez les étapes suivantes :

   - À l'étape 1 : Type d'importation, choisissez JSON, puis Next.
   - À l'étape 2 : Télécharger, cliquez pour télécharger le fichier dans un carré bleu.
   - Dans Choisir le fichier à télécharger, entrez :
     - Nom du fichier :D:\<username>\workspace\BANKDEMO\config\BANKDEMO_ED.json.
     - Choisissez Ouvrir.
   - Choisissez Next (Suivant).
   - À l'étape 3 : les régions effacent les ports des points de terminaison.
   - Choisissez Next (Suivant).
   - À l'étape 4 : Importer, choisissez Importer.

- Choisissez Finish (Terminer).

La liste affiche désormais un nouveau nom de serveur`BANKDEMO`.

Démarrez le serveur BANKDEMO depuis Enterprise Developer

1. Choisissez Enterprise Developer.
2. Dans Server Explorer, choisissez Default, puis Refresh dans le menu contextuel.

   La liste des serveurs devrait désormais également afficher BANKDEMO.
3. Choisissez BANKDEMO.
4. Dans le menu contextuel, choisissez Associer au projet, puis BANKDEMO.
5. Dans le menu contextuel, choisissez Démarrer.

   La vue de la console doit afficher le journal du démarrage du serveur.

   Si le message `BANKDEMO CASSI5030I PLTPI Phase 2 List(PI) Processing Completed` s'affiche, le serveur est prêt à tester l'application CICS BANKDEMO.

Démarrez le terminal Rumba 3270

1. Depuis le démarrage de Windows, lancez Micro Focus Rumba+ Desktop/Rumba+ Desktop.
2. Dans Bienvenue, choisissez CREATE NEW SESSION/Mainframe Display.
3. Dans Mainframe Display, choisissez Connection/ Configure.
4. Dans Configuration de session, choisissez Connection/ TN3270.
5. Dans Nom d'hôte/adresse, choisissez Insérer et entrez l'adresse IP`127.0.0.1`.
6. Dans Port Telnet, entrez port`6000`.
7. Choisissez Appliquer.
8. Choisissez Se connecter.

   L'écran de bienvenue du CICS affiche un écran avec le message de ligne 1 :`This is the Micro Focus MFE CICS region BANKDEMO`.
9. Appuyez sur Ctrl+Shift+Z pour effacer l'écran.

Exécuter une BankDemo transaction

1. Dans un écran vide, entrez`BANK`.
2. À l'écran BANK10, dans le champ de saisie du nom d'utilisateur... :, entrez `guest` et appuyez sur Entrée.
3. À l'écran BANK20, dans le champ de saisie situé avant Calculer le coût d'un prêt, entrez `/` (barre oblique) et appuyez sur Entrée.
4. Dans l'écran BANK70 :

   - Dans Le montant que vous souhaitez emprunter... :, entrez`10000`.
   - Dans À un taux d'intérêt de... :, entrez`5.0`.
   - Dans Depuis combien de mois... :, entrez`10`.
   - Appuyez sur Entrée.

   Le résultat suivant doit être affiché :

   ```
   Resulting monthly payment.............:   $1023.06
   ```

   Ceci termine la configuration de l'application BANKDEMO dans Enterprise Developer.

Arrêtez le serveur BANKDEMO depuis Enterprise Developer

1. Dans Server Explorer, choisissez Default, puis Refresh dans le menu contextuel.
2. Choisissez BANKDEMO.
3. Dans le menu contextuel, choisissez Stop.

   La vue de la console doit afficher le journal de l'arrêt du serveur.

   Si le message `Server: BANKDEMO stopped successfully` s'affiche, le serveur s'est correctement arrêté.

Exercice 1 : Améliorer le calcul du prêt dans l'application BANKDEMO

Rubriques

- Ajouter une règle d'analyse des prêts à Enterprise Developer Code Analysis

- Étape 1 : Effectuer une analyse du code pour le calcul du prêt
- Étape 2 : Modifier la carte CICS BMS et le programme COBOL et tester
- Étape 3 : Ajouter le calcul du montant total dans le programme COBOL
- Étape 4 : valider les modifications et exécuter le pipeline CI/CD

Dans ce scénario, vous allez suivre le processus consistant à apporter un exemple de modification au code, à le déployer et à le tester.

Le service des prêts souhaite un nouveau champ sur l'écran de calcul du prêt BANK7 0 pour afficher le montant total du prêt. Cela nécessite une modification de l'écran BMS MBANK7 0.CBL, l'ajout d'un nouveau champ et le programme de gestion d'écran correspondant SBANK7 0P.CBL avec les cahiers associés. En outre, la routine de calcul des prêts dans BBANK7 0P.CBL doit être étendue avec la formule supplémentaire.

Pour terminer cet exercice, assurez-vous de remplir les conditions préalables suivantes.

- Téléchargez BANKDEMO-exercise.zip pour`D:\PhotonUser\My Files\Home Folder`.
- Extrayez le fichier zip dans`D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise`.
- Créez un dossier`D:\PhotonUser\My Files\Home Folder\AnalysisRules`.
- Copiez le fichier `Loan+Calculation+Update.General-1.xml` de règles du `BANKDEMO-exercise` dossier vers`D:\PhotonUser\My Files\Home Folder\AnalysisRules`.

> **Note**
>
> Les modifications de code dans *.CBL et *.CPY sont signalées par EXER01 dans les colonnes 1 à 6 pour cet exercice.

Ajouter une règle d'analyse des prêts à Enterprise Developer Code Analysis

Les règles d'analyse définies dans Rocket Enterprise Analyzer peuvent être exportées depuis Enterprise Analyzer et importées dans Enterprise Developer pour exécuter les mêmes règles d'analyse dans toutes les sources du projet Enterprise Developer.

1. Ouvrir `Window/Preferences/Micro Focus/COBOL/Code Analysis/Rules`.

2. Choisissez Modifier... et entrez le nom du dossier `D:\PhotonUser\My Files \Home Folder\AnalysisRules` contenant le fichier de règles`Loan+Calculation +Update.General-1.xml`.
3. Choisissez Finish (Terminer).
4. Choisissez Appliquer, puis Fermer.
5. Dans le menu contextuel du projet BANKDEMO, choisissez Code Analysis.

   Vous devriez voir une entrée pour la mise à jour du calcul du prêt.

Étape 1 : Effectuer une analyse du code pour le calcul du prêt

Avec la nouvelle règle d'analyse, nous voulons identifier les programmes COBOL et les lignes de code qui correspondent aux modèles de recherche, `*LOAN*` ainsi que `*RATE*` dans les expressions`*PAYMENT*`, les instructions et les variables. Cela vous aidera à naviguer dans le code et à identifier les modifications de code requises.

1. Dans le menu contextuel du projet BANKDEMO, choisissez Analyse du code/Mise à jour du calcul du prêt.

   Cela exécutera la règle de recherche et listera les résultats dans un nouvel onglet appelé Analyse du code. L'analyse est terminée lorsque la barre de progression verte en bas à droite disparaît.

   L'onglet Analyse du code doit afficher une liste détaillée des instructions`BBANK20P.CBL`, expressions `BBANK70P.CBL` et `SBANK70P.CBL` variables correspondant aux modèles de recherche, et chacune répertoriant chacune d'entre elles.

   En regardant le résultat, seuls les littéraux déplacés correspondent au modèle de recherche. `BBANK20P.CBL` Ce programme peut donc être ignoré.
2. Dans la barre de menu de l'onglet, choisissez - Icône pour tout réduire.
3. Développez `SBANK70P.CBL` et sélectionnez n'importe quelle ligne dans n'importe quel ordre en double-cliquant pour voir comment cela ouvrira le code source et surlignera la ligne sélectionnée dans le code source. Vous reconnaîtrez également que toutes les lignes de source identifiées sont marquées.

Étape 2 : Modifier la carte CICS BMS et le programme COBOL et tester

Nous allons d'abord modifier la carte `MBANK70.BMS` BMS, le programme de gestion d'écran `SBANK70P.CBL` et le cahier `CBANKDAT.CPY` pour afficher le nouveau champ. Pour éviter tout codage inutile dans cet exercice, les modules source modifiés sont disponibles dans le `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01` dossier. Normalement, un développeur utilise les résultats de l'analyse du code pour naviguer et modifier les sources. Si vous avez le temps et souhaitez effectuer les modifications manuelles, utilisez les informations fournies dans *Modification manuelle dans MBANK7 0.BMS et SBANK7 0P.CBL (facultatif) *.

Pour des modifications rapides, copiez les fichiers suivants :

1. `..\BANKDEMO-exercise\Exercis01\screens\MBANK70.BMS` à `D:\PhotonUser\workspace\bankdemo\source\screens`.
2. `.\BANKDEMO-exercise\Exercis01\cobol\SBANK70P.CBL` à `D:\PhotonUser\workspace\bankdemo\source\cobol`.
3. `..\BANKDEMO-exercise\Exercis01\copybook\CBANKDAT.CPY` à `D:\PhotonUser\workspace\bankdemo\source\copybook`.
4. Pour vous assurer que tous les programmes concernés par les modifications sont compilés, sélectionnez Project/Clean.../Cleantous les projets.

Pour les modifications manuelles apportées à `MBANK70.BMS` et`SBANK70P.CBL`, procédez comme suit :

- Pour une modification manuelle de la `MBANK70.BMS` source BMS, ajoutez après le `PAYMENT` champ :
  - TXT09 avec les mêmes attributs que TXT08 et valeur INITIALE « Montant total du prêt »
  - TOTAL avec les mêmes attributs que PAYMENT

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes :

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270

3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................:`.

4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

Étape 3 : Ajouter le calcul du montant total dans le programme COBOL

Dans un deuxième temps, nous modifierons `BBANK70P.CBL` et ajouterons le calcul du montant total du prêt. La source préparée avec les modifications requises est disponible dans `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01` le dossier. Si vous avez le temps et souhaitez effectuer les modifications manuelles, utilisez les informations fournies dans *Modification manuelle dans BBANK7 0P.CBL (facultatif) *.

Pour un changement rapide, copiez le fichier suivant :

- `..\BANKDEMO-exercise\Exercis01\source\cobol\BBANK70P.CBL` à `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Pour apporter une modification manuelle à`BBANK70P.CBL`, procédez comme suit :

- Utilisez le résultat de l'analyse du code pour identifier les modifications requises.

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes :

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270
3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................: $10230.60`.

4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

Étape 4 : valider les modifications et exécuter le pipeline CI/CD

Validez les modifications dans le CodeCommit référentiel central et déclenchez le CI/CD pipeline pour créer, tester et déployer les modifications.

1. Dans le projet BANKDEMO, dans le menu contextuel, choisissez Team/Commit.
2. Dans l'onglet Git Staging, entrez le message de validation suivant :`Added Total Amount Calculation`.
3. Choisissez Commit et Push... .
4. Ouvrez la CodePipeline console et vérifiez l'état de l'exécution du pipeline.

> **Note**
>
> Si vous rencontrez un problème avec la fonction Commit ou Push de Teams ou Enterprise Developer, utilisez l'interface de ligne de commande Git Bash.

Exercice 2 : Extraire le calcul du prêt dans BankDemo l'application

Rubriques

- Étape 1 : Refactoriser la routine de calcul des prêts dans une section COBOL
- Étape 2 : Extraire la routine de calcul des prêts dans un programme COBOL autonome
- Étape 3 : valider les modifications et exécuter le CI/CD pipeline

Dans le prochain exercice, vous allez travailler sur un autre exemple de demande de modification. Dans ce scénario, le service des prêts souhaite réutiliser la routine de calcul des prêts de manière autonome. WebService La routine doit rester en COBOL et doit également être appelable à partir du programme CICS COBOL existant. `BBANK70P.CBL`

Étape 1 : Refactoriser la routine de calcul des prêts dans une section COBOL

Dans un premier temps, nous extrayons la routine de calcul du prêt dans une section COBOL. Cette étape est nécessaire pour extraire le code dans un programme COBOL autonome à l'étape suivante.

1. Ouvrez `BBANK70P.CBL` dans l'éditeur COBOL.

2. Dans l'éditeur, choisissez dans le menu contextuel Analyse du code/Mise à jour du calcul du prêt. Cela analysera uniquement la source actuelle à la recherche de modèles définis dans la règle d'analyse.
3. Dans le résultat de l'onglet Analyse du code, recherchez la première instruction `DIVIDE WS-LOAN-INTEREST BY 12` arithmétique.
4. Double-cliquez sur l'instruction pour accéder à la ligne source dans l'éditeur. Il s'agit du premier énoncé de la routine de calcul des prêts.
5. Marquez le bloc de code suivant pour que la routine de calcul du prêt soit extraite dans une section.

```
DIVIDE WS-LOAN-INTEREST BY 12
             GIVING WS-LOAN-INTEREST ROUNDED.
           COMPUTE WS-LOAN-MONTHLY-PAYMENT ROUNDED =
             ((WS-LOAN-INTEREST * ((1 + WS-LOAN-INTEREST)
                 ** WS-LOAN-TERM)) /
             (((1 + WS-LOAN-INTEREST) * WS-LOAN-TERM) - 1 ))
                 * WS-LOAN-PRINCIPAL.
EXER01     COMPUTE WS-LOAN-TOTAL-PAYMENT =
EXER01               (WS-LOAN-MONTHLY-PAYMENT * WS-LOAN-TERM).
```

6. Dans le menu contextuel de l'éditeur, choisissez Refactor/Extract to Section... .
7. Entrez le nom de la nouvelle section : LOAN-CALCULATION.
8. Choisissez OK.

   Le bloc de code marqué a maintenant été extrait dans la nouvelle `LOAN-CALCULATION` section et le bloc de code a été remplacé par l'`PERFROM LOAN-CALCULATIO`Ninstruction.

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes.

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270
3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................: $10230.60`.
4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

> **Note**
>
> Si vous souhaitez éviter les étapes ci-dessus pour extraire le bloc de code dans une section, vous pouvez copier la source modifiée pour l'étape 1 de `..\BANKDEMO-exercise\Exercis02\Step1\cobol\BBANK70P.CBL` vers`D:\PhotonUser\workspace\bankdemo\source\cobol`.

Étape 2 : Extraire la routine de calcul des prêts dans un programme COBOL autonome

À l'étape 2, le bloc de code de la `LOAN-CALCULATION` section sera extrait vers un programme autonome et le code d'origine sera remplacé par un code permettant d'appeler le nouveau sous-programme.

1. Ouvrez `BBANK70P.CBL` dans l'éditeur et recherchez la nouvelle `PERFORM LOAN-CALCULATION` déclaration créée à l'étape 1.
2. Placez le curseur dans le nom de la section. Il sera marqué en gris.
3. Dans le menu contextuel, sélectionnez Refactor->Extraire la section/le paragraphe vers le programme... .
4. Dans Extraire la section/le paragraphe à programmer, entrez le nouveau nom de fichier : LOANCALC.CBL.
5. Choisissez OK.

   Le nouveau `LOANCALC.CBL` programme s'ouvre dans l'éditeur.
6. Faites défiler la page vers le bas et passez en revue le code extrait et généré pour l'interface d'appel.
7. Sélectionnez l'éditeur avec `BBANK70P.CBL` et accédez à`LOAN-CALCULATION SECTION`. Vérifiez le code généré pour appeler le nouveau sous-programme`LOANCALC.CBL`.

> **Note**
>
> L'`CALL`instruction utilise `DFHEIBLK` et appelle `LOANCALC` avec `DFHCOMMAREA` des blocs de contrôle CICS. Comme nous voulons appeler le nouveau `LOANCALC.CBL` sous-programme un programme non-CICS, nous devons supprimer `DFHEIBLK` et `DFHCOMMAREA` annuler l'appel en commentant ou en supprimant.

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes.

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270
3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................: $10230.60`.
4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

> **Note**
>
> Si vous souhaitez éviter les étapes ci-dessus pour extraire le bloc de code dans une section, vous pouvez copier la source modifiée pour l'étape 1 depuis `..\BANKDEMO-exercise\Exercis02\Step2\cobol\BBANK70P.CBL` et `LOANCALC.CBL` versD:`\PhotonUser\workspace\bankdemo\source\cobol`.

Étape 3 : valider les modifications et exécuter le CI/CD pipeline

Validez les modifications dans le CodeCommit référentiel central et déclenchez le CI/CD pipeline pour créer, tester et déployer les modifications.

1. Dans le projet BANKDEMO, dans le menu contextuel, choisissez Team/Commit.
2. Dans l'onglet Git Staging
   - Ajoutez les étapes non mises en scène LOANCALC.CBL et LoanCalc.CBL.MFDirSet.
   - Entrez un message de validation :`Added Total Amount Calculation`.
3. Choisissez Commit et Push... .
4. Ouvrez la CodePipeline console et vérifiez l'état de l'exécution du pipeline.

> **Note**
>
> Si vous rencontrez un problème avec la fonction Commit ou Push de Teams ou Enterprise Developer, utilisez l'interface de ligne de commande Git Bash.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin qu'elles ne continuent pas à vous être facturées. Procédez comme suit :

- Supprimez le CodePipeline pipeline. Pour plus d'informations, voir Supprimer un pipeline CodePipeline dans le Guide de AWS CodePipeline l'utilisateur.
- Supprimez le CodeCommit référentiel. Pour plus d'informations, voir Supprimer un CodeCommit référentiel dans le Guide de AWS CodeCommit l'utilisateur.
- Supprimez le compartiment S3 ;. Pour plus d'informations, consultez Supprimer un compartiment dans le guide de l'utilisateur d'Amazon Simple Storage Service.
- Supprimez la AWS CloudFormation pile. Pour plus d'informations, voir Supprimer une pile sur la AWS CloudFormation console dans le Guide de AWS CloudFormation l'utilisateur.

Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer

AWS La modernisation du mainframe fournit plusieurs outils via Amazon AppStream 2.0. AppStream 2.0 est un service de streaming d'applications sécurisé et entièrement géré qui vous permet de diffuser des applications de bureau aux utilisateurs sans avoir à réécrire les applications. AppStream La version 2.0 fournit aux utilisateurs un accès instantané aux applications dont ils ont besoin avec une expérience utilisateur réactive et fluide sur l'appareil de leur choix. L'utilisation de la AppStream version 2.0 pour héberger des outils spécifiques au moteur d'exécution permet aux équipes d'application des clients d'utiliser les outils directement depuis leur navigateur Web, en interagissant avec les fichiers d'application stockés dans des compartiments ou des référentiels Amazon S3. CodeCommit

Pour plus d'informations sur la prise en charge des navigateurs dans la AppStream version 2.0, consultez la section Configuration système requise et prise en charge des fonctionnalités (navigateur Web) dans le guide d'administration Amazon AppStream 2.0. Si vous rencontrez des problèmes lors de l'utilisation de la AppStream version 2.0, consultez la section Résolution AppStream des problèmes liés aux utilisateurs de la AppStream version 2.0 dans le guide d'administration Amazon 2.0.

Ce document est destiné aux membres de l'équipe des opérations clients. Il décrit comment configurer les flottes et les piles Amazon AppStream 2.0 pour héberger les outils Rocket Enterprise Analyzer et Rocket Enterprise Developer utilisés dans le cadre de la modernisation des AWS

mainframes. Rocket Enterprise Analyzer est généralement utilisé pendant la phase d'évaluation et Rocket Enterprise Developer est généralement utilisé pendant la phase de migration et de modernisation de l'approche de modernisation du AWS mainframe. Si vous prévoyez d'utiliser à la fois Enterprise Analyzer et Enterprise Developer, vous devez créer des flottes et des piles distinctes pour chaque outil. Chaque outil nécessite son propre parc et sa propre pile, car leurs conditions de licence sont différentes.

> **Important**
>
> Les étapes de ce didacticiel sont basées sur le AWS CloudFormation modèle téléchargeable cfn-m2- appstream-fleet-ea-ed .yml.

Rubriques

- Prérequis
- Étape 1 : Obtenir les images AppStream 2.0
- Étape 2 : Création de la pile à l'aide du AWS CloudFormation modèle
- Étape 3 : créer un utilisateur dans la AppStream version 2.0
- Étape 4 : Connectez-vous à la AppStream version 2.0
- Étape 5 : vérifier les compartiments dans Amazon S3 (facultatif)
- Étapes suivantes
- Nettoyage des ressources

Prérequis

- Téléchargez le modèle : cfn-m2- appstream-fleet-ea-ed .yml.
- Obtenez l'ID de votre VPC et de votre groupe de sécurité par défaut. Pour plus d'informations sur le VPC par défaut, consultez Default VPCs dans le guide de l'utilisateur Amazon VPC. Pour plus d'informations sur le groupe de sécurité par défaut, consultez la section Groupes de sécurité par défaut et personnalisés dans le guide de EC2 l'utilisateur Amazon.
- Vérifiez que vous disposez des autorisations suivantes :
  - créez des piles, des flottes et des utilisateurs dans AppStream la version 2.0.
  - créez des piles à AWS CloudFormation l'aide d'un modèle.
  - créez des buckets et chargez des fichiers dans des buckets dans Amazon S3.

- télécharger les informations d'identification (`access_key_idetsecret_access_key`) depuis IAM.

Étape 1 : Obtenir les images AppStream 2.0

Au cours de cette étape, vous partagez les images AppStream 2.0 pour Enterprise Analyzer et Enterprise Developer avec votre AWS compte.

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le volet de navigation de gauche, sélectionnez Outils.
3. Dans Analyse, développement et création d'actifs, choisissez Partager des actifs avec mon AWS compte.

Étape 2 : Création de la pile à l'aide du AWS CloudFormation modèle

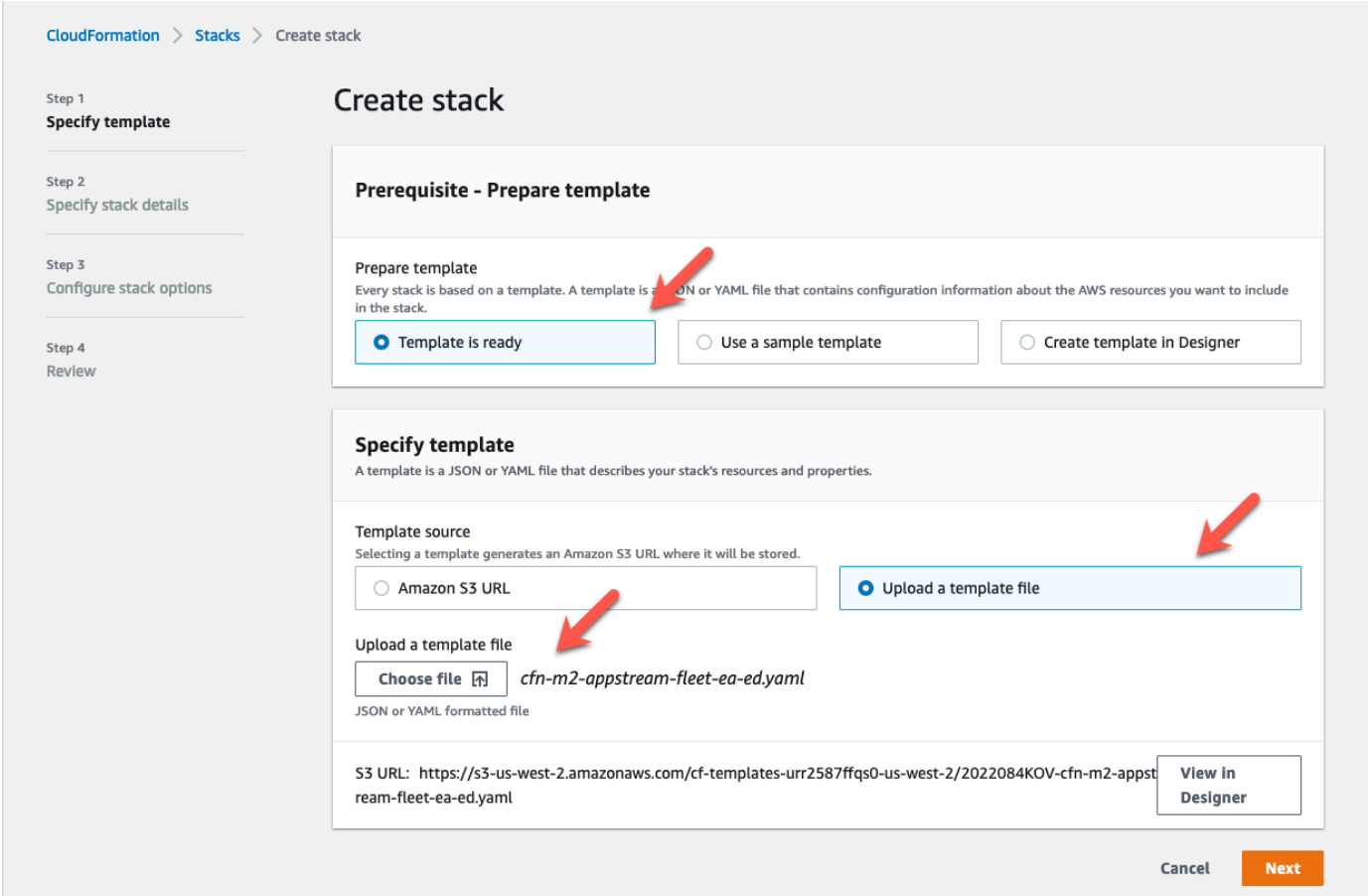
Au cours de cette étape, vous allez utiliser le AWS CloudFormation modèle téléchargé pour créer une pile et une flotte AppStream 2.0 pour exécuter Rocket Enterprise Analyzer. Vous pouvez répéter cette étape ultérieurement pour créer une autre pile et une autre flotte AppStream 2.0 pour exécuter Rocket Enterprise Developer, car chaque outil nécessite sa propre flotte et sa propre pile AppStream 2.0. Pour plus d'informations sur les AWS CloudFormation piles, consultez la section Utilisation des piles dans le Guide de l'AWS CloudFormation utilisateur.

> **Note**
>
> AWS La modernisation du mainframe ajoute des frais supplémentaires à la tarification AppStream 2.0 standard pour l'utilisation d'Enterprise Analyzer et d'Enterprise Developer. Pour plus d'informations, consultez la section Tarification de la modernisation des AWS mainframes.

1. Téléchargez le modèle cfn-m2- appstream-fleet-ea-ed .yml, si nécessaire.
2. Ouvrez la AWS CloudFormation console et choisissez Create Stack et avec de nouvelles ressources (standard).
3. Dans Prérequis - Préparer le modèle, sélectionnez Le modèle est prêt.
4. Dans Spécifier le modèle, choisissez Charger un fichier modèle.

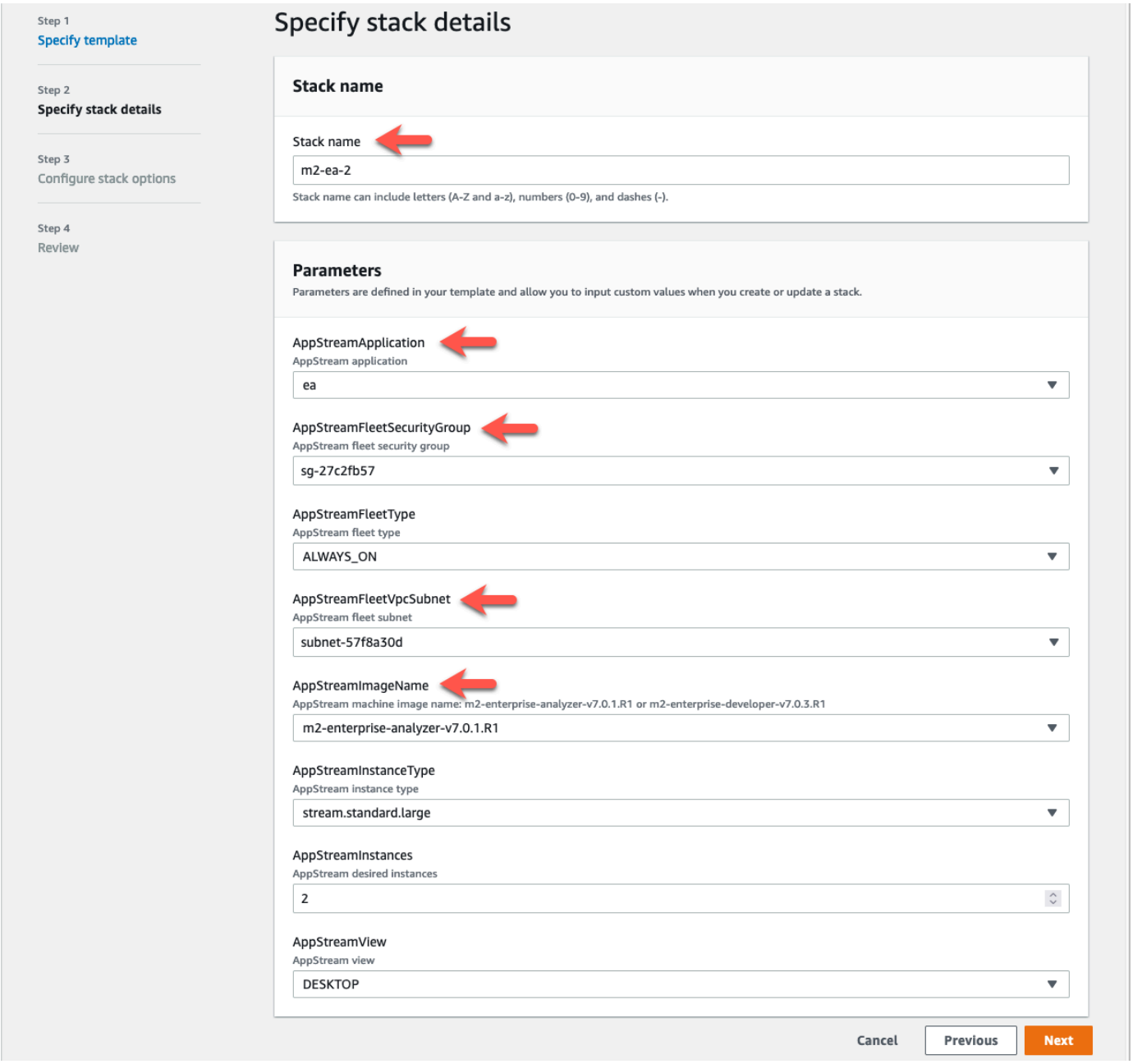
5. Dans Télécharger un fichier modèle, choisissez Choisir un fichier et chargez le modèle [cfn-m2-appstream-fleet-ea-ed .yml.]

6. Choisissez Next (Suivant).



7. Dans Spécifier les détails de la pile, entrez les informations suivantes :

   - Dans Nom de la pile, entrez le nom de votre choix. Par exemple, **`m2-ea`**.
   - Dans AppStreamApplication, choisissez ea.
   - Dans AppStreamFleetSecurityGroup, choisissez le groupe de sécurité par défaut de votre VPC par défaut.
   - Dans AppStreamFleetVpcSubnet, choisissez un sous-réseau au sein de votre VPC par défaut.
   - Dans AppStreamImageName, choisissez l'image commençant par`m2-enterprise-analyzer`. Cette image contient la version actuellement prise en charge de l'outil Rocket Enterprise Analyzer.
   - Acceptez les valeurs par défaut pour les autres champs, puis choisissez Next.



8. Acceptez toutes les valeurs par défaut, puis sélectionnez à nouveau Next.
9. Lors de la révision, assurez-vous que tous les paramètres correspondent à vos attentes.
10. Faites défiler la page vers le bas, sélectionnez Je reconnais qu'AWS CloudFormation peut créer des ressources IAM avec des noms personnalisés, puis sélectionnez Create Stack.

La création de la pile et de la flotte prend entre 20 et 30 minutes. Vous pouvez choisir Actualiser pour voir les AWS CloudFormation événements au fur et à mesure qu'ils se produisent.

Étape 3 : créer un utilisateur dans la AppStream version 2.0

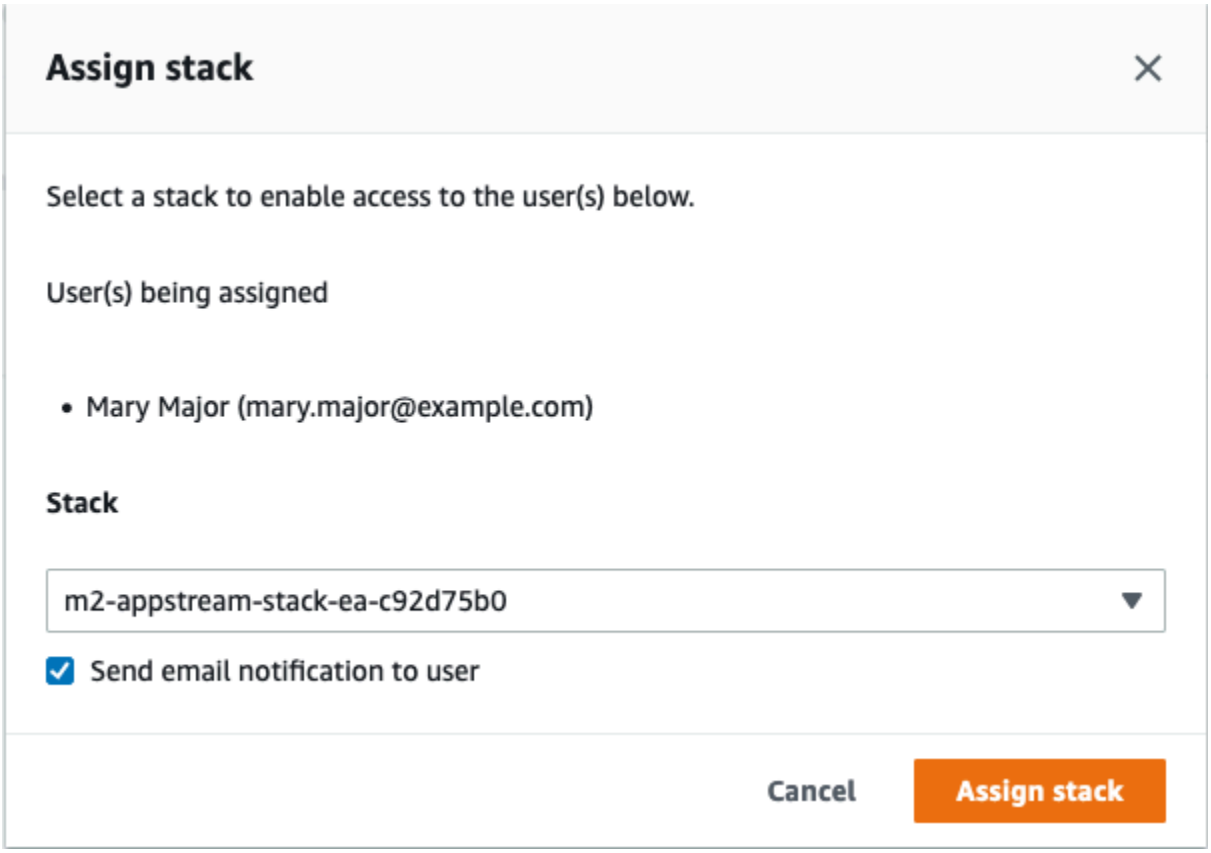
En attendant de terminer la création AWS CloudFormation de la pile, vous pouvez créer un ou plusieurs utilisateurs dans la AppStream version 2.0. Ces utilisateurs sont ceux qui utiliseront Enterprise Analyzer dans la AppStream version 2.0. Vous devrez spécifier une adresse e-mail pour chaque utilisateur et vous assurer que chaque utilisateur dispose des autorisations suffisantes pour créer des compartiments dans Amazon S3, charger des fichiers dans un compartiment et créer un lien vers un compartiment pour mapper son contenu.

1. Ouvrez la console AppStream 2.0.
2. Dans le volet de navigation de gauche, sélectionnez Groupe d'utilisateurs.
3. Choisissez Create user (Créer un utilisateur).
4. Indiquez une adresse e-mail à laquelle l'utilisateur pourra recevoir une invitation par e-mail à utiliser la AppStream version 2.0, un prénom et un nom de famille, puis choisissez Create user.
5. Répétez l'opération si nécessaire pour créer d'autres utilisateurs. L'adresse e-mail de chaque utilisateur doit être unique.

Pour plus d'informations sur la création d'utilisateurs AppStream 2.0, consultez AppStream la section Groupes d'utilisateurs 2.0 dans le guide d'administration Amazon AppStream 2.0.

Lorsque vous avez AWS CloudFormation terminé de créer la pile, vous pouvez affecter l'utilisateur que vous avez créé à la pile, comme suit :

1. Ouvrez la console AppStream 2.0.
2. Choisissez le nom d'utilisateur.
3. Choisissez Action, puis Attribuer une pile.
4. Dans Affecter une pile, choisissez la pile qui commence par `m2-appstream-stack-ea`.
5. Choisissez Attribuer une pile.



L'affectation d'un utilisateur à une pile entraîne l'envoi par AppStream 2.0 d'un e-mail à l'utilisateur à l'adresse que vous avez fournie. Cet e-mail contient un lien vers la page de connexion AppStream 2.0.

Étape 4 : Connectez-vous à la AppStream version 2.0

Au cours de cette étape, vous vous connectez à la AppStream version 2.0 en utilisant le lien contenu dans l'e-mail envoyé par la AppStream version 2.0 à l'utilisateur que vous avez créé[Étape 3 : créer un utilisateur dans la AppStream version 2.0](#).

1. Connectez-vous à AppStream 2.0 en utilisant le lien fourni dans l'e-mail envoyé par AppStream 2.0.
2. Modifiez votre mot de passe si vous y êtes invité. L'écran AppStream 2.0 que vous voyez est similaire au suivant :

3. Choisissez Desktop.
4. Dans la barre des tâches, choisissez Rechercher et entrez **`D:`** pour accéder au dossier d'accueil.

> **Note**
>
> Si vous ignorez cette étape, un `Device not ready` message d'erreur peut s'afficher lorsque vous tentez d'accéder au dossier d'accueil.

À tout moment, si vous rencontrez des difficultés pour vous connecter à la AppStream version 2.0, vous pouvez redémarrer votre flotte AppStream 2.0 et essayer de vous reconnecter en procédant comme suit.

1. Ouvrez la console AppStream 2.0.
2. Dans le menu de navigation de gauche, sélectionnez Fleets.
3. Choisissez la flotte que vous souhaitez utiliser.
4. Choisissez Action, puis Arrêter.
5. Attendez que la flotte s'arrête.
6. Choisissez Action, puis sélectionnez Démarrer.

Ce processus peut prendre environ 10 minutes.

Étape 5 : vérifier les compartiments dans Amazon S3 (facultatif)

L'une des tâches effectuées par le AWS CloudFormation modèle que vous avez utilisé pour créer la pile était de créer deux compartiments dans Amazon S3, nécessaires pour enregistrer et restaurer les données utilisateur et les paramètres des applications au cours des sessions de travail. Ces compartiments sont les suivants :

- Le nom commence par`appstream2-`. Ce bucket mappe les données vers votre dossier personnel dans la AppStream version 2.0 (`D:\PhotonUser\My Files\Home Folder`).

  > **Note**
  >
  > Le dossier d'accueil est unique pour une adresse e-mail donnée et est partagé entre toutes les flottes et tous les stocks d'un compte donné AWS . Le nom du dossier personnel est un SHA256 hachage de l'adresse e-mail de l'utilisateur et est stocké sur un chemin basé sur ce hachage.

- Le nom commence par`appstream-app-settings-`. Ce compartiment contient les informations de session utilisateur pour la AppStream version 2.0 et inclut des paramètres tels que les favoris du navigateur, les profils de connexion aux applications et à l'IDE, ainsi que les personnalisations de l'interface utilisateur. Pour plus d'informations, consultez Comment fonctionne la persistance des paramètres d'application dans le guide d'administration Amazon AppStream 2.0.

Pour vérifier que les compartiments ont été créés, procédez comme suit :

1. Ouvrez la console Amazon S3.
2. Dans le menu de navigation de gauche, choisissez Buckets.
3. Dans Rechercher des compartiments par nom, entrez **`appstream`** pour filtrer la liste.

Si vous voyez les compartiments, aucune autre action n'est nécessaire. Sachez simplement que les seaux existent. Si vous ne voyez pas les buckets, cela signifie que l'exécution du AWS CloudFormation modèle n'est pas terminée ou qu'une erreur s'est produite. Accédez à la AWS CloudFormation console et consultez les messages de création de pile.

Étapes suivantes

Maintenant que l'infrastructure AppStream 2.0 est configurée, vous pouvez configurer et commencer à utiliser Enterprise Analyzer. Pour de plus amples informations, veuillez consulter Tutoriel :

Configuration d'Enterprise Analyzer sur la version 2.0 AppStream . Vous pouvez également configurer Enterprise Developer. Pour de plus amples informations, veuillez consulter Tutoriel : Configurer Rocket Enterprise Developer sur AppStream 2.0.

Nettoyage des ressources

La procédure de nettoyage de la pile et des flottes créées est décrite dans Créer une flotte et une pile AppStream 2.0.

Lorsque les objets AppStream 2.0 ont été supprimés, l'administrateur du compte peut également, le cas échéant, nettoyer les compartiments Amazon S3 pour les paramètres de l'application et les dossiers personnels.

> **Note**
>
> Le dossier d'accueil d'un utilisateur donné étant unique dans toutes les flottes, vous devrez peut-être le conserver si d'autres piles AppStream 2.0 sont actives sur le même compte.

Enfin, la AppStream version 2.0 ne vous permet pas actuellement de supprimer des utilisateurs à l'aide de la console. Vous devez plutôt utiliser l'API de service avec la CLI. Pour plus d'informations, consultez la section Administration du groupe d'utilisateurs dans le guide d'administration Amazon AppStream 2.0.

Tutoriel : Utiliser des modèles avec Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer)

Ce didacticiel explique comment utiliser des modèles et des projets prédéfinis avec Rocket Enterprise Developer. Il couvre trois cas d'utilisation. Tous les cas d'utilisation utilisent le code d'exemple fourni dans l' BankDemo exemple. Pour télécharger l'extrait, choisissez `bankdemo.zip`.

> **Important**
>
> Si vous utilisez la version d'Enterprise Developer pour Windows, les fichiers binaires générés par le compilateur ne peuvent s'exécuter que sur le serveur Enterprise fourni avec Enterprise Developer. Vous ne pouvez pas les exécuter sous le moteur d'exécution AWS Mainframe Modernization, qui est basé sur Linux.

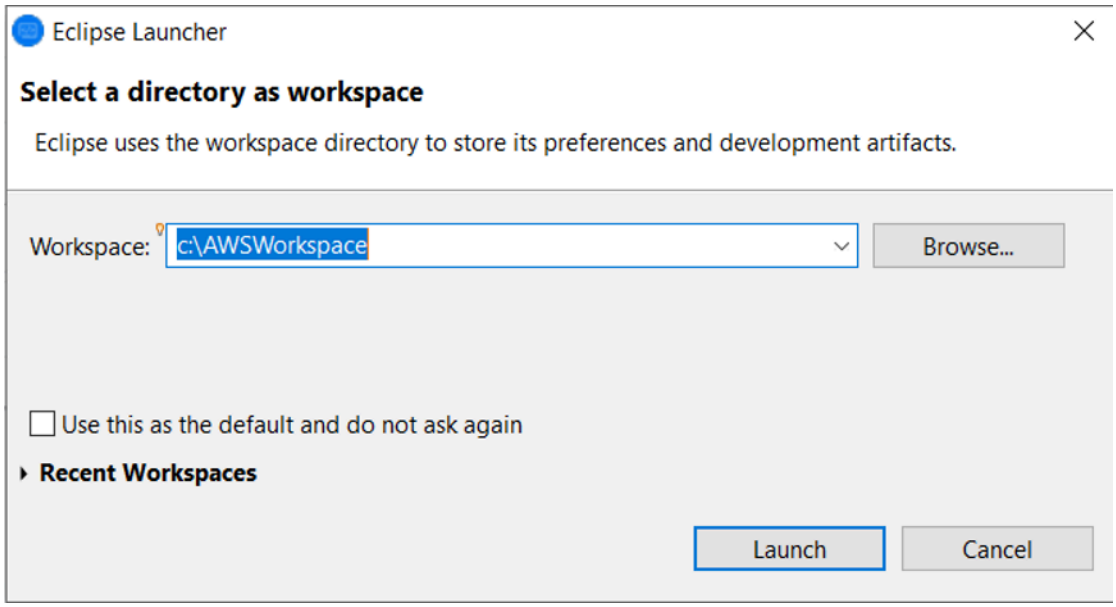
Rubriques

- Cas d'utilisation 1 - Utilisation du modèle de projet COBOL contenant les composants source
- Cas d'utilisation 2 - Utilisation du modèle de projet COBOL sans composants source
- Cas d'utilisation 3 - Utilisation du projet COBOL prédéfini lié aux dossiers sources
- Utilisation du modèle JSON de définition de région

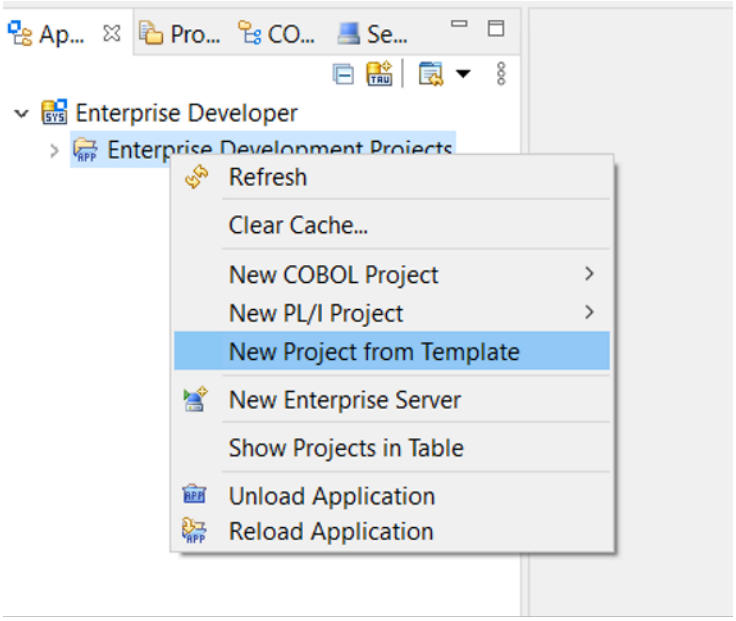
Cas d'utilisation 1 - Utilisation du modèle de projet COBOL contenant les composants source

Ce cas d'utilisation vous oblige à copier les composants source dans la structure de répertoire du modèle dans le cadre des étapes de préconfiguration de la démonstration. `bankdemo.zip`Cela a été modifié par rapport à la `AWSTemplates.zip` livraison originale pour éviter d'avoir deux copies de la source.

1. Démarrez Enterprise Developer et spécifiez l'espace de travail choisi.

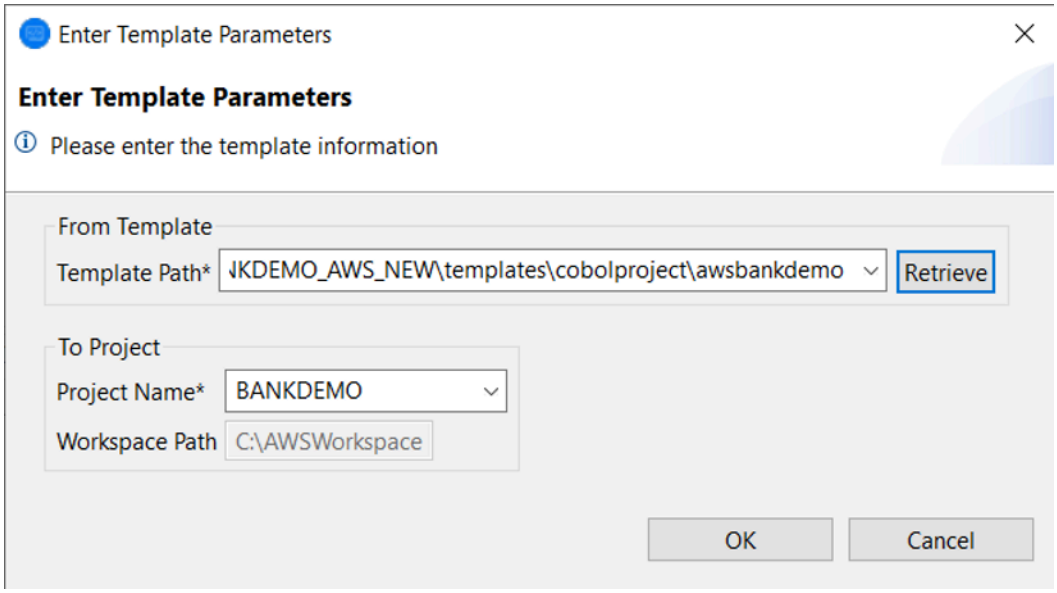


2. Dans la vue Explorateur d'applications, dans l'élément d'arborescence du projet de développement d'entreprise, choisissez Nouveau projet à partir d'un modèle dans le menu contextuel.

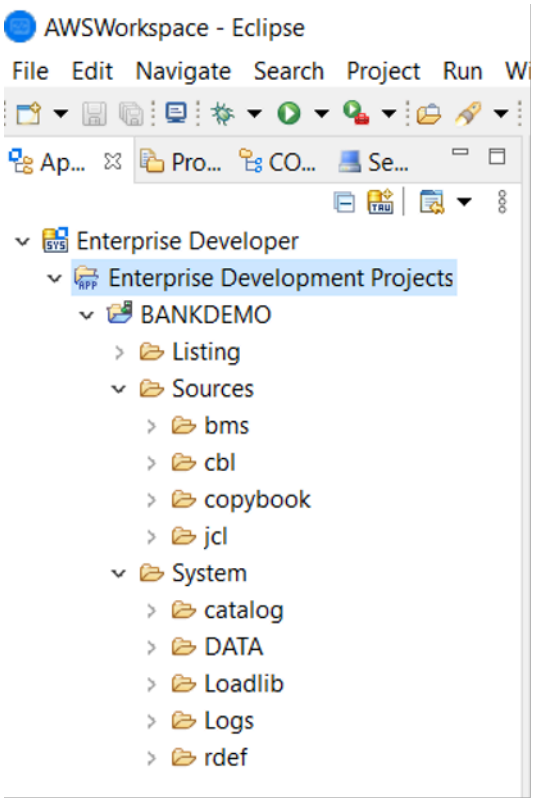


3. Entrez les paramètres du modèle comme indiqué.

> **Note**
>
> Le chemin du modèle fera référence à l'endroit où le fichier ZIP a été extrait.



4. En choisissant OK, vous créerez un projet Eclipse de développement local basé sur le modèle fourni, avec une structure complète de source et d'environnement d'exécution.



La `System` structure contient un fichier de définition de ressource complet avec les entrées requises pour BANKDEMO, le catalogue requis avec les entrées ajoutées et les fichiers de données ASCII correspondants.

Comme la structure du modèle source contient tous les éléments source, ces fichiers sont copiés dans le projet local et sont donc automatiquement intégrés dans Enterprise Developer.

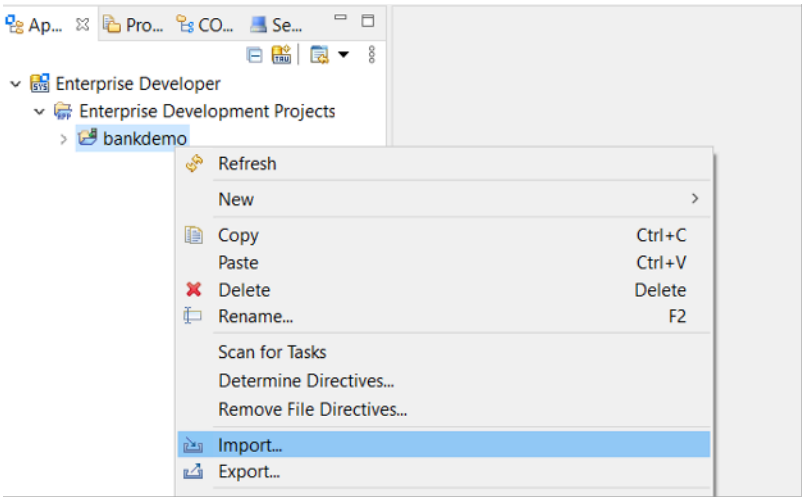
Cas d'utilisation 2 - Utilisation du modèle de projet COBOL sans composants source

Les étapes 1 à 3 sont identiques à[Cas d'utilisation 1 - Utilisation du modèle de projet COBOL contenant les composants source](#).

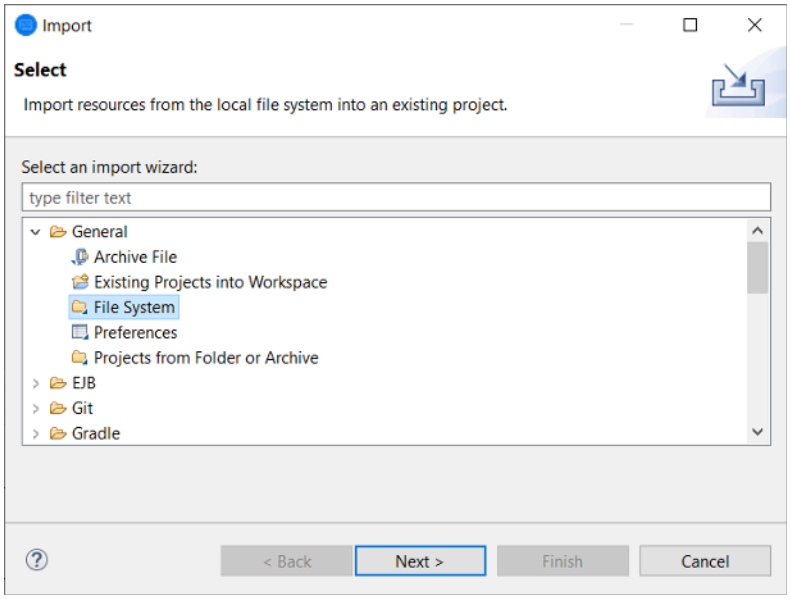
Dans ce cas d'utilisation, la `System` structure contient également un fichier de définition de ressource complet avec les entrées requises pour BankDemo, le catalogue requis avec les entrées ajoutées et les fichiers de données ASCII correspondants.

Cependant, la structure source du modèle ne contient aucun composant. Vous devez les importer dans le projet à partir du référentiel source que vous utilisez.

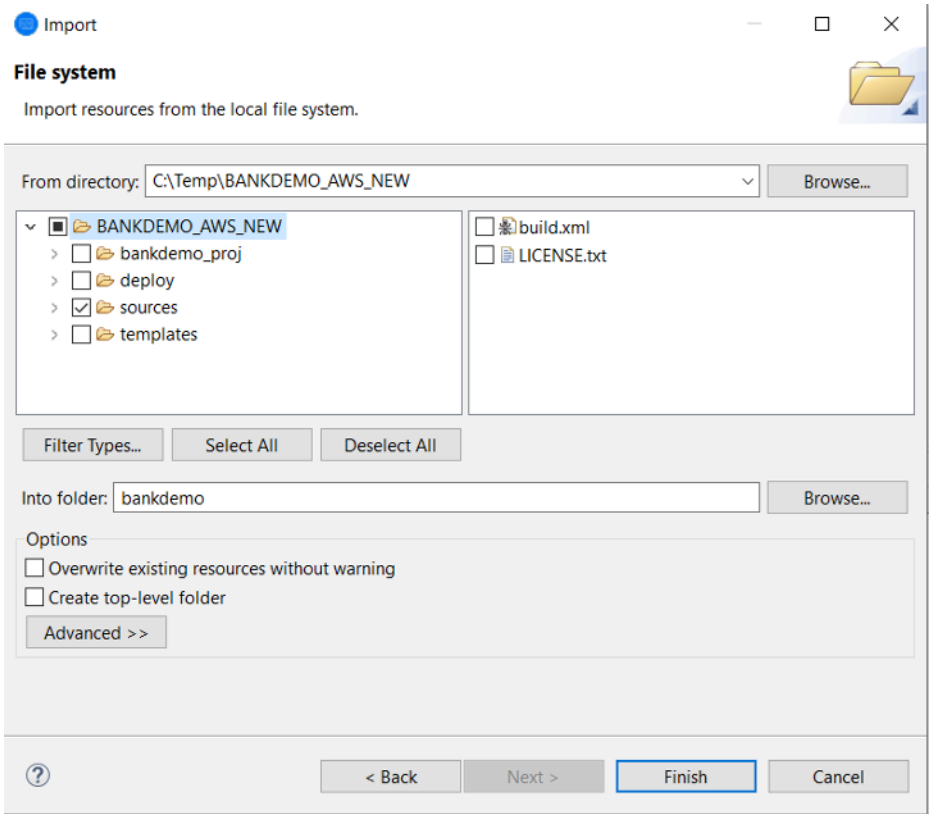
1. Choisissez le nom du projet. Dans le menu contextuel correspondant, choisissez Importer.



2. Dans la boîte de dialogue qui s'affiche, dans la section Général, sélectionnez Système de fichiers, puis Suivant.



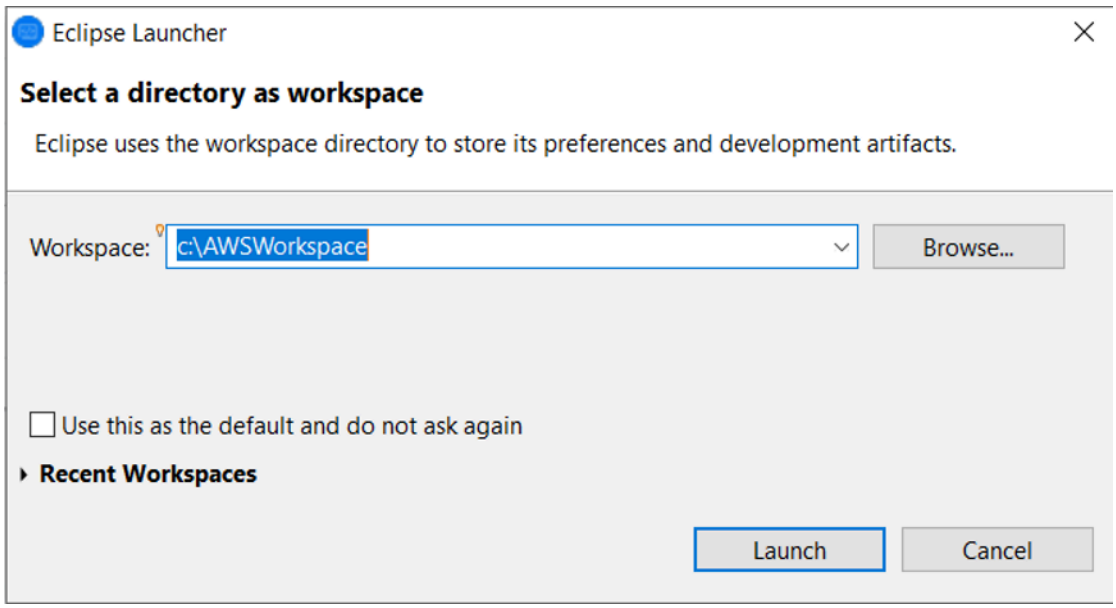
3. Renseignez le champ Depuis le répertoire en parcourant le système de fichiers pour pointer vers le dossier du référentiel. Choisissez tous les dossiers que vous souhaitez importer, tels que`sources`. Le `Into folder` champ sera prérempli. Choisissez Finish (Terminer).



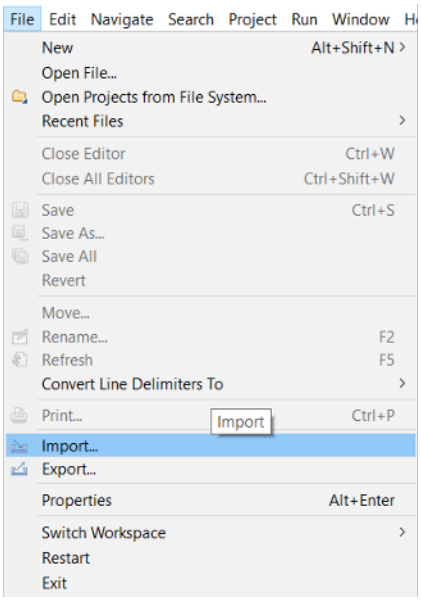
Une fois que la structure du modèle source contient tous les éléments source, ceux-ci sont automatiquement créés dans Enterprise Developer.

Cas d'utilisation 3 - Utilisation du projet COBOL prédéfini lié aux dossiers sources

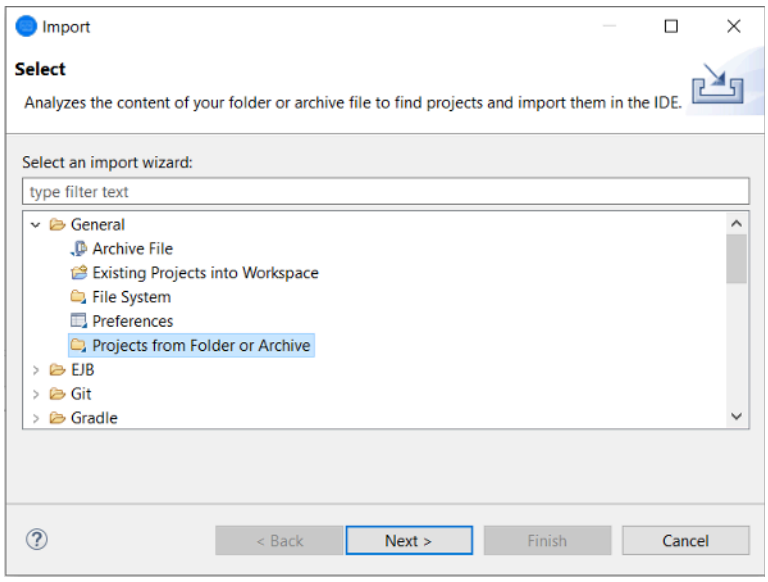
1. Démarrez Enterprise Developer et spécifiez l'espace de travail choisi.



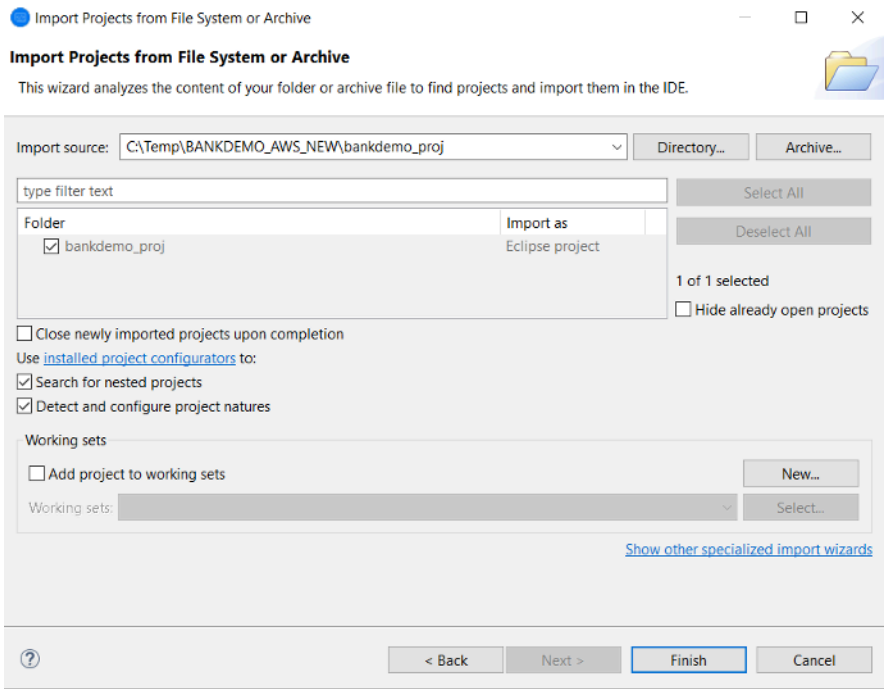
2. Dans le menu File (Fichier), choisissez Import (Importer).



3. Dans la boîte de dialogue qui s'affiche, sous Général, choisissez Projets depuis un dossier ou une archive, puis cliquez sur Suivant.



4. Renseignez la source d'importation, choisissez le répertoire et parcourez le système de fichiers pour sélectionner le dossier de projet prédéfini. Le projet qu'il contient contient des liens vers les dossiers sources du même référentiel.

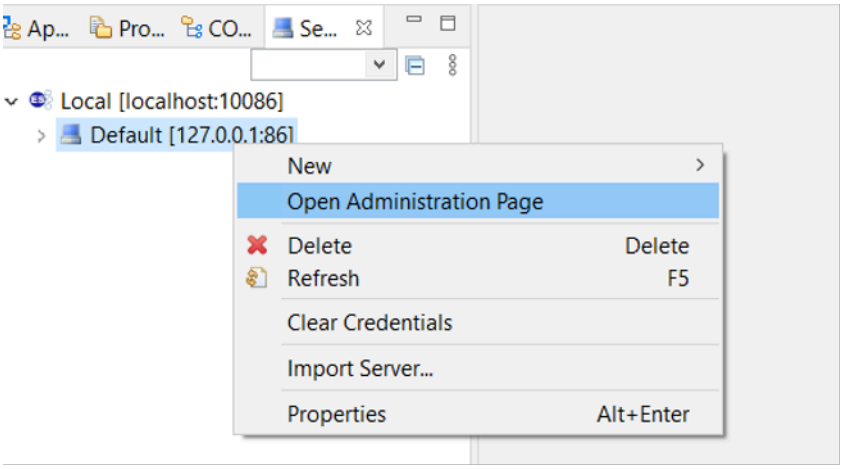


Choisissez Finish (Terminer).

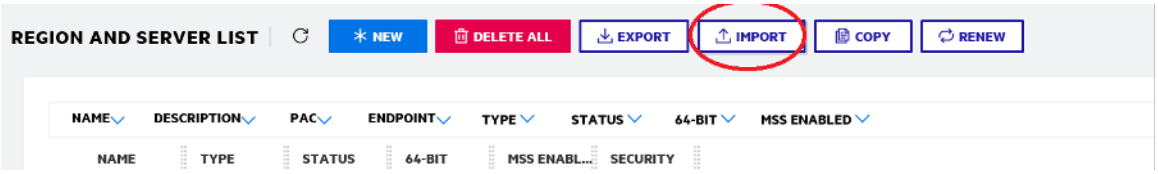
Le projet étant alimenté par les liens vers le dossier source, le code est automatiquement créé.

Utilisation du modèle JSON de définition de région

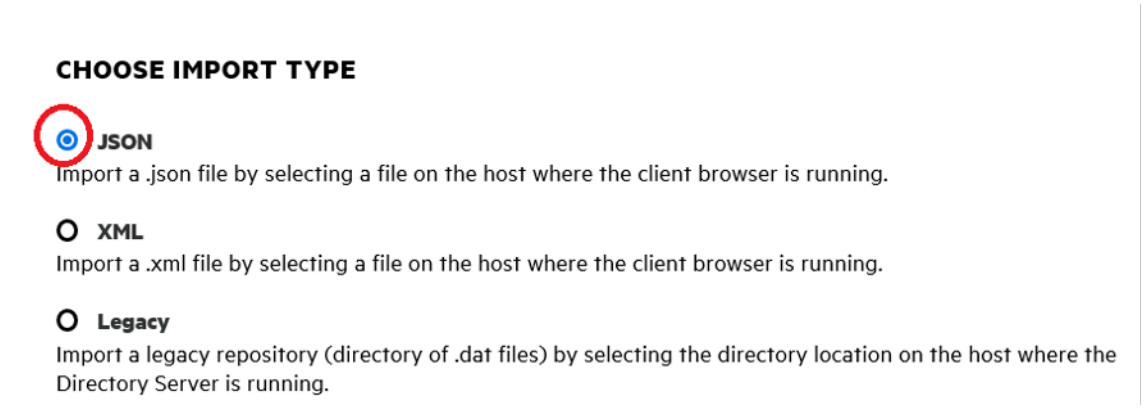
1. Passez à la vue Explorateur de serveurs. Dans le menu contextuel correspondant, choisissez Ouvrir la page d'administration, qui lance le navigateur par défaut.



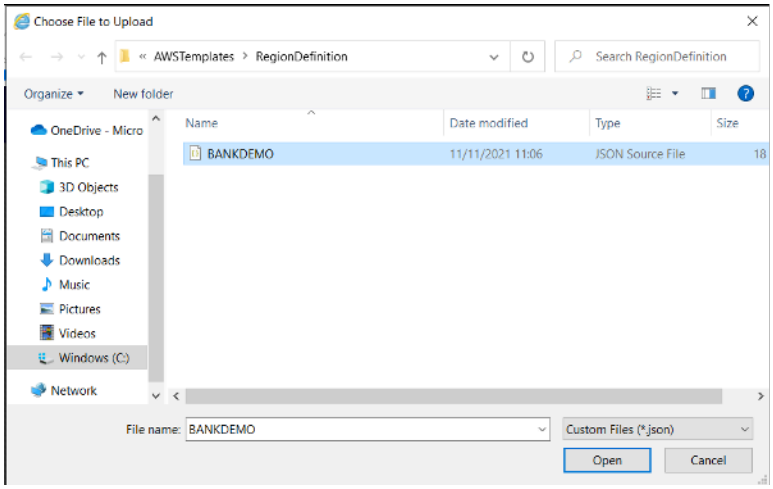
2. Dans l'écran Enterprise Server Common Web Administration (ESCWA) qui s'affiche, choisissez Importer.



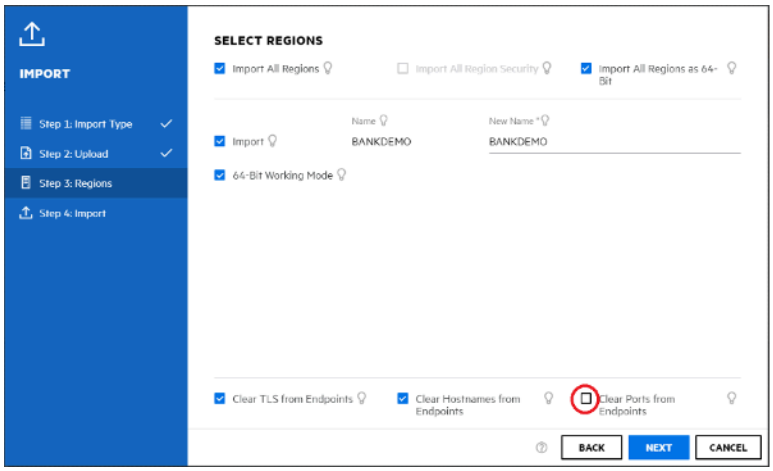
3. Choisissez le type d'importation JSON, puis cliquez sur Next.



4. Téléchargez le `BANKDEMO.JSON` fichier fourni.

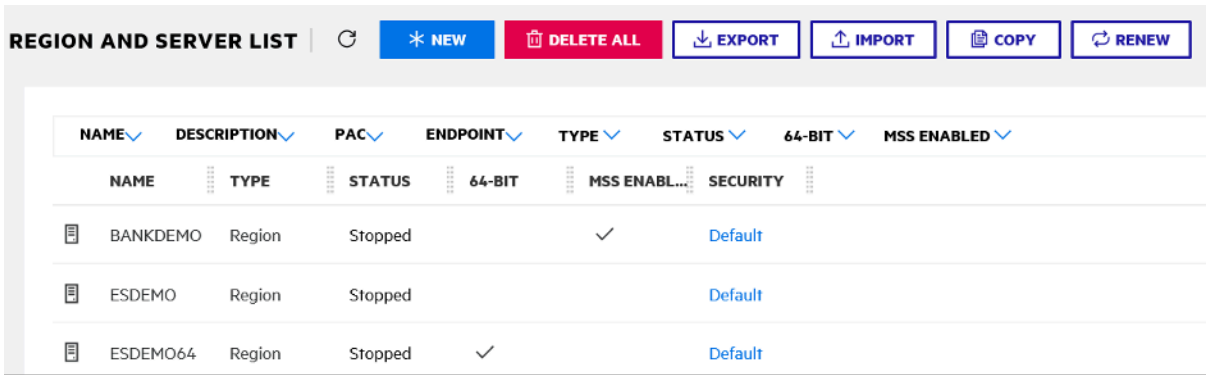


Une fois sélectionné, choisissez Next.

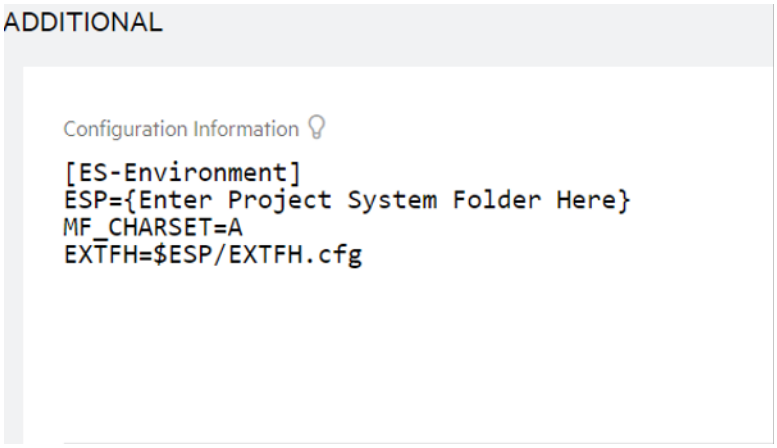


Dans le panneau Sélectionner les régions, assurez-vous que l'option Effacer les ports des points de terminaison n'est pas sélectionnée, puis continuez à sélectionner Suivant dans les panneaux jusqu'à ce que le panneau Perform Import apparaisse. Choisissez ensuite Importer dans le volet de navigation de gauche.

Enfin, cliquez sur Terminer. La région BANKDEMO sera ensuite ajoutée à la liste des serveurs.



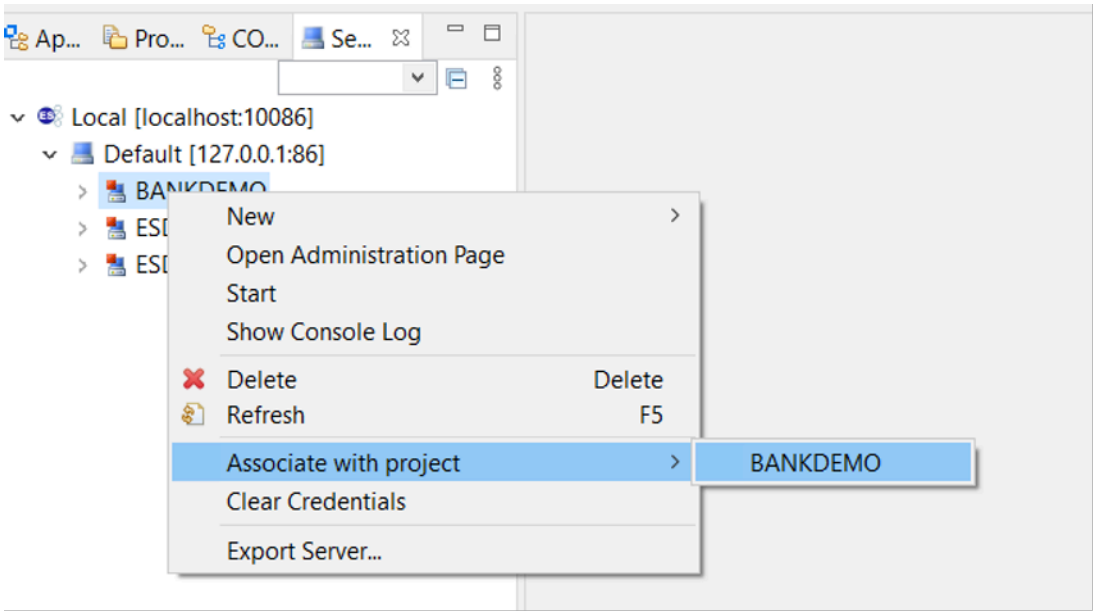
5. Accédez aux propriétés générales de la région BANKDEMO.
6. Accédez à la section Configuration.
7. La variable d'environnement ESP doit être définie sur le `System` dossier correspondant au projet Eclipse créé au cours des étapes précédentes. Cela devrait être le cas`workspacefolder/projectname/System`.



8. Cliquez sur Apply.

   La région est désormais entièrement configurée pour fonctionner conjointement avec le projet Eclipse COBOL.

9. Enfin, de retour dans Enterprise Developer, associez la région importée au projet.



L'environnement Enterprise Developer est désormais prêt à être utilisé, avec une version fonctionnelle complète de BankDemo. Vous pouvez modifier, compiler et déboguer le code par rapport à la région.

> **Important**
>
> Si vous utilisez la version d'Enterprise Developer pour Windows, les fichiers binaires générés par le compilateur ne peuvent s'exécuter que sur le serveur Enterprise fourni avec Enterprise Developer. Vous ne pouvez pas les exécuter sous le moteur d'exécution AWS Mainframe Modernization, qui est basé sur Linux.

Tutoriel : Configuration d'Enterprise Analyzer sur la version 2.0 AppStream

Ce didacticiel explique comment configurer Rocket Enterprise Analyzer (anciennement Micro Focus Enterprise Analyzer) pour analyser une ou plusieurs applications mainframe. L'outil Enterprise Analyzer fournit plusieurs rapports basés sur son analyse du code source de l'application et des définitions du système.

Cette configuration est conçue pour favoriser la collaboration au sein des équipes. L'installation utilise un compartiment Amazon S3 pour partager le code source avec des disques virtuels. Pour ce faire, vous devez utiliser Rclone) sur l'ordinateur Windows. Avec une instance Amazon RDS commune exécutant PostgreSQL, tous les membres de l'équipe peuvent accéder à tous les rapports demandés.

Les membres de l'équipe peuvent également monter le disque virtuel sauvegardé par Amazon S3 sur leurs machines personnelles et mettre à jour le compartiment source depuis leur poste de travail. Ils peuvent potentiellement utiliser des scripts ou toute autre forme d'automatisation sur leurs machines s'ils sont connectés à d'autres systèmes internes sur site.

La configuration est basée sur les images Windows AppStream 2.0 que AWS Mainframe Modernization partage avec le client. La configuration repose également sur la création de flottes et de piles AppStream 2.0, comme décrit dans. Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer

> **Important**
>
> Les étapes de ce didacticiel supposent que vous avez configuré la AppStream version 2.0 avec le AWS CloudFormation modèle téléchargeable cfn-m2- appstream-fleet-ea-ed .yml.

Pour de plus amples informations, veuillez consulter Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer.
Pour effectuer les étapes de ce didacticiel, vous devez avoir configuré votre parc et votre stack Enterprise Analyzer et ils doivent être en cours d'exécution.

Pour une description complète des fonctionnalités et des livrables d'Enterprise Analyzer, consultez la documentation d'Enterprise Analyzer sur le site Web de Rocket Software (anciennement Micro Focus).

Contenu de l'image

Outre l'application Enterprise Analyzer elle-même, l'image contient les outils et bibliothèques suivants.

Outils tiers

- Python
- Recloner
- pgAdmin
- git-scm
- pilote ODBC pour PostgreSQL

Bibliothèques de `C:\Users\Public`

- BankDemo code source et définition du projet pour Enterprise Developer :`m2-bankdemo-template.zip`.
- Package d'installation MFA pour le mainframe :. `mfa.zip` Pour plus d'informations, consultez la section Présentation de l'accès au mainframe dans la documentation Micro Focus Enterprise Developer.
- Fichiers de commande et de configuration pour Rclone (instructions pour leur utilisation dans les didacticiels) : `m2-rclone.cmd` et`m2-rclone.conf`.

Rubriques

- Prérequis

- Étape 1 : configuration
- Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows
- Étape 3 : créer une source ODBC pour l'instance Amazon RDS
- Sessions suivantes
- Résolution des problèmes de connexion aux espaces
- Nettoyage des ressources

Prérequis

- Téléchargez le code source et les définitions du système pour l'application client que vous souhaitez analyser dans un compartiment S3. Les définitions du système incluent CICS CSD, les définitions d' DB2 objets, etc. Vous pouvez créer une structure de dossiers dans le compartiment adaptée à la manière dont vous souhaitez organiser les artefacts de l'application. Par exemple, lorsque vous décompressez l' BankDemo échantillon, sa structure est la suivante :

```
demo
     |--> jcl
     |--> RDEF
     |--> transaction
     |--> xa
```

- Créez et démarrez une instance Amazon RDS exécutant PostgreSQL. Cette instance stockera les données et les résultats produits par Enterprise Analyzer. Vous pouvez partager cette instance avec tous les membres de l'équipe chargée de l'application. Créez également un schéma vide appelé `m2_ea` (ou tout autre nom approprié) dans la base de données. Définissez les informations d'identification pour les utilisateurs autorisés qui leur permettent de créer, d'insérer, de mettre à jour et de supprimer des éléments dans ce schéma. Vous pouvez obtenir le nom de la base de données, l'URL du point de terminaison du serveur et le port TCP depuis la console Amazon RDS ou auprès de l'administrateur du compte.
- Assurez-vous d'avoir configuré l'accès programmatique à votre Compte AWS. Pour plus d'informations, consultez la section Accès par programmation dans le Référence générale d'Amazon Web Services.

Étape 1 : configuration

1. Démarrez une session avec la AppStream version 2.0 avec l'URL que vous avez reçue dans le message électronique de bienvenue de la AppStream version 2.0.
2. Utilisez votre adresse e-mail comme nom d'utilisateur et définissez votre mot de passe permanent.
3. Sélectionnez la pile Enterprise Analyzer.
4. Sur la page du menu AppStream 2.0, choisissez Bureau pour accéder au bureau Windows que le parc diffuse.

Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows

> **Note**
>
> Si vous avez déjà utilisé Rclone lors de la version préliminaire de AWS Mainframe Modernization, vous devez effectuer la mise à jour `m2-rclone.cmd` vers la version la plus récente située dans. `C:\Users\Public`

1. Copiez les `m2-rclone.cmd` fichiers `m2-rclone.conf` et fournis dans votre dossier `C:\Users\Public` de base à `C:\Users\PhotonUser\My Files\Home Folder` l'aide de l'explorateur de fichiers.
2. Mettez à jour les paramètres de `m2-rclone.conf` configuration avec votre clé AWS d'accès et le secret correspondant, ainsi que votre Région AWS.

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. Dans `m2-rclone.cmd`, effectuez les modifications suivantes :

   - `amzn-s3-demo-bucket`Changez le nom de votre compartiment Amazon S3. Par exemple, `m2-s3-mybucket`.

- Passez `your-s3-folder-key` à la clé de votre compartiment Amazon S3. Par exemple, `myProject`.
- Accédez `your-local-folder-path` au chemin du répertoire dans lequel vous souhaitez que les fichiers d'application soient synchronisés à partir du compartiment Amazon S3 qui les contient. Par exemple, `D:\PhotonUser\My Files\Home Folder\m2-new`. Ce répertoire synchronisé doit être un sous-répertoire du dossier d'accueil pour que la AppStream version 2.0 puisse le sauvegarder et le restaurer correctement au début et à la fin de la session.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:amzn-s3-demo-bucket/your-s3-
folder-key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Ouvrez une invite de commande Windows, envoyez-la `C:\Users\PhotonUser\My Files\Home Folder` si nécessaire et lancez-la`m2-rclone.cmd`. Ce script de commande exécute une boucle continue, synchronisant votre compartiment et votre clé Amazon S3 avec le dossier local toutes les 10 secondes. Vous pouvez ajuster le délai d'attente selon vos besoins. Vous devriez voir le code source de l'application situé dans le compartiment Amazon S3 de l'Explorateur de fichiers Windows.

Pour ajouter de nouveaux fichiers à l'ensemble sur lequel vous travaillez ou pour mettre à jour les fichiers existants, chargez les fichiers dans le compartiment Amazon S3 et ils seront synchronisés avec votre répertoire lors de la prochaine itération définie dans. `m2-rclone.cmd` De même, si vous souhaitez supprimer certains fichiers, supprimez-les du compartiment Amazon S3. La prochaine opération de synchronisation les supprimera de votre répertoire local.

Étape 3 : créer une source ODBC pour l'instance Amazon RDS

1. Pour démarrer l'outil EA_Admin, accédez au menu de sélection d'applications dans le coin supérieur gauche de la fenêtre du navigateur et choisissez MF EA_Admin.
2. Dans le menu Administrer, choisissez Sources de données ODBC, puis choisissez Ajouter dans l'onglet DSN utilisateur.
3. Dans la boîte de dialogue Créer une nouvelle source de données, choisissez le pilote Unicode PostgreSQL, puis cliquez sur Terminer.

4. Dans la boîte de dialogue de configuration du pilote ODBC Unicode PostgreSQL (PSQLodBC), définissez et notez le nom de la source de données que vous souhaitez. Complétez les paramètres suivants avec les valeurs de l'instance RDS que vous avez créée précédemment :

   Description

   Description facultative pour vous aider à identifier rapidement cette connexion à la base de données.

   Base de données

   La base de données Amazon RDS que vous avez créée précédemment.

   Serveur

   Le point de terminaison Amazon RDS.

   Port

   Le port Amazon RDS.

   Nom utilisateur

   Tel que défini dans l'instance Amazon RDS.

   Mot de passe

   Tel que défini dans l'instance Amazon RDS.

5. Choisissez Test pour vérifier que la connexion à Amazon RDS est réussie, puis sélectionnez Enregistrer pour enregistrer votre nouveau DSN utilisateur.
6. Attendez de voir le message confirmant la création de l'espace de travail approprié, puis cliquez sur OK pour terminer avec les sources de données ODBC et fermer l'outil EA_Admin.
7. Accédez à nouveau au menu de sélection d'applications, puis choisissez Enterprise Analyzer pour démarrer l'outil. Choisissez Créer un nouveau.
8. Dans la fenêtre de configuration de l'espace de travail, entrez le nom de votre espace de travail et définissez son emplacement. L'espace de travail peut être le disque basé sur Amazon S3 si vous travaillez sous cette configuration, ou votre dossier personnel si vous préférez.
9. Choisissez Choose Other Database pour vous connecter à votre instance Amazon RDS.
10. Choisissez l'icône Postgre parmi les options, puis cliquez sur OK.

11. Pour les paramètres Windows, sous Options — Définir les paramètres de connexion, entrez le nom de la source de données que vous avez créée. Entrez également le nom de la base de données, le nom du schéma, le nom d'utilisateur et le mot de passe. Choisissez OK.
12. Attendez qu'Enterprise Analyzer crée toutes les tables, tous les index, etc. dont il a besoin pour stocker les résultats. Ce processus peut prendre quelques minutes. Enterprise Analyzer confirme le moment où la base de données et l'espace de travail sont prêts à être utilisés.
13. Accédez à nouveau au menu de sélection d'applications et choisissez Enterprise Analyzer pour démarrer l'outil.
14. La fenêtre de démarrage d'Enterprise Analyzer apparaît dans le nouvel emplacement d'espace de travail sélectionné. Choisissez OK.
15. Accédez à votre référentiel dans le volet de gauche, sélectionnez le nom du référentiel, puis choisissez Ajouter des fichiers/dossiers à votre espace de travail. Sélectionnez le dossier dans lequel le code de votre application est stocké pour l'ajouter à l'espace de travail. Vous pouvez utiliser l' BankDemoexemple de code précédent si vous le souhaitez. Lorsque Enterprise Analyzer vous invite à vérifier ces fichiers, choisissez Verify pour démarrer le rapport de vérification initial d'Enterprise Analyzer. Le traitement peut prendre quelques minutes, en fonction de la taille de votre demande.
16. Développez votre espace de travail pour afficher les fichiers et dossiers que vous y avez ajoutés. Les types d'objets et les rapports de complexité cyclomatique sont également visibles dans le quadrant supérieur du volet Chart Viewer.

Vous pouvez désormais utiliser Enterprise Analyzer pour toutes les tâches nécessaires.

Sessions suivantes

1. Démarrez une session avec la AppStream version 2.0 avec l'URL que vous avez reçue dans le message électronique de bienvenue de la AppStream version 2.0.
2. Connectez-vous à l'aide de votre e-mail et de votre mot de passe permanent.
3. Sélectionnez la pile Enterprise Analyzer.
4. Lancez `Rclone` pour vous connecter au disque sauvegardé par Amazon S3 si vous utilisez cette option pour partager les fichiers de l'espace de travail.
5. Lancez Enterprise Analyzer pour effectuer vos tâches.

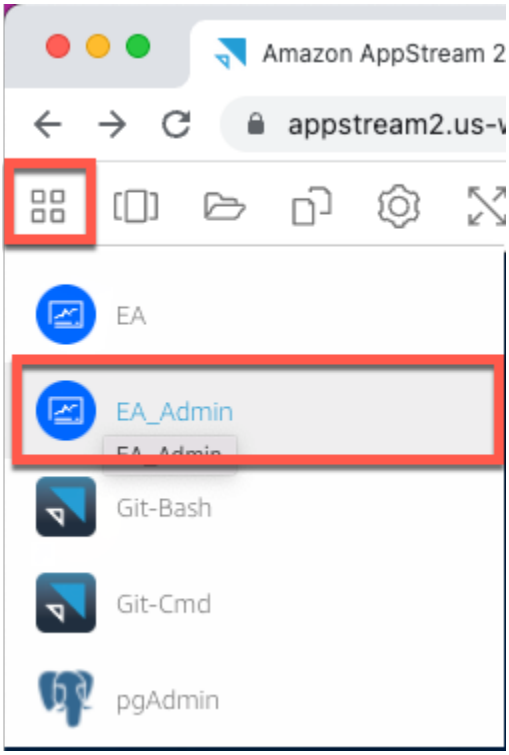
Résolution des problèmes de connexion aux espaces

Lorsque vous essayez de vous reconnecter à votre espace de travail Enterprise Analyzer, le message d'erreur suivant peut s'afficher :

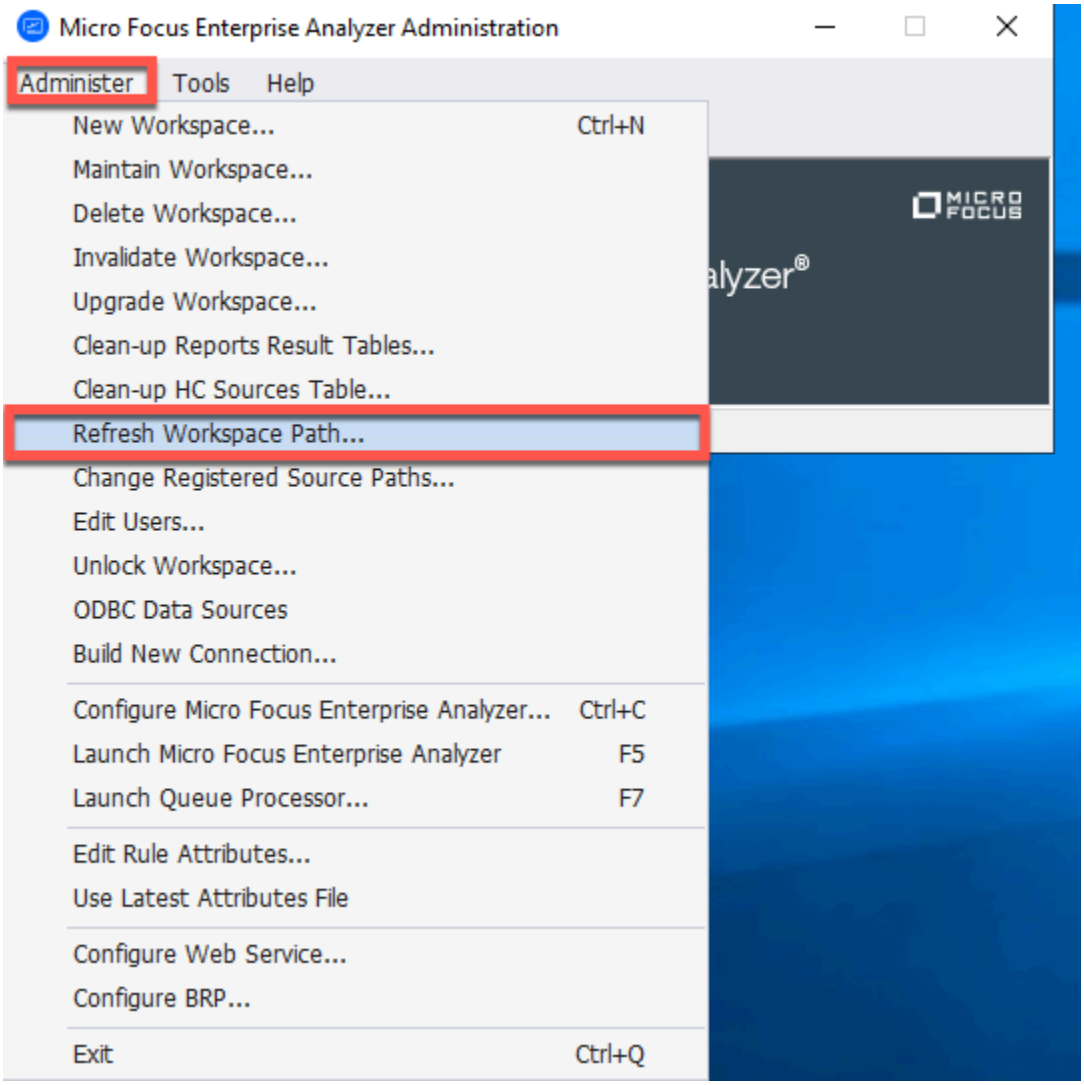
```
Cannot access the workspace directory D:\PhotonUser\My Files\Home Folder\EA_BankDemo.
 The workspace has been created on a non-shared disk of the EC2AMAZ-E6LC33H computer.
 Would you like to correct the workspace directory location?
```

Pour résoudre ce problème, cliquez sur OK pour effacer le message, puis effectuez les étapes suivantes.

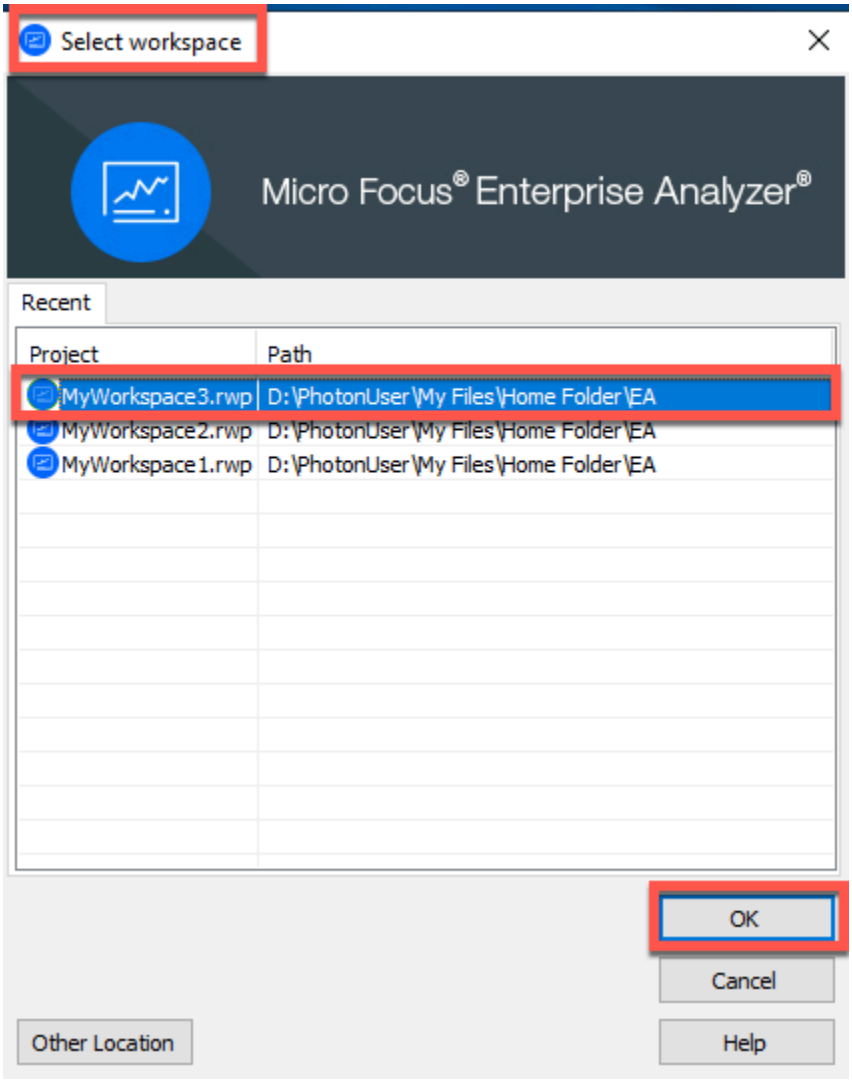
1. Dans la AppStream version 2.0, cliquez sur l'icône Lancer l'application dans la barre d'outils, puis choisissez EA_Admin pour démarrer l'outil d'administration d'Enterprise Analyzer.



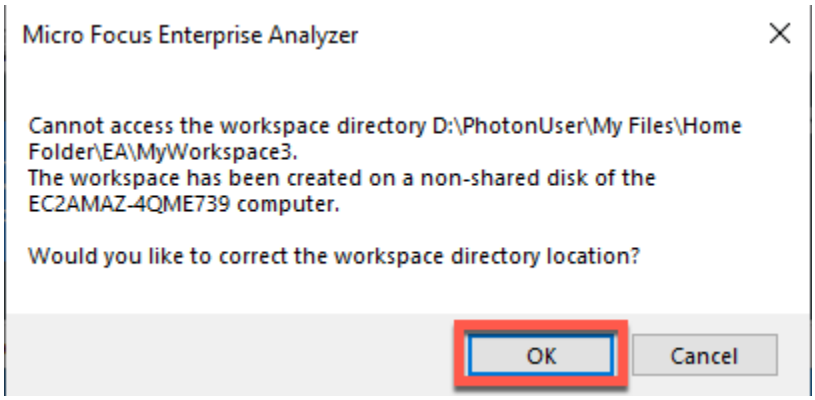
2. Dans le menu Administrer, choisissez Actualiser le chemin de l'espace de travail... .



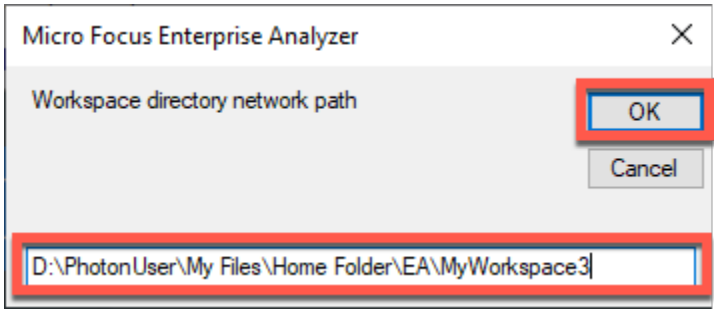
3. Sous Sélectionner un espace de travail, choisissez l'espace de travail de votre choix, puis cliquez sur OK.



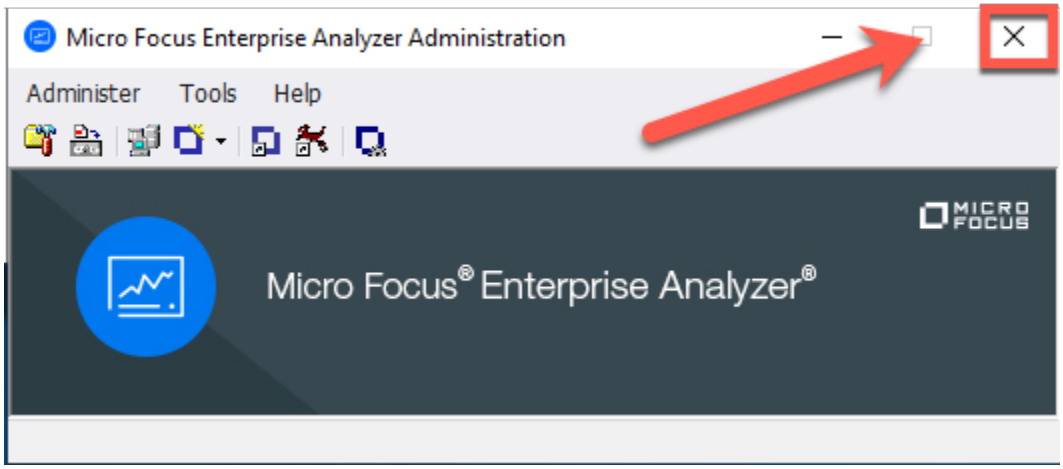
4. Cliquez sur OK pour confirmer le message d'erreur.



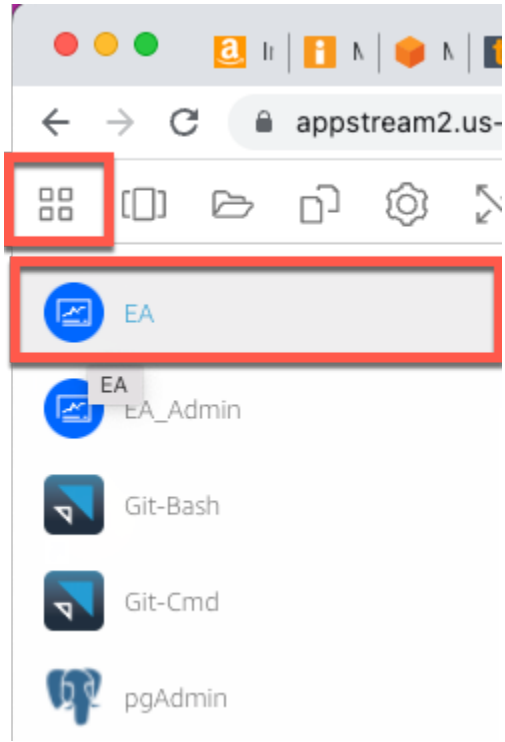
5. Sous Chemin réseau du répertoire de l'espace de travail, entrez le chemin d'accès correct à votre espace de travail, par exemple,`D:\PhotonUser\My Files\Home Folder\EA\MyWorkspace3`.



6. Fermez l'outil d'administration de Micro Focus Enterprise Analyzer.



7. Dans la AppStream version 2.0, cliquez sur l'icône Lancer l'application dans la barre d'outils, puis choisissez EA pour démarrer Micro Focus Enterprise Analyzer.



8. Répétez les étapes 3 à 5.

Micro Focus Enterprise Analyzer devrait maintenant s'ouvrir avec l'espace de travail existant.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin de ne pas avoir à payer de frais supplémentaires. Procédez comme suit :

- Utilisez l'outil EA_Admin pour supprimer l'espace de travail.
- Supprimez les compartiments S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez la section Suppression d'un compartiment dans le guide de l'utilisateur Amazon S3.
- Supprimez la base de données que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Suppression d'une instance de base de données.

Tutoriel : Configurer Rocket Enterprise Developer sur AppStream 2.0

Ce didacticiel explique comment configurer Rocket Enterprise Developer (anciennement Micro Focus Enterprise Developer) pour une ou plusieurs applications mainframe afin de les maintenir, de les compiler et de les tester à l'aide des fonctionnalités Enterprise Developer. La configuration est basée sur les images Windows AppStream 2.0 que AWS Mainframe Modernization partage avec le client et sur la création de flottes et de piles AppStream 2.0, comme décrit dans. Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer

> **Important**
>
> Les étapes de ce didacticiel supposent que vous configurez la AppStream version 2.0 à l'aide du AWS CloudFormation modèle téléchargeable cfn-m2- appstream-fleet-ea-ed .yaml. Pour de plus amples informations, veuillez consulter Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Rocket Enterprise Analyzer et Rocket Enterprise Developer.
>
> Vous devez effectuer les étapes de cette configuration lorsque le parc et le stack d'Enterprise Developer sont opérationnels.

Pour une description complète des fonctionnalités et des livrables d'Enterprise Developer v7, consultez sa documentation up-to-date en ligne (v7.0) sur le site de Rocket Software (anciennement Micro Focus).

Contenu de l'image

Outre Enterprise Developer lui-même, l'image contient l'image contient Rumba (un émulateur TN3270). Il contient également les outils et bibliothèques suivants.

Outils tiers

- Python
- Recloner
- pgAdmin
- git-scm
- pilote ODBC pour PostgreSQL

Bibliothèques de `C:\Users\Public`

- BankDemo code source et définition du projet pour Enterprise Developer :`m2-bankdemo-template.zip`.
- Package d'installation MFA pour le mainframe :. `mfa.zip` Pour plus d'informations, consultez la section Présentation de l'accès au mainframe dans la documentation Micro Focus Enterprise Developer.
- Fichiers de commande et de configuration pour Rclone (instructions pour leur utilisation dans les didacticiels) : `m2-rclone.cmd` et`m2-rclone.conf`.

Si vous devez accéder à du code source qui n'est pas encore chargé dans CodeCommit les référentiels, mais qui est disponible dans un compartiment Amazon S3, par exemple pour effectuer le chargement initial du code source dans git, suivez la procédure de création d'un disque Windows virtuel comme décrit dansTutoriel : Configuration d'Enterprise Analyzer sur la version 2.0 AppStream .

Rubriques

- Prérequis
- Étape 1 : Configuration par les utilisateurs individuels d'Enterprise Developer
- Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows (facultatif)

- Étape 3 : Cloner le dépôt
- Sessions suivantes
- Nettoyage des ressources

Prérequis

- Un ou plusieurs CodeCommit référentiels chargés avec le code source de l'application à maintenir. La configuration du référentiel doit répondre aux exigences du CI/CD pipeline ci-dessus pour créer des synergies en combinant les deux outils.
- Chaque utilisateur doit disposer d'informations d'identification pour le CodeCommit ou les référentiels définis par l'administrateur du compte conformément aux informations de la section Authentification et contrôle d'accès pour AWS CodeCommit. La structure de ces informations d'identification est examinée dans Authentification et contrôle d'accès pour AWS CodeCommit et la référence complète pour les autorisations IAM CodeCommit se trouve dans la référence des CodeCommit autorisations : l'administrateur peut définir des politiques IAM distinctes pour des rôles distincts en ayant des informations d'identification spécifiques au rôle de chaque référentiel et en limitant les autorisations de l'utilisateur à l'ensemble de tâches spécifiques qu'il doit accomplir sur un référentiel donné. Ainsi, pour chaque responsable du CodeCommit référentiel, l'administrateur du compte créera un utilisateur principal et accordera à cet utilisateur l'autorisation d'accéder au ou aux référentiels requis en sélectionnant la ou les politiques IAM appropriées pour l'accès. CodeCommit

Étape 1 : Configuration par les utilisateurs individuels d'Enterprise Developer

1. Obtenez vos informations d'identification IAM :
    1. Connectez-vous à la AWS console à l'adresse https://console.aws.amazon.com/iam/.
    2. Suivez la procédure décrite à l'étape 3 de Configuration pour les utilisateurs HTTPS à l'aide des informations d'identification Git dans le guide de AWS CodeCommit l'utilisateur.
    3. Copiez les informations de CodeCommit connexion spécifiques qu'IAM a générées pour vous, soit en affichant, copiant puis collant ces informations dans un fichier sécurisé sur votre ordinateur local, soit en choisissant Télécharger les informations d'identification pour télécharger ces informations sous forme de fichier .CSV. Vous avez besoin de ces informations pour vous connecter à CodeCommit.

2. Démarrez une session avec AppStream 2.0 en fonction de l'URL reçue dans l'e-mail de bienvenue. Utilisez votre adresse e-mail comme nom d'utilisateur et créez votre mot de passe.
3. Sélectionnez votre stack de développeurs d'entreprise.
4. Sur la page de menu, choisissez Desktop pour accéder au bureau Windows diffusé par le parc.

Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows (facultatif)

Si Rclone est nécessaire (voir ci-dessus), créez le dossier virtuel basé sur Amazon S3 sous Windows : (facultatif si tous les artefacts de l'application proviennent CodeCommit exclusivement de l'accès).

> **Note**
>
> Si vous avez déjà utilisé Rclone lors de la version préliminaire de AWS Mainframe Modernization, vous devez effectuer la mise à jour `m2-rclone.cmd` vers la version la plus récente située dans. `C:\Users\Public`

1. Copiez les `m2-rclone.cmd` fichiers `m2-rclone.conf` et fournis dans votre dossier `C:\Users\Public` de base à `C:\Users\PhotonUser\My Files\Home Folder` l'aide de l'explorateur de fichiers.
2. Mettez à jour les paramètres de `m2-rclone.conf` configuration avec votre clé AWS d'accès et le secret correspondant, ainsi que votre Région AWS.

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. Dans `m2-rclone.cmd`, effectuez les modifications suivantes :
   - `amzn-s3-demo-bucket`Changez le nom de votre compartiment Amazon S3. Par exemple, `m2-s3-mybucket`.

- Passez `your-s3-folder-key` à la clé de votre compartiment Amazon S3. Par exemple, `myProject`.
- Accédez `your-local-folder-path` au chemin du répertoire dans lequel vous souhaitez que les fichiers d'application soient synchronisés à partir du compartiment Amazon S3 qui les contient. Par exemple, `D:\PhotonUser\My Files\Home Folder\m2-new`. Ce répertoire synchronisé doit être un sous-répertoire du dossier d'accueil pour que la AppStream version 2.0 puisse le sauvegarder et le restaurer correctement au début et à la fin de la session.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:amzn-s3-demo-bucket/your-s3-
folder-key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Ouvrez une invite de commande Windows, envoyez-la `C:\Users\PhotonUser\My Files\Home Folder` si nécessaire et lancez-la`m2-rclone.cmd`. Ce script de commande exécute une boucle continue, synchronisant votre compartiment et votre clé Amazon S3 avec le dossier local toutes les 10 secondes. Vous pouvez ajuster le délai d'attente selon vos besoins. Vous devriez voir le code source de l'application situé dans le compartiment Amazon S3 dans l'Explorateur de fichiers Windows.

Pour ajouter de nouveaux fichiers à l'ensemble sur lequel vous travaillez ou pour mettre à jour les fichiers existants, chargez les fichiers dans le compartiment Amazon S3 et ils seront synchronisés avec votre répertoire lors de la prochaine itération définie dans. `m2-rclone.cmd` De même, si vous souhaitez supprimer certains fichiers, supprimez-les du compartiment Amazon S3. La prochaine opération de synchronisation les supprimera de votre répertoire local.

Étape 3 : Cloner le dépôt

1. Accédez au menu de sélection d'applications dans le coin supérieur gauche de la fenêtre du navigateur et sélectionnez Enterprise Developer.
2. Terminez la création de l'espace de travail requise par Enterprise Developer dans votre dossier principal en choisissant `C:\Users\PhotonUser\My Files\Home Folder` (alias`D:\PhotonUser\My Files\Home Folder`) comme emplacement pour l'espace de travail.

3. Dans Enterprise Developer, clonez votre CodeCommit dépôt en accédant à l'explorateur de projets, en cliquant avec le bouton droit de la souris et en choisissant Importer, Importer..., Git, Projects from Git Clone URI. Entrez ensuite vos informations de CodeCommit connexion spécifiques et complétez la boîte de dialogue Eclipse pour importer le code.

Le dépôt CodeCommit git est désormais cloné dans votre espace de travail local.

Votre espace de travail Enterprise Developer est maintenant prêt à démarrer les travaux de maintenance de votre application. Vous pouvez notamment utiliser l'instance locale d'Enterprise Server (ES) intégrée à Enterprise Developer pour déboguer et exécuter votre application de manière interactive afin de valider vos modifications localement.

> **Note**
>
> L'environnement de développement d'entreprise local, y compris l'instance locale de serveur d'entreprise, s'exécute sous Windows tandis que AWS Mainframe Modernization s'exécute sous Linux. Nous vous recommandons d'exécuter des tests complémentaires dans l'environnement Linux fourni par AWS Mainframe Modernization après avoir validé CodeCommit et reconstruit la nouvelle application pour cette cible et avant de déployer la nouvelle application en production.

Sessions suivantes

Lorsque vous sélectionnez un dossier géré par la AppStream version 2.0, tel que le dossier de base pour le clonage de votre CodeCommit dépôt, il sera enregistré et restauré de manière transparente d'une session à l'autre. Procédez comme suit la prochaine fois que vous aurez besoin de travailler avec l'application :

1. Démarrez une session avec AppStream 2.0 en fonction de l'URL reçue dans l'e-mail de bienvenue.
2. Connectez-vous avec votre e-mail et votre mot de passe permanent.
3. Sélectionnez la pile Enterprise Developer.
4. Lancez `Rclone` pour vous connecter (voir ci-dessus) au disque sauvegardé par Amazon S3 lorsque cette option est utilisée pour partager les fichiers de l'espace de travail.
5. Lancez Enterprise Developer pour faire votre travail.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin qu'elles ne continuent pas à vous être facturées. Procédez comme suit :

- Supprimez le CodeCommit référentiel que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un CodeCommit référentiel dans le Guide de AWS CodeCommit l'utilisateur.
- Supprimez la base de données que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Suppression d'une instance de base de données.

Utilitaires de traitement par lots disponibles dans AWS Mainframe Modernization

Les applications mainframe utilisent souvent des programmes utilitaires par lots pour exécuter des fonctions spécifiques telles que le tri des données, le transfert de fichiers par FTP, le chargement de données dans des bases de données DB2, le déchargement de données depuis des bases de données, etc.

Lorsque vous migrez vos applications vers la modernisation du AWS mainframe, vous avez besoin d'utilitaires de remplacement fonctionnellement équivalents, capables d'effectuer les mêmes tâches que ceux que vous utilisiez sur le mainframe. Certains de ces utilitaires sont peut-être déjà disponibles dans le cadre des moteurs d'exécution AWS Mainframe Modernization, mais nous proposons les utilitaires de remplacement suivants :

- M2SFTP - permet un transfert de fichiers sécurisé à l'aide du protocole SFTP.
- M2WAIT - attend pendant un certain temps avant de passer à l'étape suivante d'un traitement par lots.
- TXT2PDF - convertit les fichiers texte au format PDF.
- M2DFUTIL : fournit des fonctions de sauvegarde, de restauration, de suppression et de copie sur des ensembles de données similaires au support fourni par l'utilitaire ADRDSSU du mainframe.
- M2RUNCMD - vous permet d'exécuter des commandes, des scripts et des appels système de Rocket Software (anciennement Micro Focus) directement depuis JCL.

Nous avons développé ces utilitaires par lots en fonction des commentaires des clients et les avons conçus pour fournir les mêmes fonctionnalités que les utilitaires du mainframe. L'objectif est de faciliter au maximum votre transition du mainframe à la modernisation du AWS mainframe.

Rubriques

- Emplacement binaire
- Utilitaire batch M2SFTP
- Utilitaire batch M2WAIT
- TXT2Utilitaire PDF par lots
- Utilitaire de traitement par lots M2DFUTIL
- Utilitaire batch M2RUNCMD

Emplacement binaire

Ces utilitaires sont préinstallés sur les produits Rocket Enterprise Developer (ED) et Rocket Software (ES). Vous pouvez les trouver à l'emplacement suivant pour toutes les variantes de ED et ES :

- Linux : `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

Utilitaire batch M2SFTP

M2SFTP est un utilitaire JCL conçu pour effectuer des transferts de fichiers sécurisés entre systèmes à l'aide du protocole SFTP (Secure File Transfer Protocol). Le programme utilise le client Putty SFTP pour effectuer `psftp` les transferts de fichiers réels. Le programme fonctionne de la même manière qu'un utilitaire FTP sur ordinateur central et utilise l'authentification par utilisateur et mot de passe.

> **Note**
>
> L'authentification par clé publique n'est pas prise en charge.

Pour convertir le FTP de votre mainframe JCLs en SFTP, remplacez `PGM=FTP` par. `PGM=M2SFTP`

Rubriques

- Plateformes prises en charge
- Installation des dépendances
- Configuration du M2SFTP pour la gestion de la modernisation du AWS mainframe
- Configurer le M2SFTP pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream
- Exemple JCLs
- Référence de commande client Putty SFTP (PSFTP)
- Étapes suivantes

Plateformes prises en charge

Vous pouvez utiliser M2SFTP sur l'une des plateformes suivantes :

- AWS Modernisation du mainframe Rocket Software (anciennement Micro Focus) Géré
- Rocket Software Runtime (sur Amazon EC2)
- Toutes les variantes des produits Rocket Software Enterprise Developer (ED) et Rocket Software Enterprise Server (ES).

Installation des dépendances

Pour installer le client SFTP Putty sous Windows

- Téléchargez le client SFTP PuTTY et installez-le.

Pour installer le client SFTP Putty sous Linux :

- Exécutez la commande suivante pour installer le client SFTP Putty :

```
sudo yum -y install putty
```

Configuration du M2SFTP pour la gestion de la modernisation du AWS mainframe

Si vos applications migrées s'exécutent sur AWS Mainframe Modernization Managed, vous devez configurer M2SFTP comme suit.

- Définissez les variables d'environnement Rocket Enterprise Server appropriées pour le MFFTP. Voici quelques exemples :

  - `MFFTP_TEMP_DIR`
  - `MFFTP_SENDEOL`
  - `MFFTP_TIME`
  - `MFFTP_ABEND`

  Vous pouvez définir aussi peu ou autant de variables que vous le souhaitez. Vous pouvez les définir dans votre JCL à l'aide de l'`ENVAR DD`instruction. Pour plus d'informations sur ces variables, consultez la section Variables de contrôle MFFTP dans la documentation de Micro Focus.

Pour tester votre configuration, consultezExemple JCLs.

Configurer le M2SFTP pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez M2SFTP comme suit.

1. Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez des deux-points (`:`) pour séparer les chemins sous Linux des points-virgules (`;`) sous Windows.

   - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`
2. Définissez les variables d'environnement Rocket Enterprise Server appropriées pour le MFFTP. Voici quelques exemples :

   - `MFFTP_TEMP_DIR`

- MFFTP_SENDEOL
- MFFTP_TIME
- MFFTP_ABEND

Vous pouvez définir aussi peu ou autant de variables que vous le souhaitez. Vous pouvez les définir dans votre JCL à l'aide de l'ENVAR DDinstruction. Pour plus d'informations sur ces variables, consultez la section Variables de contrôle MFFTP dans la documentation de Micro Focus.

Pour tester votre configuration, consultezExemple JCLs.

Exemple JCLs

Pour tester l'installation, vous pouvez utiliser l'un des exemples de fichiers JCL suivants.

SFTP1M2.jcl

Cette JCL montre comment appeler M2SFTP pour envoyer un fichier à un serveur SFTP distant. Notez les variables d'environnement définies dans l'ENVVAR DDinstruction.

```
//M2SFTP1 JOB 'M2SFTP1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Sample SFTP JCL step to send a file to SFTP server*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP,
//            PARM='127.0.0.1 (EXIT=99 TIMEOUT 300'
//*
//SYSFTPD  DD  *
RECFM FB
LRECL 80
SBSENDEOL CRLF
MBSENDEOL CRLF
TRAILINGBLANKS FALSE
/*
//NETRC    DD  *
```

```
machine 127.0.0.1 login sftpuser password sftppass
/*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
type a
locsite notrailingblanks
cd files
put 'AWS.M2.TXT2PDF1.PDF' AWS.M2.TXT2PDF1.pdf
put 'AWS.M2.CARDDEMO.CARDDATA.PS' AWS.M2.CARDDEMO.CARDDATA.PS1.txt
quit
/*
//ENVVAR   DD *
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

SFTP2M2.jcl

Cette JCL montre comment appeler M2SFTP pour recevoir un fichier d'un serveur SFTP distant. Notez les variables d'environnement définies dans l'ENVVAR DDinstruction.

```
//M2SFTP2 JOB 'M2SFTP2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Sample SFTP JCL step to receive a file from SFTP server*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP
//*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
open 127.0.0.1
sftpuser
sftppass
```

```
cd files
locsite recfm=fb lrecl=150
get AWS.M2.CARDDEMO.CARDDATA.PS.txt +
'AWS.M2.CARDDEMO.CARDDATA.PS2' (replace
quit
/*
//ENVVAR   DD *
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

> **Note**
>
> Nous recommandons vivement de stocker les informations d'identification FTP dans un fichier NETRC et de restreindre l'accès aux seuls utilisateurs autorisés.

Référence de commande client Putty SFTP (PSFTP)

Le client PSFTP ne prend pas en charge toutes les commandes FTP. La liste suivante répertorie toutes les commandes prises en charge par PSFTP.

Command	Description
!	Exécuter une commande locale
au revoir	Terminez votre session SFTP
cd	Modifier votre répertoire de travail à distance
chmod	Modifier les autorisations et les modes des fichiers
close	Terminez votre session SFTP mais ne quittez pas PSFTP
del	Supprimer des fichiers sur le serveur distant
dir	Lister les fichiers distants

Command	Description
exit	Terminez votre session SFTP
get	Téléchargez un fichier depuis le serveur vers votre machine locale
aide	Donnez de l'aide
lcd	Modifier le répertoire de travail local
lpwd	Imprimer le répertoire de travail local
ls	Lister les fichiers distants
mget	Téléchargez plusieurs fichiers à la fois
mkdir	Création de répertoires sur le serveur distant
mput	Téléchargez plusieurs fichiers à la fois
mv	Déplacer ou renommer des fichiers sur le serveur distant
ouvrir	Se connecter à un hôte
put	Téléchargez un fichier depuis votre machine locale vers le serveur
mobylette	Imprimez votre répertoire de télétravail
quit	Terminez votre session SFTP
reget	Poursuivre le téléchargement des fichiers
ren	Déplacer ou renommer des fichiers sur le serveur distant
réputation	Poursuivre le téléchargement des fichiers
rm	Supprimer des fichiers sur le serveur distant

Command	Description
rmdir	Supprimer des répertoires sur le serveur distant

Étapes suivantes

Pour charger et télécharger des fichiers dans Amazon Simple Storage Service à l'aide du protocole SFTP, vous pouvez utiliser le protocole M2SFTP conjointement avec le protocole AWS Transfer Family, comme décrit dans les articles de blog suivants.

- Utilisation de répertoires logiques AWS SFTP pour créer un service de distribution de données simple
- Activer l'authentification par mot de passe pour AWS Transfer for SFTP l'utilisation AWS Secrets Manager

Utilitaire batch M2WAIT

M2WAIT est un utilitaire pour mainframe qui vous permet d'introduire une période d'attente dans vos scripts JCL en spécifiant une durée en secondes, minutes ou heures. Vous pouvez appeler M2WAIT directement depuis JCL en indiquant le temps d'attente comme paramètre d'entrée. En interne, le programme M2WAIT appelle le module fourni par Rocket Software (anciennement Micro Focus) `C $SLEEP` pour attendre un certain temps.

> **Note**
>
> Vous pouvez utiliser des alias Micro Focus pour remplacer le contenu de vos scripts JCL. Pour plus d'informations, consultez la section JES Alias dans la documentation de Micro Focus.

Rubriques

- Plateformes prises en charge
- Configuration de M2WAIT pour la gestion de la AWS modernisation du mainframe
- Configurer M2WAIT pour le runtime AWS Mainframe Modernization sur Amazon EC2 (y compris AppStream 2.0)
- Exemple de JCL

Plateformes prises en charge

Vous pouvez utiliser M2WAIT sur l'une des plateformes suivantes :

- AWS Modernisation du mainframe Rocket Software (anciennement Micro Focus) Géré
- Rocket Software Runtime (sur Amazon EC2)
- Toutes les variantes des produits Rocket Software Enterprise Developer (ED) et Rocket Software Enterprise Server (ES).

Configuration de M2WAIT pour la gestion de la AWS modernisation du mainframe

Si vos applications migrées s'exécutent sur AWS Mainframe Modernization Managed, vous devez configurer M2WAIT comme suit.

- Utilisez le programme M2WAIT dans votre JCL en passant le paramètre d'entrée comme indiqué dans. Exemple de JCL

Configurer M2WAIT pour le runtime AWS Mainframe Modernization sur Amazon EC2 (y compris AppStream 2.0)

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez M2WAIT comme suit.

1. Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez des deux-points (`:`) pour séparer les chemins sous Linux des points-virgules (`;`) sous Windows.
   - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`
2. Utilisez le programme M2WAIT dans votre JCL en transmettant le paramètre d'entrée comme indiqué dans. Exemple de JCL

Exemple de JCL

Pour tester l'installation, vous pouvez utiliser le `M2WAIT1.jcl` programme.

Cet exemple de JCL montre comment appeler M2WAIT et lui transmettre plusieurs durées différentes.

```
//M2WAIT1 JOB 'M2WAIT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Wait for 12 Seconds*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2WAIT,PARM='S012'
//SYSOUT DD SYSOUT=*
//*
//*-------------------------------------------------------------------**
//* Wait for 0 Seconds (defaulted to 10 Seconds)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=M2WAIT,PARM='S000'
//SYSOUT DD SYSOUT=*
//*
//*-------------------------------------------------------------------**
//* Wait for 1 Minute*
//*-------------------------------------------------------------------**
//*
//STEP03 EXEC PGM=M2WAIT,PARM='M001'
//SYSOUT DD SYSOUT=*
//*
//
```

TXT2Utilitaire PDF par lots

TXT2PDF est un utilitaire mainframe couramment utilisé pour convertir un fichier texte en fichier PDF. Cet utilitaire utilise le même code source pour le TXT2 format PDF (z/OS logiciel gratuit). Nous l'avons modifié pour qu'il fonctionne dans l'environnement d'exécution du logiciel AWS Mainframe Modernization Rocket (anciennement Micro Focus).

Rubriques

- Plateformes prises en charge
- Configuration du TXT2 PDF pour la gestion de la modernisation AWS du mainframe

- Configuration du TXT2 PDF pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris la AppStream version 2.0)
- Exemple de JCL
- Modifications
- Références

Plateformes prises en charge

Vous pouvez utiliser le TXT2 format PDF sur l'une des plateformes suivantes :

- AWS Modernisation des ordinateurs centraux : Rocket Software Managed
- Rocket Software Runtime (sur Amazon EC2)
- Toutes les variantes des produits Rocket Enterprise Developer (ED) et Rocket Enterprise Server (ES).

Configuration du TXT2 PDF pour la gestion de la modernisation AWS du mainframe

Si vos applications migrées s'exécutent sur AWS Mainframe Modernization Managed, configurez le TXT2 PDF comme suit.

- Créez une bibliothèque REXX EXEC appelée. `AWS.M2.REXX.EXEC` Téléchargez ces modules REXX et copiez-les dans la bibliothèque.
  - `TXT2PDF.rex`- TXT2 PDF z/OS logiciel gratuit (modifié)
  - `TXT2PDFD.rex`- TXT2 PDF z/OS logiciel gratuit (non modifié)
  - `TXT2PDFX.rex`- TXT2 PDF z/OS logiciel gratuit (modifié)
  - `M2GETOS.rex`- Pour vérifier le type de système d'exploitation (Windows ou Linux)

Pour tester votre configuration, consultezExemple de JCL.

Configuration du TXT2 PDF pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris la AppStream version 2.0)

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez le TXT2 PDF comme suit.

1. Définissez la variable d'environnement Rocket Software `MFREXX_CHARSET` sur la valeur appropriée, telle que « `A` » pour les données ASCII.

   > **Important**
   >
   > La saisie d'une valeur incorrecte peut entraîner des problèmes de conversion des données (de l'EBCDIC au format ASCII), rendant ainsi le PDF illisible ou inutilisable. Nous vous recommandons de `MFREXX_CHARSET` le régler en conséquence`MF_CHARSET`.

2. Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez des deux-points (`:`) pour séparer les chemins sous Linux des points-virgules (`;`) sous Windows.

   - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

3. Créez une bibliothèque REXX EXEC appelée. `AWS.M2.REXX.EXEC` ` Téléchargez ces modules REXX et copiez-les dans la bibliothèque.

   - `TXT2PDF.rex`- TXT2 PDF z/OS logiciel gratuit (modifié)
   - `TXT2PDFD.rex`- TXT2 PDF z/OS logiciel gratuit (non modifié)
   - `TXT2PDFX.rex`- TXT2 PDF z/OS logiciel gratuit (modifié)
   - `M2GETOS.rex`- Pour vérifier le type de système d'exploitation (Windows ou Linux)

Pour tester votre configuration, consultezExemple de JCL.

Exemple de JCL

Pour tester l'installation, vous pouvez utiliser l'un des exemples de fichiers JCL suivants.

TXT2PDF1.jcl

Cet exemple de fichier JCL utilise un nom DD pour la conversion TXT2 PDF.

```
//TXT2PDF1 JOB 'TXT2PDF1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
```

```
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* PRE DELETE*
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//OUTDD    DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=VB,BLKSIZE=0)
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
```

```
OUT DD:OUTDD +
CC YES
/*
//*
//*-------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF1.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

TXT2PDF2.jcl

Cet exemple de JCL utilise un nom DSN pour la conversion TXT2 PDF.

```
//TXT2PDF2 JOB 'TXT2PDF2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* PRE DELETE*
//*-------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF2.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*-------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*-------------------------------------------------------------------**
```

```
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT 'AWS.M2.TXT2PDF2.PDF.VB' +
CC YES
/*
//*
//*--------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*--------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF2.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

Modifications

Pour que le programme TXT2 PDF s'exécute dans l'environnement d'exécution du logiciel AWS Mainframe Modernization Rocket, nous avons apporté les modifications suivantes :

- Modifications apportées au code source pour garantir la compatibilité avec le runtime Rocket Software REXX
- Modifications visant à garantir que le programme peut s'exécuter à la fois sur les systèmes d'exploitation Windows et Linux
- Modifications pour prendre en charge les environnements d'exécution EBCDIC et ASCII

Références

TXT2Références PDF et code source :

- Convertisseur de texte en PDF
- z/OS Outils TCP/IP et de messagerie gratuits
- TXT2Guide de référence de l'utilisateur au format PDF

Utilitaire de traitement par lots M2DFUTIL

M2DFUTIL est un utilitaire JCL qui fournit des fonctions de sauvegarde, de restauration, de suppression et de copie sur des ensembles de données, de la même manière que le support fourni par l'utilitaire ADRDSSU du mainframe. Ce programme conserve de nombreux paramètres SYSIN d'ADRDSSU, ce qui simplifie le processus de migration vers ce nouvel utilitaire.

Rubriques

- Plateformes prises en charge
- Exigences relatives à la plateforme
- Assistance future prévue
- Localisation des actifs
- Configurer le runtime M2DFUTIL ou AWS Mainframe Modernization sur Amazon EC2 (y compris 2.0) AppStream
- Syntaxe générale

- Exemple JCLs

Plateformes prises en charge

Vous pouvez utiliser M2DFUTIL sur l'une des plateformes suivantes :

- Rocket Software (anciennement Micro Focus) ES sous Windows (64 bits et 32 bits)
- Rocket Software ES sous Linux (64 bits)

Exigences relatives à la plateforme

M2DFUTIL dépend de l'appel d'un script pour effectuer un test d'expression régulière. Sous Windows, vous devez installer Windows Services pour Linux (WSL) pour que ce script s'exécute.

Assistance future prévue

Les fonctionnalités qui ne sont pas actuellement disponibles dans l'utilitaire ADRDSSU du mainframe, mais qui seront étendues à l'avenir sont les suivantes :

- M2 géré
- VSAM
- Support COPY pour le changement de nom de fichier
- Support de renommage pour RESTORE
- Plusieurs options INCLUDE et EXCLUDE
- Clause BY pour la sous-sélection par DSORG, CREDT, EXPDT
- Clause MWAIT permettant de réessayer les échecs de la file d'attente
- Support de stockage S3 pour DUMP/RESTORE

Localisation des actifs

Le module de chargement de cet utilitaire est `M2DFUTIL.so` appelé sous `M2DFUTIL.dll` Linux et Windows. Ce module de chargement se trouve aux emplacements suivants :

- Linux : `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`

- Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

Le script utilisé pour tester les expressions régulières est appelé`compare.sh`. Ce script se trouve aux emplacements suivants :

- Linux : `/opt/aws/m2/microfocus/utilities/scripts`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\scripts`

Configurer le runtime M2DFUTIL ou AWS Mainframe Modernization sur Amazon EC2 (y compris 2.0) AppStream

Configurez la région de votre serveur d'entreprise avec les éléments suivants :

- Ajoutez les variables suivantes dans [ES-Environment]
  - `M2DFUTILS_BASE_LOC`- L'emplacement par défaut pour la sortie DUMP
  - `M2DFUTILS_SCRIPTPATH`- L'emplacement du `compare.sh` script documenté dans Asset Locations
  - `M2DFUTILS_VERBOSE`- [VERBEUX ou NORMAL]. Cela contrôle le niveau de détail de la `SYSPRINT` sortie
- Vérifiez que le chemin du module de charge est ajouté au `JES\Configuration\JES Program Path` paramètre
- Vérifiez que les scripts du répertoire des utilitaires disposent des autorisations d'exécution. Vous pouvez ajouter une autorisation d'exécution à l'aide de la `chmod + x <script name>` commande, dans l'environnement Linux

Syntaxe générale

DUMP

Permet de copier des fichiers de l'emplacement catalogué actuel vers un emplacement de sauvegarde. Cet emplacement doit actuellement être un système de fichiers.

Processus

DUMP effectuera les opérations suivantes :

1. Créez le répertoire de localisation cible.

2. Cataloguez le répertoire de localisation cible en tant que membre du PDS.
3. Déterminez les fichiers à inclure en traitant le paramètre INCLUDE.
4. Désélectionnez les fichiers inclus en traitant le paramètre EXCLUDE.
5. Déterminez si les fichiers à vider doivent être SUPPRIMÉS.
6. Mettez en file d'attente les fichiers à traiter.
7. Copiez les fichiers.
8. Exportez les informations DCB cataloguées des fichiers copiés vers un fichier secondaire situé à l'emplacement cible afin de faciliter les futures opérations de restauration.

Syntaxe

```
DUMP
TARGET ( TARGET LOCATION  )    -
INCLUDE ( DSN. )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Paramètres requis

Les paramètres requis pour DUMP sont les suivants :

- `SYSPRINT DD NAME`- Pour contenir des informations de journalisation supplémentaires
- `TARGET`- Emplacement cible. Il peut s'agir de :
  - Chemin complet de l'emplacement du dépotoir
  - Nom du sous-répertoire créé à l'emplacement défini dans la variable M2DFUTILS_BASE_LOC
- `INCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe
- `EXCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

Paramètres facultatifs

- ANNULER - Annulez en cas d'erreur. Les fichiers traités seront conservés
- (Par défaut) IGNORER - Ignore toute erreur et tout processus jusqu'à la fin

- SUPPRIMER - Si aucune erreur ENQ ne se produit, le fichier est supprimé et n'est pas catalogué

DELETE

Permet de supprimer et de décataloguer des fichiers en masse. Les fichiers ne sont pas sauvegardés.

Processus

DELETE effectuera les opérations suivantes :

1. Déterminez les fichiers à inclure en traitant le paramètre INCLUDE.
2. Désélectionnez les fichiers inclus en traitant le paramètre EXCLUDE.
3. Mettez en file d'attente les fichiers à traiter. Régler la disposition sur OLD, DELETE, KEEP.

Syntaxe

```
DELETE
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Paramètres requis

Les paramètres requis pour DELETE sont les suivants :

- `SYSPRINT DD NAME`- Pour contenir des informations de journalisation supplémentaires
- `INCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe
- `EXCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

Paramètres facultatifs

- ANNULER - Annulez en cas d'erreur. Les fichiers traités seront conservés
- (Par défaut) IGNORER - Ignore toute erreur et tout processus jusqu'à la fin

RESTORE

Permet de restaurer des fichiers précédemment sauvegardés à l'aide de DUMP. Les fichiers sont restaurés à leur emplacement catalogué d'origine, sauf si RENAME est utilisé pour modifier le DSNAME restauré.

Processus

RESTORE effectuera les opérations suivantes :

1. Validez le répertoire de localisation source.
2. Déterminez les fichiers à inclure en traitant le fichier d'exportation du catalogue.
3. Désélectionnez les fichiers inclus en traitant le paramètre EXCLUDE.
4. Mettez en file d'attente les fichiers à traiter.
5. Cataloguez les fichiers qui ne sont pas catalogués en fonction de leurs informations d'exportation.
6. Si un fichier est déjà catalogué et que les informations du catalogue d'exportation sont les mêmes, RESTORE remplacera le jeu de données catalogué si l'option REPLACE est définie.

Syntaxe

```
RESTORE
SOURCE ( TARGET LOCATION )
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ REPLACE]
```

Paramètres requis

Les paramètres requis pour RESTORE sont les suivants :

- `SYSPRINT DD NAME`- Pour contenir des informations de journalisation supplémentaires
- `SOURCE`- Emplacement de la source. Il peut s'agir de :
  - Chemin complet de l'emplacement du dépotoir
  - Nom du sous-répertoire créé à l'emplacement défini dans la variable M2DFUTILS_BASE_LOC
- `INCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

- `EXCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

Paramètres facultatifs

- ANNULER - Annulez en cas d'erreur. Fichiers traités conservés
- (Par défaut) IGNORER - Ignore toute erreur et tout processus jusqu'à la fin
- REMPLACER - Si le fichier à restaurer est déjà catalogué et que les notices du catalogue sont les mêmes, remplacez le fichier catalogué

Exemple JCLs

tâche DUMP

Cette tâche créera un sous-répertoire appelé`TESTDUMP`. Il s'agit de l'emplacement de sauvegarde par défaut spécifié par la variable M2DFUTILS_BASE_LOC. Il créera une bibliothèque PDS pour cette sauvegarde appelée`M2DFUTILS.TESTDUMP`. Les données du catalogue exportées sont stockées dans un fichier séquentiel de lignes situé dans le répertoire de sauvegarde appelé`CATDUMP.DAT`. Tous les fichiers sélectionnés seront copiés dans ce répertoire de sauvegarde.

```
//M2DFDMP JOB 'M2DFDMP',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDUMP.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSIN    DD *
DUMP TARGET(TESTDUMP)               -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 CANCEL
/*
//
```

SUPPRIMER une tâche

Cette tâche supprimera tous les fichiers du catalogue qui correspondent au paramètre INCLUDE.

```
/M2DFDEL JOB 'M2DFDEL',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDEL.SYSPRINT,
```

```
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DELETE                                -
     INCLUDE(TEST.FB.FILE*.ABC)         -
 CANCEL
/*
 //
```

tâche RESTORE

Cette tâche restaurera les fichiers correspondant au paramètre INCLUDE à partir de l'emplacement de TESTDUMP sauvegarde. Les fichiers catalogués seront remplacés si le fichier catalogué est le même que celui de l'export CATDUMP et si l'option REPLACE est spécifiée.

```
//M2DFREST JOB 'M2DFREST',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
////SYSPRINT DD DSN=TESTREST.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
RESTORE SOURCE(TESTDUMP)                -
     INCLUDE(TEST.FB.FILE*.ABC)         -
 IGNORE
 REPLACE
/*
//
```

Utilitaire batch M2RUNCMD

Vous pouvez utiliser M2RUNCMD, un utilitaire de traitement par lots, pour exécuter des commandes, des scripts et des appels système de Rocket Software (anciennement Micro Focus) directement depuis JCL au lieu de les exécuter à partir d'un terminal ou d'une invite de commande. Le résultat des commandes est enregistré dans le journal des spoules du traitement par lots.

Rubriques

- Plateformes prises en charge
- Configurer M2RUNCMD pour le runtime AWS Mainframe Modernization sur Amazon EC2 (y compris 2.0) AppStream

- Exemple JCLs

Plateformes prises en charge

Vous pouvez utiliser M2RUNCMD sur les plateformes suivantes :

- Rocket Software Runtime (sur Amazon EC2)
- Toutes les variantes des produits Rocket Software Enterprise Developer (ED) et Rocket Software Enterprise Server (ES).

Configurer M2RUNCMD pour le runtime AWS Mainframe Modernization sur Amazon EC2 (y compris 2.0) AppStream

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez M2RUNCMD comme suit.

- Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez deux points (:) pour séparer les chemins sous Linux des points-virgules (;) sous Windows.
  - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
  - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
  - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

Exemple JCLs

Pour tester l'installation, vous pouvez utiliser l'un des exemples suivants JCLs.

RUNSCRL1.jcl

Cet exemple de JCL crée un script et l'exécute. La première étape consiste à créer un script appelé `/tmp/TEST_SCRIPT.sh` et dont le contenu provient de données `SYSUT1` in-stream. La deuxième étape définit l'autorisation d'exécution et exécute le script créé lors de la première étape. Vous pouvez également choisir de n'effectuer que la deuxième étape pour exécuter le logiciel Rocket et les commandes système déjà existants.

```
//RUNSCRL1 JOB 'RUN SCRIPT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
```

```
//*--------------------------------------------------------------------*
//*  CREATE SCRIPT (LINUX)
//*--------------------------------------------------------------------*
//*
//STEP0010 EXEC PGM=IEBGENER
//*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//SYSUT1   DD *
#!/bin/bash

set -x

## ECHO PATH ENVIRONMNET VARIABLE
echo $PATH

## CLOSE/DISABLE VSAM FILE
casfile -r$ES_SERVER -oc  -ed -dACCTFIL

## OPEN/ENABLE VSAM FILE
casfile -r$ES_SERVER -ooi -ee -dACCTFIL

exit $?
/*
//SYSUT2   DD DSN=&&TEMP,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=LSEQ,LRECL=300,DSORG=PS,BLKSIZE=0)
//*MFE: %PCDSN='/tmp/TEST_SCRIPT.sh'
//*
//*--------------------------------------------------------------------*
//*   RUN SCRIPT (LINUX)                                               *
//*--------------------------------------------------------------------*
//*
//STEP0020 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*RUN SCRIPT
 sh /tmp/TEST_SCRIPT.sh
/*
//
```

SYSOUT

Le résultat de la commande ou du script exécuté est écrit dans le SYS0UT journal. Pour chaque commande exécutée, il affiche la commande, le code de sortie et le code de retour.

```
************ CMD Start ************

CMD_STR: sh /tmp/TEST_SCRIPT.sh

CMD_OUT:


+ echo /opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
/opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
+ casfile -rMYDEV -oc -ed -dACCTFIL

-Return Code:   0

Highest return code:    0

+ casfile -rMYDEV -ooi -ee -dACCTFIL

-Return Code:   8

Highest return code:    8

+ exit 8


CMD_RC=8


************  CMD End  ************
```

RUNCMDL1.jcl

Cet exemple de JCL utilise RUNCMD pour exécuter plusieurs commandes.

```
//RUNCMDL1 JOB 'RUN CMD',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*   RUN SYSTEM COMMANDS                                              *
```

```
//*--------------------------------------------------------------------*
//*
//STEP0001 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*LIST DIRECTORY
 ls
*ECHO PATH ENVIRONMNET VARIABLE
 echo $PATH
/*
//
```

Transfert de fichiers dans le cadre de la AWS modernisation du mainframe

AWS Mainframe Modernization File Transfer vous permet de transférer et de convertir des ensembles de données du mainframe vers Amazon S3 pour des cas d'utilisation liés à la modernisation, à la migration et à l'augmentation du mainframe. Il simplifie le processus de transfert des ensembles de données de votre ordinateur central vers le AWS Cloud. Les principales fonctionnalités incluent : la découverte des ensembles de données et des artefacts sources du mainframe, ainsi que l'évolutivité et l'efficacité pour des transferts de données plus rapides vers Amazon S3. Le transfert de fichiers prend en charge différents types d'ensembles de données mainframe tels que séquentiel, PDS, GDS, GDG et VSAM KSDS. Le service transfère les ensembles de données vers un compartiment Amazon S3 intermédiaire, les convertit dans la page de code cible spécifiée, puis les déplace vers le compartiment S3 cible de votre choix.

Rubriques

- Qu'est-ce que le transfert de fichiers pour la modernisation du mainframe AWS ?
- Installation d'un agent de transfert de fichiers
- Configuration d'un agent de transfert de fichiers
- Création de points de terminaison de transfert de données pour le transfert de fichiers
- Création de tâches de transfert dans File Transfer
- Tutoriel : Démarrage avec le transfert de fichiers AWS Mainframe Modernization
- Encodages source et cible pris en charge dans le transfert de fichiers AWS Mainframe Modernization

Qu'est-ce que le transfert de fichiers pour la modernisation du mainframe AWS ?

Avec AWS Mainframe Modernization File Transfer, vous pouvez transférer et convertir des ensembles de données et des fichiers à l'aide d'un service entièrement géré afin d'accélérer et de simplifier les cas d'utilisation de la modernisation, de la migration et de l'augmentation vers le service AWS Mainframe Modernization et Amazon S3.

Rubriques

- Avantages du transfert de fichiers lié à la modernisation du mainframe AWS
- Comment fonctionne le transfert de fichiers pour la modernisation du mainframe AWS

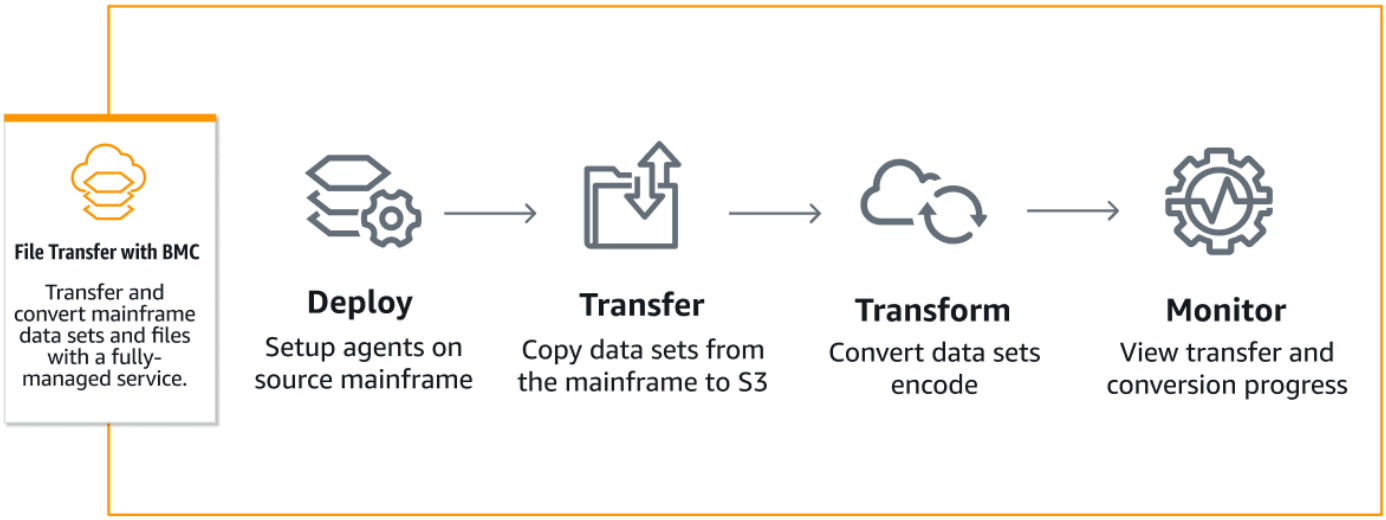
Avantages du transfert de fichiers lié à la modernisation du mainframe AWS

AWS Mainframe Modernization File Transfer vous aide à transférer des ensembles de données du mainframe vers Amazon S3. Certains avantages incluent :

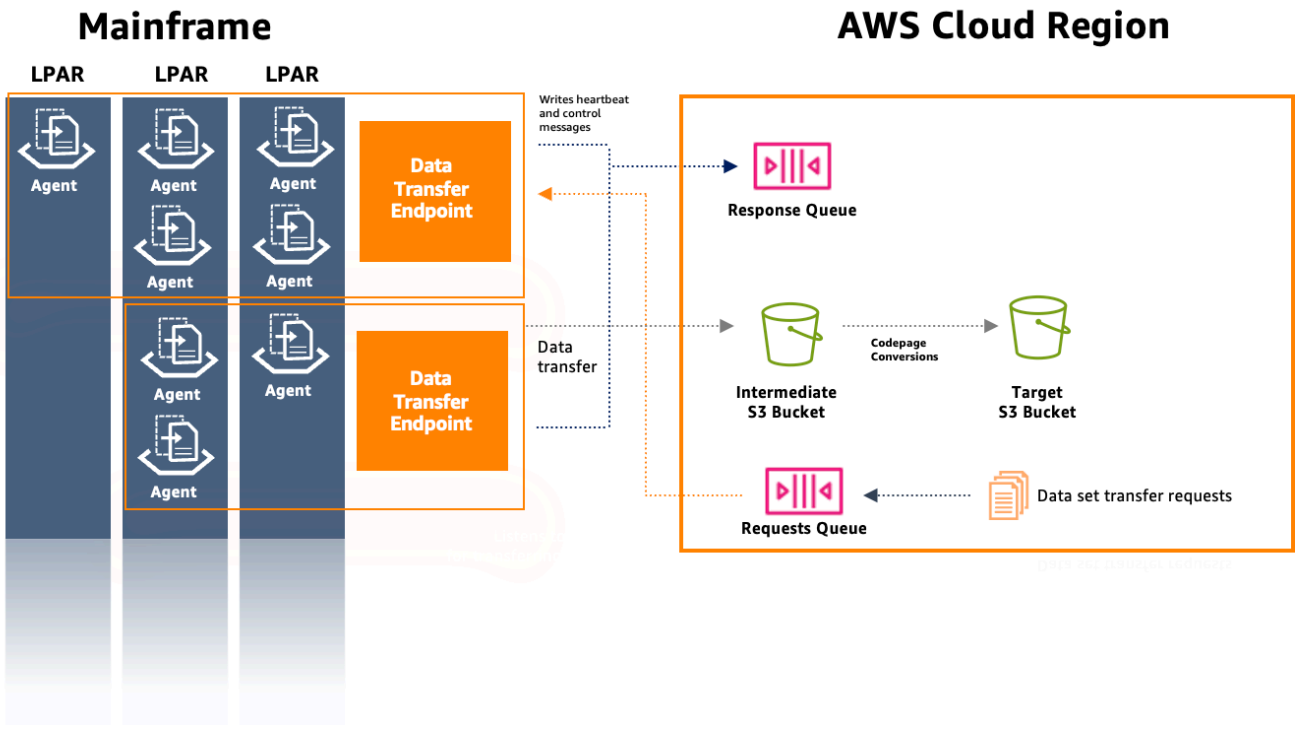
- Découverte des ensembles de données et artefacts sources du mainframe
- Transferts automatisés et conversion des ensembles de données
- Évolutivité, efficacité et rapidité pour accélérer les transferts de jeux de données vers AWS

Comment fonctionne le transfert de fichiers pour la modernisation du mainframe AWS

La figure suivante donne un aperçu du fonctionnement conceptuel du transfert de fichiers pour la modernisation du mainframe AWS.



La figure suivante est une présentation architecturale de la fonctionnalité de transfert de fichiers de modernisation du mainframe AWS.



Installation d'un agent de transfert de fichiers

Vous pouvez utiliser ce document comme step-by-step guide pour installer un agent sur le mainframe source.

> **Note**
>
> Ce guide s'adresse uniquement aux programmeurs de systèmes mainframe.

Rubriques

- Étape 1 : créer un ensemble de données ZFS pour l'agent M2
- Étape 2 : Formater l'ensemble de données au format ZFS
- Étape 3 : monter le système de fichiers
- Étape 4 : vérifier le support
- Étape 5 : Entrez OMVS
- Étape 6 : définir la variable d'environnement du répertoire d'installation de l'agent
- Étape 7 : définir la variable d'environnement du répertoire de travail

- Étape 8 : Création du répertoire de travail
- Étape 9 : Copiez le fichier tar de l'agent et copiez le répertoire de travail
- Étape 10 : terminer l'installation de l'agent

Étape 1 : créer un ensemble de données ZFS pour l'agent M2

Créez un fichier ZFS pour l'installation de l'agent M2 à l'aide du langage JCL (Job Control Language) ci-dessous :

```
DEFINE   EXEC   PGM=IDCAMS
SYSPRINT DD     SYSOUT=A
SYSIN    DD     *
 DEFINE CLUSTER (NAME(yourhlq.M2AGENT.ZFS) -
 VOLUMES(*) -
 LINEAR CYL(1000 200))
```

Étape 2 : Formater l'ensemble de données au format ZFS

Après avoir créé le jeu de données, formatez-le en tant que système de fichiers ZFS.

Pour ce faire, vous pouvez utiliser le JCL suivant :

```
FORMAT  EXEC PGM=IOEAGFMT,
PARM=('-aggregate yourhlq.M2AGENT.ZFS -size 1200') ,
SYSPRINT DD  SYSOUT=*
```

Soumettez cette tâche et vérifiez si elle s'est terminée correctement.

Étape 3 : monter le système de fichiers

Pour monter le système de fichiers, utilisez la `MOUNT` commande. Vous pouvez monter le système de fichiers en ligne de commande, dans ISPF ou par lots.

Par exemple :

```
MOUNT FILESYSTEM('yourhlq.M2AGENT.ZFS') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/usr/lpp/aws/
m2-agent')
```

Vous utiliserez ce point de montage à l'étape 6.

> **Note**
>
> La définition du chemin de montage est facultative et vous devez utiliser un répertoire existant pour cela.

Étape 4 : vérifier le support

Vérifiez que le système de fichiers est correctement monté à l'aide de `D OMVS,F` la commande ou en vérifiant dans Unix System Service (USS).

Étape 5 : Entrez OMVS

Utilisez la commande suivante pour entrer dans OMVS :

```
TSO OMVS
```

Étape 6 : définir la variable d'environnement du répertoire d'installation de l'agent

Utilisez la commande suivante pour définir l'environnement du répertoire d'installation de l'agent :

```
export AGENT_DIR=/usr/lpp/aws/m2-agent
```

> **Note**
>
> Le point de montage est défini à l'étape 3.

Étape 7 : définir la variable d'environnement du répertoire de travail

Utilisez la commande suivante pour définir la variable d'environnement du répertoire de travail :

```
export WORK_DIR=$AGENT_DIR/tmp
```

Étape 8 : Création du répertoire de travail

Utilisez la commande suivante pour définir l'environnement du répertoire de travail :

```
mkdir -p $WORK_DIR
```

Étape 9 : Copiez le fichier tar de l'agent et copiez le répertoire de travail

Téléchargez le fichier tar de l'agent depuis AWS en utilisant le lien de l'agent M2.

Le mécanisme de transfert dépend de votre environnement, mais assurez-vous que le fichier tar est transféré en mode binaire.

Étape 10 : terminer l'installation de l'agent

Suivez ces étapes pour terminer l'installation de l'agent.

1. Définissez la variable d'environnement m2-agent version sur la version actuellement installée à l'aide de la commande suivante :

   ```
   export M2_AGENT_VERSION=1.0.0
   ```

2. Extrayez le package tar de l'agent à l'aide de la commande suivante :

   ```
   tar -xpf m2-agent-$M2_AGENT_VERSION.tar -C $AGENT_DIR
   ```

3. Créez un lien `current-version` symbolique vers le répertoire d'installation actuel de l'agent à l'aide de la commande suivante :

   ```
   ln -s $AGENT_DIR/m2-agent-v$M2_AGENT_VERSION $AGENT_DIR/current-version
   ```

4. Mettez à jour et soumettez `CPY#PDS` pour créer les ensembles de données de l'agent de transfert de fichiers.

   > **Note**
   >
   > JCL utilise le. `SYS2.AWS.M2 HLQ`

   Pour créer l'agent de transfert de fichiers, mettez à jour les trois variables symboliques HLQ (High Level Qualifier)`VOLSER`, `AGNTPATH` à utiliser ultérieurement dans la JCL :

   ```
   oedit $AGENT_DIR/current-version/installation/CPY#PDS
   ```

> **Note**
>
> Cette JCL est conçue pour configurer certains aspects de l'installation de l'agent sur le mainframe. Il alloue les ensembles de données nécessaires, puis copie des fichiers spécifiques du système de fichiers Unix vers ces ensembles de données.

Configuration d'un agent de transfert de fichiers

Une fois que vous avez installé un agent de transfert de fichiers, procédez comme suit pour configurer l'agent. Si vous devez installer un nouvel agent, suivez les instructions de la the section called "Installation d'un agent de transfert de fichiers" page.

Rubriques

- Étape 1 : configurer les autorisations et démarrer le contrôle des tâches (STC)
- Étape 2 : créer des compartiments Amazon S3
- Étape 3 : Création d'une clé gérée par AWS KMS le client pour le chiffrement
- Étape 4 : Création d'un AWS Secrets Manager secret pour les informations d'identification du mainframe
- Étape 5 : Création d'une politique IAM
- Étape 6 : Création d'un utilisateur IAM avec des informations d'identification d'accès à long terme
- Étape 7 : Création d'un rôle IAM que l'agent devra assumer
- Étape 8 : Configuration de l'agent

Étape 1 : configurer les autorisations et démarrer le contrôle des tâches (STC)

1. Mettez à jour et soumettez l'un des `SYS2.AWS.M2.SAMPLIB(SEC#RACF)` (pour configurer les autorisations RACF) ou `SYS2.AWS.M2.SAMPLIB(SEC#TSS)` (pour configurer les autorisations TSS) conformément à leurs instructions. Ces membres ont été créés à l'`CPY#PDS`étape précédente.

> **Note**
>
> `SYS2.AWS.M2`doit être remplacé par le qualificatif de haut niveau (HLQ) choisi lors de l'installation.

2. Mettez à jour l'exportation PWD dans la JCL `SYS2.AWS.M2.SAMPLIB(M2AGENT)` STC, si le chemin de répertoire par défaut de l'agent de transfert de fichiers (`/usr/lpp/aws/m2-agent`) a été modifié.
3. Mettez à jour le PROC conformément aux normes de votre site :
   a. Mettez à jour la carte PROC conformément à vos exigences d'installation.
   b. Mettez à jour le STEPLIB avec le. `M2 LOADLIB PDSE ALIAS`
   c. Modifiez le fichier PWD pour indiquer le chemin d'installation de l'agent (seul celui-ci est inclus).
   d. Mettez à jour `JAVA_HOME` si nécessaire.
4. Mettez à jour et copiez le `SYS2.AWS.M2.SAMPLIB(M2AGENT)` JCL PROCLIBs dans `SYS1.PROCLIB` ou l'un des fichiers de votre `PROCLIB` concaténation.
5. Ajoutez `SYS2.AWS.M2.LOADLIB` à la liste APF à l'aide de la commande suivante :

```
SETPROG APF ADD DSNAME(SYS2.AWS.M2.LOADLIB) SMS
```

6. Définissez le groupe et le propriétaire de l'agent sur l'agent (user/group (M2USER/M2GROUP). Utilisez la commande suivante dans l'OMVS :

```
chown -R M2USER:M2GROUP $AGENT_DIR/current-version
```

> **Note**
>
> Modifiez le M2USER et le M2GROUP avec les noms que vous avez utilisés dans le travail de définition de sécurité.

Étape 2 : créer des compartiments Amazon S3

Le transfert de fichiers AWS Mainframe Modernization nécessite un compartiment Amazon S3 intermédiaire comme zone de travail. Nous vous recommandons de créer un bucket spécialement pour cela.

Vous pouvez éventuellement créer un nouveau compartiment Amazon S3 cible pour les ensembles de données transférés. Sinon, vous pouvez également utiliser votre compartiment Amazon S3 existant. Pour plus d'informations sur la création de compartiments Amazon S3, consultez Création d'un compartiment.

Étape 3 : Création d'une clé gérée par AWS KMS le client pour le chiffrement

Pour créer une clé gérée par le client dans AWS KMS

1. Ouvrez la AWS KMS console à l'adressehttps://console.aws.amazon.com/kms.
2. Choisissez Clés gérées par le client dans le volet de navigation de gauche.
3. Choisissez Create key.
4. Sous Configurer la clé, choisissez le type de clé symétrique et Utilisation de la clé comme Chiffrer et déchiffrer. Utilisez d'autres configurations par défaut.
5. Choisissez Next (Suivant).
6. Dans Ajouter des étiquettes, ajoutez un alias et une description pour votre clé.
7. Choisissez Next (Suivant).
8. Sous Définir les autorisations administratives clés, choisissez au moins un utilisateur et un rôle IAM qui administrent cette clé.
9. Choisissez Next (Suivant).
10. Facultativement, sous Définir les autorisations administratives clés, choisissez au moins un utilisateur et un rôle IAM autorisés à utiliser cette clé.
11. Choisissez Next (Suivant).
12. Dans la section Modifier la politique clé, choisissez Modifier et ajoutez la syntaxe suivante à la politique clé. Cela permet au service de modernisation du AWS mainframe de lire et d'utiliser ces clés pour le chiffrement/déchiffrement.

> **Important**
>
> Ajoutez l'instruction aux instructions existantes. Ne remplacez pas ce qui figure déjà dans la politique.

```
{
    "Sid" : "Enable AWS M2 File Transfer Permissions",
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
    },
    "Action" : [
        "kms:Encrypt",
        "kms:Decrypt"
    ],
   "Resource" : "*"
},
```

13. Choisissez Next (Suivant).
14. Sur la page Révision, vérifiez tous les détails, puis choisissez Terminer.

Copiez et enregistrez l'ARN de la clé gérée par le client en ouvrant la clé KMS nouvellement créée. Il sera utilisé ultérieurement dans la politique.

Étape 4 : Création d'un AWS Secrets Manager secret pour les informations d'identification du mainframe

Les informations d'identification du mainframe sont nécessaires pour accéder aux ensembles de données à transférer et elles doivent être conservées en AWS Secrets Manager secret.

Pour créer un AWS Secrets Manager secret

1. Ouvrez la console du gestionnaire de secrets à l'adressehttps://console.aws.amazon.com/secretsmanager.
2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Dans Choisir le type de secret, choisissez Autre type de secret.

4. Utilisez la valeur clé de `userId` l'userID du mainframe qui a accès aux ensembles de données. Utilisez la valeur clé `password` pour le champ du mot de passe.
5. Pour la clé de chiffrement, choisissez la clé gérée par le AWS client créée précédemment.
6. Choisissez Next (Suivant).
7. Sur la page Configurer le secret, saisissez un nom et une description.
8. Sur la même page, modifiez les autorisations relatives aux ressources et utilisez la politique de ressources suivante afin que le service de modernisation du AWS mainframe puisse y accéder.

```
{
  "Version" : "2012-10-17",
  "Statement" : [ {
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
    },
    "Action" : [ "secretsmanager:GetSecretValue",
                 "secretsmanager:DescribeSecret" ],
    "Resource" : "*"
  } ]
}
```

9. Choisissez Enregistrer pour enregistrer les autorisations mises à jour.
10. Choisissez Next (Suivant).
11. Passez par la page Configurer les rotations, puis choisissez Next.
12. Sur la page de révision, vérifiez toutes les configurations et choisissez Store pour enregistrer le secret.

> **Important**
>
> Les clés `password` secrètes `userId` et majuscules distinguent les majuscules des minuscules et doivent être saisies comme indiqué.

Étape 5 : Création d'une politique IAM

Pour créer une nouvelle politique avec les autorisations requises pour l'agent

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam.

2. Choisissez Politiques sous Gestion des accès.
3. Sélectionnez Créer une politique.
4. Sur la page Spécifier les autorisations, sous Éditeur de politiques, passez de l'éditeur visuel à l'éditeur JSON et remplacez le contenu par le modèle suivant :
5.
```
{
"Version": "2012-10-17",
    "Statement": [
        {
         "Sid": "FileTransferAgentSQSReceive",
            "Effect": "Allow",
            "Action": [
                "sqs:DeleteMessage",
                "sqs:ReceiveMessage"
            ],
            "Resource": "arn:aws:sqs:*:111122223333:m2-*-request-queue.fifo"
        },
        {
         "Sid": "FileTransferAgentSQSSend",
            "Effect": "Allow",
            "Action": "sqs:SendMessage",
            "Resource": "arn:aws:sqs:*:111122223333:m2-*-response-queue.fifo"
        },
        {
         "Sid": "FileTransferWorkingS3",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "<file-transfer-endpoint-intermediate-bucket-arn>/*"
        },
        {
         "Sid": "FileTransferAgentKMSDecrypt",
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "<kms-key-arn>"
        }
    ]
}
```
6. Remplacez les ARN 111122223333 de la file de demande et de la file de réponse par votre compte.

> **Note**
>
> Il s'agit d'ARN génériques qui correspondent aux deux files d'attente Amazon SQS créées lors de l'initialisation du point de terminaison de transfert de données. Après avoir créé un point de terminaison de transfert de fichiers, remplacez éventuellement ces ARN par les valeurs réelles d'Amazon SQS.

7. `file-transfer-endpoint-intermediate-bucket-arn`Remplacez-le par l'ARN du bucket de transfert créé précédemment. Laissez le caractère générique « /* » à la fin.
8. `kms-key-arn`Remplacez-le par l'ARN de la AWS KMS clé créée précédemment.
9. Choisissez Next (Suivant).
10. Sur la page Réviser et créer, ajoutez le nom et la description de la politique.
11. Sélectionnez Créer une politique.

Étape 6 : Création d'un utilisateur IAM avec des informations d'identification d'accès à long terme

Créez un utilisateur IAM qui autorise l'agent mainframe à se connecter à votre AWS compte. L'agent se connectera à cet utilisateur, puis assumera un rôle que vous définissez avec les autorisations nécessaires pour utiliser les files d'attente de réponses et de demandes Amazon SQS et pour enregistrer des ensembles de données dans des compartiments Amazon S3.

Pour créer cet utilisateur IAM

1. Accédez à la console IAM à https://console.aws.amazon.com/iam l'adresse.
2. Choisissez Utilisateurs sous Gestion des accès.
3. Choisissez Create user (Créer un utilisateur).
4. Ajoutez un nom d'utilisateur significatif sous Détails de l'utilisateur. Par exemple, `Configure-ft-agent`.
5. Choisissez Next (Suivant).
6. Dans les options d'autorisations, choisissez l'option Joindre directement les politiques, mais n'attachez aucune politique d'autorisation. Ces autorisations seront gérées par un rôle qui y sera rattaché.
7. Choisissez Next (Suivant).

8. Vérifiez les détails, puis choisissez Create user.
9. Une fois l'utilisateur créé, choisissez-le et ouvrez l'onglet Informations d'identification de sécurité.
10. Sous Clés d'accès, choisissez Créer une clé d'accès.
11. Choisissez ensuite Autre lorsque vous êtes invité à saisir Cas d'utilisation.
12. Choisissez Next (Suivant).
13. Vous pouvez éventuellement définir une balise de description telle que,`Access key for configuring file transfer agent`.
14. Choisissez Create access key (Créer une clé d'accès).
15. Copiez et enregistrez en toute sécurité la clé d'accès et la clé d'accès secrète générées. Ils seront utilisés ultérieurement.

Pour plus d'informations sur la création d'une clé d'accès IAM, consultez la section Gestion des clés d'accès pour les utilisateurs IAM.

> **Important**
>
> Enregistrez la clé d'accès et la clé d'accès secrète affichées sur la dernière page de l'assistant de création de clé d'accès, avant de choisir OK. Ces clés sont utilisées pour configurer l'agent mainframe et ne peuvent pas être récupérées ultérieurement.

> **Note**
>
> Enregistrez l'ARN de l'utilisateur IAM utilisé pour établir une relation de confiance avec un rôle IAM.

Étape 7 : Création d'un rôle IAM que l'agent devra assumer

Pour créer un nouveau rôle IAM pour l'agent

1. Choisissez Rôles dans la console IAM à https://console.aws.amazon.com/iam l'adresse.
2. Sélectionnez Create role (Créer un rôle).
3. Sur la page Sélectionner une entité de confiance, choisissez Politique de confiance personnalisée pour le type d'entité de confiance.

4. Remplacez la politique de confiance personnalisée par la suivante et `<iam-user-arn>` remplacez-la par l'ARN de l'utilisateur créé précédemment.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
         "Sid": "FileTransferAgent",
         "Effect": "Allow",
         "Principal": {
            "AWS": "<IAM-User-arn>"
         },
         "Action": "sts:AssumeRole"
    } ]
}
```

5. Choisissez Next (Suivant).
6. Dans Ajouter des autorisations, filtrez le nom de la politique que vous avez créé précédemment et choisissez-le.
7. Choisissez Next (Suivant).
8. Nommez le rôle, puis choisissez Create Role.

> **Note**
>
> Enregistrez le nom du rôle, que vous utiliserez ultérieurement pour configurer l'agent mainframe.

Étape 8 : Configuration de l'agent

Pour configurer l'agent de transfert de fichiers

1. Accédez à `$AGENT_DIR/current-version/config`.
2. Modifiez le fichier de configuration de l'agent `appication.properties` pour ajouter une configuration d'environnement à l'aide de la commande suivante :

```
oedit $AGENT_DIR/current-version/config/application.properties
```

   Par exemple :

```
agent.environments[0].account-id=<AWS_ACCOUNT_ID>
agent.environments[0].agent-role-name=<AWS_IAM_ROLE_NAME>
agent.environments[0].access-key-id=<AWS_IAM_ROLE_ACCESS_KEY>
agent.environments[0].secret-access-id=<AWS_IAM_ROLE_SECRET_KEY>
agent.environments[0].bucket-name=<AWS_S3_BUCKET_NAME>
agent.environments[0].environment-name=<AWS_REGION>
agent.environments[0].region=<AWS_REGION>
zos.complex-name=<File_Transfer_Endpoint_Name>
```

Où :

- `AWS_ACCOUNT_ID`est l'identifiant du AWS compte.
- `AWS_IAM_ROLE_NAME`est le nom du rôle IAM créé dans lethe section called "Étape 7 : Création d'un rôle IAM que l'agent devra assumer".
- `AWS_IAM_ROLE_ACCESS_KEY`est la clé d'accès de l'utilisateur IAM créé dansthe section called "Étape 6 : Création d'un utilisateur IAM avec des informations d'identification d'accès à long terme".
- `AWS_IAM_ROLE_SECRET_KEY`est la clé secrète d'accès de l'utilisateur IAM créé dansthe section called "Étape 6 : Création d'un utilisateur IAM avec des informations d'identification d'accès à long terme".
- `AWS_S3_BUCKET_NAME`est le nom du compartiment de transfert créé avec le point de terminaison du transfert de données.
- `AWS_REGION`est la région dans laquelle vous configurez l'agent de transfert de fichiers.

> **Note**
>
> Vous pouvez transférer l'agent de transfert de fichiers vers plusieurs régions et comptes en AWS définissant plusieurs environnements.

- (Facultatif). `zos.complex-name`est le nom complexe que vous avez créé lors de la création d'un point de terminaison de transfert de fichiers.

> **Note**
>
> Ce champ n'est nécessaire que si vous souhaitez personnaliser le nom du complexe (qui est par défaut votre nom sysplex) identique à celui que vous avez défini lors de la création de votre point de terminaison de transfert de fichiers. Pour de plus amples

informations, veuillez consulter the section called "Création de points de terminaison de transfert de données".

> **Important**
>
> Il peut y avoir plusieurs sections de ce type, à condition que l'index entre crochets — `[0]` — soit incrémenté pour chacune d'entre elles.

Vous devez redémarrer l'agent pour que les modifications prennent effet.

Prérequis

1. Lorsqu'un paramètre est ajouté ou supprimé, l'agent doit être arrêté et démarré. Démarrez l'agent de transfert de fichiers à l'aide de la commande suivante dans la CLI :

   ```
   /S M2AGENT
   ```

   Pour arrêter l'agent M2, utilisez la commande suivante dans la CLI :

   ```
   /P M2AGENT
   ```

2. Vous pouvez configurer l'agent de transfert de fichiers pour transférer des données vers plusieurs régions et comptes en AWS définissant des entrées d'environnement.

   > **Note**
   >
   > Remplacez les valeurs par les valeurs de paramètres que vous avez créées et configurées précédemment.

   ```
   #Region 1
   agent.environments[0].account-id=AWS_ACCOUNT_ID
   agent.environments[0].agent-role-name=AWS_IAM_ROLE_NAME
   agent.environments[0].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
   agent.environments[0].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
   agent.environments[0].bucket-name=AWS_S3_BUCKET_NAME
   agent.environments[0].environment-name=AWS_REGION
   ```

```
agent.environments[0].region=AWS_REGION

#Region 2
agent.environments[1].account-id=AWS_ACCOUNT_ID
agent.environments[1].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[1].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[1].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[1].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[1].environment-name=AWS_REGION
agent.environments[1].region=AWS_REGION
```

Création de points de terminaison de transfert de données pour le transfert de fichiers

Les points de terminaison de transfert de données permettent la connectivité avec le mainframe source et garantissent une haute disponibilité, une évolutivité et une gestion rationalisée des agents. Les agents individuels sont installés sur le mainframe LPARs et peuvent être regroupés dans un terminal de transfert de données. Lorsqu'une demande est faite pour transférer un ensemble de données, un agent du point de terminaison du transfert de données gère ce transfert spécifique. Pour initier des transferts de données, au moins un agent du point de terminaison de transfert de données doit être en ligne.

Cette procédure suppose que vous avez terminé les étapes de configuration de Configuration pour la modernisation du AWS mainframe l'agent de transfert de fichiers sur le mainframe source.

Création de points de terminaison de transfert de données

Pour créer des points de terminaison de transfert de données pour le transfert de fichiers, procédez comme suit dans la console de modernisation du AWS mainframe.

Pour créer un point de terminaison de transfert de données

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez transférer les fichiers de votre mainframe vers un compartiment Amazon S3.
3. Sur la page Points de terminaison de transfert de données, sous Transfert de fichiers, choisissez Créer un point de terminaison de transfert de données.

4. Sur la page Conditions requises pour le terminal de transfert de données, lisez toutes les instructions pour vous assurer que vous avez effectué ces étapes sur le mainframe source. Une fois confirmé, choisissez Next.
5. Sur la page Configurer le point de terminaison de transfert de données, ajoutez des informations de base pour votre point de terminaison de transfert de données.

    1. Dans la section des informations de base, entrez le nom de votre point de terminaison de transfert de données.

        > **Note**
        >
        > Le nom du point de terminaison du transfert de données doit correspondre au nom Sysplex, sauf si vous spécifiez un nom complexe dans la configuration de l'agent.

    2. Description facultative.
    3. La clé KMS utilisée pour chiffrer le secret.

        > **Note**
        >
        > Vous devez ajouter la politique basée sur les ressources suivante pour KMS afin que le service de modernisation du AWS mainframe puisse lire et utiliser ces clés pour le chiffrement/déchiffrement :

        ```
        {
          "Sid" : "Enable AWS M2 Permissions",
          "Effect" : "Allow",
          "Principal" : {
              "Service" : "m2.amazonaws.com"
                    },
           "Action" : [
               "kms:Encrypt",
               "kms:Decrypt"
                      ],
           "Resource" : "*"
        }
        ```

    4. Spécifiez l'emplacement S3 pour les données intermédiaires, qui est l'emplacement S3 intermédiaire où les ensembles de données transférés depuis le mainframe sont stockés avant d'être convertis et transférés vers le compartiment Amazon S3 cible.

.

> **Note**
>
> Il est recommandé de créer un nouveau compartiment Amazon S3 pour vos tâches de transfert. Pour plus d'informations, consultez la section Création d'un bucket. Vous pouvez également parcourir vos compartiments Amazon S3 existants en choisissant l'option Parcourir S3.

5. Après avoir saisi les champs obligatoires, choisissez Next.
6. Sur la page Vérifier et créer un point de terminaison de transfert de données, vérifiez si vous avez rempli les conditions requises et consultez les informations de base. Une fois confirmé, choisissez Créer un point de terminaison de transfert de données.

Vous serez redirigé vers la page de présentation des points de terminaison de transfert de données où vous pourrez voir la liste de tous les points de terminaison de transfert de données. Vous pourrez également voir les points de terminaison de transfert de données disponibles ou en panne.

Vous pouvez également rechercher des points de terminaison de transfert de données par nom et accéder à des informations supplémentaires pour chaque agent disponible.

Création de tâches de transfert dans File Transfer

Les tâches de transfert sont utilisées pour spécifier les ensembles de données à transférer du mainframe vers Amazon S3 et vous permettent de choisir les options de conversion des pages de code.

Ces instructions supposent que vous avez effectué les étapes Configuration pour la modernisation du AWS mainframe et que vous avez crééthe section called "Création de points de terminaison de transfert de données".

Rubriques

- Création de tâches de transfert
- Afficher les tâches de transfert

Création de tâches de transfert

Pour créer des tâches de transfert dans File Transfer, procédez comme suit dans la console AWS Mainframe Modernization.

Pour créer une tâche de transfert

> **Important**
>
> Vous devez disposer d'au moins un point de terminaison de transfert de données pour créer de nouvelles tâches de transfert.

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez transférer les fichiers de votre mainframe vers un compartiment Amazon S3.
3. Sur la page Transférer des tâches, vous pouvez choisir n'importe quel point de terminaison de transfert de données pour créer des tâches de transfert.
4. Sur la page Créer une tâche de transfert, configurez les propriétés de votre tâche de transfert. Si vous n'avez créé aucune tâche de transfert auparavant, vous pouvez créer la première en choisissant l'option Créer une tâche de transfert.

   - Sur cette page, entrez les informations de base de votre tâche de transfert, notamment le nom, la description et la clé secrète de la tâche de transfert.

     > **Note**
     >
     > - Chiffrez le secret à l'aide de la clé KMS définie avec le point de terminaison du transfert de données. Le secret doit contenir les informations d'identification du mainframe nécessaires pour accéder aux ensembles de données du mainframe à l'aide des touches `userId` et`password`. Pour plus d'informations, consultez le secret d'AWS Secrets Manager.
     > - Vous devez configurer la clé secrète avec la politique basée sur les ressources suivante afin que le service AWS Mainframe Modernization puisse y accéder pour effectuer une tâche de transfert de données.
     >
     > ```
     > {
     > ```

```
  "Version" : "2012-10-17",
  "Statement" : [ {
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
    },
    "Action" : [ "secretsmanager:GetSecretValue",
                 "secretsmanager:DescribeSecret" ],
    "Resource" : "*"
  } ]
}
```

> **Note**
>
> La taille maximale du jeu de données actuellement prise en charge pour le transfert est de 90 Go.

- Sélectionnez ensuite l'emplacement du compartiment Amazon S3 cible où les ensembles de données cibles provenant du mainframe seront transférés.
- Le point de terminaison de transfert de données sélectionné précédemment sera sélectionné. Vous pouvez également sélectionner un autre point de terminaison parmi les points de terminaison disponibles.

5. Choisissez Next (Suivant).
6. Sur la page Ajouter des ensembles de données, dans la section Configuration des tâches de transfert, vous pouvez choisir de configurer votre tâche de transfert en mode binaire ou de convertir et de transférer vos ensembles de données.

   - L'option de transfert en mode binaire vous permet de transférer des ensembles de données en évitant les conversions de pages de code et en conservant leurs octets RDW (Record Descriptor Word).
   - L'option Transférer et convertir des ensembles de données vous permet de transférer des ensembles de données en définissant les pages de code source et cible de vos ensembles de données. Vous pouvez voir les pages de code disponibles pour le transfert de fichiers sur the section called “Pages de code source et cible prises en charge” cette page.

7. Entrez votre requête dans le mainframe de recherche d'ensembles de données pour rechercher sur le mainframe les ensembles de données à inclure dans votre tâche de transfert. Choisissez Afficher les ensembles de données.

   Les symboles génériques suivants peuvent être utilisés dans le cadre des critères de recherche des ensembles de données pour les ordinateurs centraux :

   - Un seul astérisque (*) utilisé comme qualificatif (entre les points ou après la dernière période) correspond à un seul qualificatif occupant cette position.
   - Un seul astérisque (*) dans un qualificatif correspond à zéro caractère ou plus à cette position.
   - Un double astérisque (**) en tant que qualificatif (entre les périodes ou après la dernière période) correspond à zéro ou plusieurs qualificatifs à cette position.
   - Un double astérisque (**) dans un qualificatif n'est pas une requête valide.
   - Un signe de pourcentage (%) correspond à n'importe quel caractère alphanumérique ou national à cette position. Vous pouvez utiliser jusqu'à huit pour cent de signes dans chaque qualificatif.

   > **Note**
   >
   > Nous vous suggérons de toujours terminer vos critères de recherche par un point suivi d'un double astérisque (.**), puis d'affiner davantage la recherche, si nécessaire.

   Pour plus d'informations sur les règles relatives aux caractères génériques, consultez la section Filtrer les noms des ensembles de données dans la documentation IBM.

8. Ces ensembles de données seront chargés dans la section Ensembles de données du mainframe, où vous pourrez rechercher ou choisir un ou plusieurs ensembles de données pour lesquels vous souhaitez configurer les conversions de pages de code. Les ensembles de données sélectionnés seront affichés dans la section Ensembles de données ajoutés. Si aucun ensemble de données n'est chargé, vous devez revenir à l'étape 7.

   > **Note**
   >
   > Vous pouvez sélectionner des ensembles de données à partir de plusieurs requêtes de recherche et les ajouter à votre tâche de transfert.

9. Dans la section Ensembles de données ajoutés, vous verrez le nom, le type et le nom du volume de vos ensembles de données.

> **Important**
>
> Pour l'option Transférer et convertir des ensembles de données, vous devez saisir manuellement la page de code source et la page de code cible pour chacun des ensembles de données que vous avez choisis. La page de code source est le format de l'ensemble de données source, et la page de code cible est le format du jeu de données cible utilisé pour convertir les ensembles de données et les stocker dans le compartiment Amazon S3 cible.

10. Après avoir confirmé les ensembles de données dans la section Ensembles de données ajoutés (et les pages de code source et cible pour l'option Transférer et convertir des ensembles de données), choisissez Next.
11. Sur la page Réviser et créer, vous pouvez consulter ou modifier les informations relatives à votre tâche de transfert.
12. Choisissez ensuite Créer une tâche de transfert.

> **Important**
>
> Cliquez sur le bouton Créer une tâche de transfert pour démarrer le transfert de données, qui est facturable conformément à la page de tarification de la modernisation du AWS mainframe. Cette facturation est basée sur la quantité de données (Go) transférée telle que mesurée par la taille de l'ensemble de données.

Afficher les tâches de transfert

Pour afficher les tâches de transfert dans File Transfer, vous devez suivre ces étapes dans la console AWS Mainframe Modernization.

Pour afficher les tâches de transfert

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.

2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez transférer les fichiers de votre mainframe vers un compartiment Amazon S3.
3. Sur la page Tâches de transfert, sélectionnez le point de terminaison de transfert de données pour afficher vos tâches de transfert.
4. Pour les terminaux dotés de tâches de transfert préexistantes, celles-ci seront affichées dans la section Tâches de transfert. Vous pouvez choisir d'afficher les détails de n'importe quelle tâche de transfert dans cette liste.

Tutoriel : Démarrage avec le transfert de fichiers AWS Mainframe Modernization

AWS Mainframe Modernization File Transfer vous permet de transférer et de convertir des ensembles de données de mainframe pour des cas d'utilisation liés à la modernisation, à la migration et à l'augmentation du mainframe.

Suivez les étapes de ce didacticiel pour comprendre le fonctionnement du transfert de fichiers AWS Mainframe Modernization.

Présentation

Le transfert de fichiers comprend les éléments suivants :

1. Agent à installer sur le mainframe source.
2. Accédez aux fonctionnalités de découverte, de transfert et de conversion des ensembles de données directement depuis la console du service de gestion AWS Mainframe Modernization.

En tant qu'utilisateur, vous pouvez transférer des ensembles de données du mainframe vers votre compartiment Amazon S3.

Rubriques

- Étape 1 : transférer le package tar des fichiers binaires de l'agent AWS vers la partition logique du mainframe
- Étape 2 : Configuration de l'agent de transfert de fichiers sur le mainframe source
- Étape 3 : Création d'un point de terminaison de transfert de données
- Étape 4 : Création d'une tâche de transfert

- Étape 5 : Afficher la progression de la tâche de transfert

Étape 1 : transférer le package tar des fichiers binaires de l'agent AWS vers la partition logique du mainframe

Téléchargez les fichiers tar à partir du lien tar de l'agent M2-Agent.

Étape 2 : Configuration de l'agent de transfert de fichiers sur le mainframe source

Au cours de cette étape, vous configurez et démarrez l'agent de transfert de fichiers AWS Mainframe Modernization sur le mainframe source. L'agent est nécessaire pour faciliter les communications entre le service de transfert de fichiers et le mainframe source. Au moins un agent est requis par ordinateur central. Plusieurs agents peuvent être démarrés pour une haute disponibilité et une évolutivité améliorée.

Suivez les instructions du the section called “Configuration d'un agent de transfert de fichiers” guide pour terminer l'installation de l'agent de transfert de fichiers sur le mainframe.

Étape 3 : Création d'un point de terminaison de transfert de données

Suivez les étapes de la the section called “Création de points de terminaison de transfert de données” page pour créer un nouveau point de terminaison de transfert de données.

Étape 4 : Création d'une tâche de transfert

Suivez les étapes de the section called “Création de tâches de transfert” la page pour créer et gérer vos tâches de transfert.

Étape 5 : Afficher la progression de la tâche de transfert

Vous pouvez consulter la progression de votre tâche de transfert dans la console de modernisation du AWS mainframe. Pour plus de détails, reportez-vous à the section called “Afficher les tâches de transfert” la section.

Encodages source et cible pris en charge dans le transfert de fichiers AWS Mainframe Modernization

Le transfert de fichiers AWS Mainframe Modernization prend en charge différents types d'ensembles de données et options de conversion de pages de code.

Types d'ensembles de données du mainframe

Le transfert de fichiers AWS Mainframe Modernization prend en charge les types d'ensembles de données mainframe suivants :

- Non VSAM : séquentiel (PS), PDS, GDS, GDG
- Types de VSAM : KDS

Pages de code prises en charge

Le transfert de fichiers AWS Mainframe Modernization prend en charge les pages de code suivantes pour la conversion des ensembles de données (depuis/vers) :

« BIG5 », « BIG5 _HKSCS », « CESU_8 », « EUC_JP », « EUC_KR », « GB18 030", « », « GBK », « IBM00858 », « IBM01140 », « IBM01145 », « IBM01145 », « IBM01145 », « « IBM01146 », « IBM01147 », « IBM01148 », « IBM01149 », « IBM037 », « 026 », « 047 », « », « », « 0 », « 0 », « 0 », « 0 », « 00 », « 0 », « 0 », « 0 », « » GB2312 IBM1 IBM1 IBM273 IBM277 IBM278 IBM28 IBM284 IBM285 IBM29 IBM297 IBM42 IBM424 IBM437 IBM5 IBM775 IBM85 IBM852 IBM855"," "," IBM857 IBM86 0", "", "", "", "IBM861", "", IBM862 "", "", IBM863 "0"," IBM864 IBM865 ", « IBM_THAI »IBM866, « ISO_2022_CN », « ISO_2022_JP », « ISO_2022_JP_2", « ISO_2022_KR », « ISO_8859_1", « ISO_8859_13 », « ISO_8859_15 », « ISO_8859_16 », « ISO_8859_2 », « ISO_8859_3 », « ISO_8859_4 », « ISO_8859_5 », « ISO_8859_9 », « ISO_8859_9" IBM87 _X0201 », « JIS_X0212_1990 », « _R », « _U », IBM868 IBM869 IBM871 IBM918 KOI8 KOI8 « SHIFT_JIS », « TIS_620 », « US_ASCII », « UTF_16 », « UTF_16BE », « UTF_16LE », « UTF_32 », « UTF_32BE », « UTF_8 », « WINDOWS_1251 », « WINDOWS_1252 », « WINDOWS_1253 », « WINDOWS_1254 », « WINDOWS_1255 », « WINDOWS_1256 », « WINDOWS_1257 », « WINDOWS_1258 », « WINDOWS_31J », « X_ _HKSCS_2001 », « X_ _SOLARIS », « X_EUCJP_OPEN », « X_EUC_JP_LINUX », « X_EUC_TW », « X_ 006 », « X_ 025 », « X_ 046 », « X_ 097 », « X_ 098", « X_ », « X_ », « X_ », « X_ » BIG5 BIG5 IBM1 IBM1 IBM1 IBM1 IBM1 IBM1112 IBM1122 IBM1123 IBM1124«, « X_ IBM1129 », « X_ », « X_ IBM1166 », « X_ IBM1364 », « X_ IBM1381 », « X_ IBM29626 C IBM1383 », « X_ », « X_ », « X_ IBM3 »,

« X_ », IBM33722 « X_ », « X_ IBM737 », « X_ IBM833 », « X_ IBM834 », « X_ IBM856 », « X_ », « X_ IBM874 », « X_ IBM875 » « X_ IBM921 », « X_ », « X_ IBM922 », « X_ IBM93 », « X_C IBM933 », « X_ », « X_ IBM935 », « X_ IBM937 », « X_ IBM939 », « X_ », IBM942 « IBM942 X_C », « X_ IBM943 IBM943 IBM948 IBM949 IBM949 IBM95 0 », « X_ », « X_ 0 », « X_ IBM964 », « X_ISO_2022_CNS IBM97 », « X_ISO_2022_CN_GB », « X_ISO_8859_11 », « X_JIS0208", « X_JISAUTODETECT », « X_JOHAB », « X_MACARABIC », « X_MACCENTRALAN » EUROPE », « X_MACCROATIAN », « X_MACCYRILLIC », « X_MACDINGBAT », « X_MACGREEK », « X_MACCHEBREW », « X_MACRILLIC », « X_MACROMANIA », « X_MACTHAI », « X_MACTURKISH », « X_MACUKRAINE », « _ ISCII91 _0213 », « X_ 0_HKSCS », « X_ 0_HKSCS_XP », « X_MSWIN_936 », « X_PCK », « X_SJIS_0213 », « X_UTF_16LE_BOM », « X_UTF_32BE_BOM », « X_UTF_ MS932 MS95 MS95 32LE_BOM », « X_WINDOWS_50220 », « X_WINDOWS_50221 », « X_WINDOWS_874 », « X_WINDOWS_949 », « X_WINDOWS_950 », « X_WINDOWS_022j » ISO2

AWS Transform pour moderniser les applications mainframe

AWS Transform for mainframe vous permet de moderniser plus rapidement les applications mainframe COBOL existantes vers des applications Java en automatisant les tâches complexes et chronophages que sont l'analyse des bases de code, la planification de la transformation, la génération de documentation technique, l'extraction de la logique métier, la décomposition du code, la planification des vagues et l'exécution du refactoring et du reforgeage. Il réduit le coût et la complexité de la modernisation des mainframes en tirant parti de l'IA générative et de l'automatisation, tout en préservant votre logique métier essentielle. AWS Transform Son interface en langage naturel et son approche axée sur les objectifs vous permettent de garder le contrôle de la transformation, ce qui vous permet de vous concentrer sur les priorités stratégiques, tandis que l'automatisation prend en charge le gros du processus de modernisation.

Pour plus d'informations sur les capacités et les fonctionnalités clés, la procédure détaillée et l'implication humaine dans les entrées et le traitement, voir Migration et modernisation des applications mainframe dans le guide de l'AWS Transform utilisateur.

Principaux avantages

AWS Transform les capacités de modernisation des applications mainframe présentent de nombreux avantages. Certains d'entre eux incluent :

- Accélérez le processus de modernisation du mainframe grâce à l'IA générative : elle vous AWS Transform aide à transformer votre code COBOL en code Java moderne en quelques mois, au lieu de la chronologie traditionnelle en années.
- Comblez les lacunes en matière de connaissances : AWS Transform génère une documentation complète pour vos applications mainframe, comblant ainsi les lacunes en matière de connaissances et permettant de prendre des décisions plus éclairées.
- Préservez la logique métier critique : AWS Transform préserve la logique métier critique de vos anciens systèmes tout en les refactorisant pour les adapter à des applications Java modernes optimisées pour le cloud.
- Décomposition des domaines logiques commerciaux et techniques : décompose AWS Transform automatiquement la base de code du mainframe en domaines métiers distincts afin de réduire les efforts manuels et le temps nécessaires à l'analyse et à la décomposition de la base de code.

- Fonctionnalités Human in the Loop (HITL) : AWS Transform propose une approche autonome et axée sur les objectifs qui vous permet de garder le contrôle du processus de modernisation du mainframe.

Procédure pas à pas de la transformation de la console d'applications du mainframe

Dans l'expérience AWS Transform Web, vous pouvez effectuer la transformation de vos applications mainframe de COBOL en Java. Pour comprendre comment utiliser cette fonction, suivez toutes les étapes de la page Transformation des applications mainframe du Guide de l'AWS Transform utilisateur.

AWS Modernisation du mainframe et réplication des données avec Precisely

AWS Mainframe Modernization propose une variété d'images Amazon Machine (AMIs). Ils AMIs facilitent le provisionnement rapide des EC2 instances Amazon, créant ainsi un environnement personnalisé pour la réplication des données depuis les systèmes mainframe jusqu'à l' AWS utilisation de Precisely. Ce guide fournit les étapes nécessaires pour y accéder et les utiliser AMIs.

Prérequis

- Assurez-vous de disposer d'un accès administrateur à un AWS compte sur lequel vous pouvez créer des EC2 instances Amazon.
- Vérifiez que le service de modernisation du AWS mainframe est disponible dans la région où vous prévoyez de créer les EC2 instances Amazon. Consultez la liste des services AWS disponibles par région.
- Identifiez l'Amazon Virtual Private Cloud (Amazon VPC) dans lequel les EC2 instances Amazon seront créées.
- Lorsque vous créez EC2 des instances Amazon dans un Amazon VPC, assurez-vous que la table de routage associée possède une passerelle Internet ou une passerelle NAT.

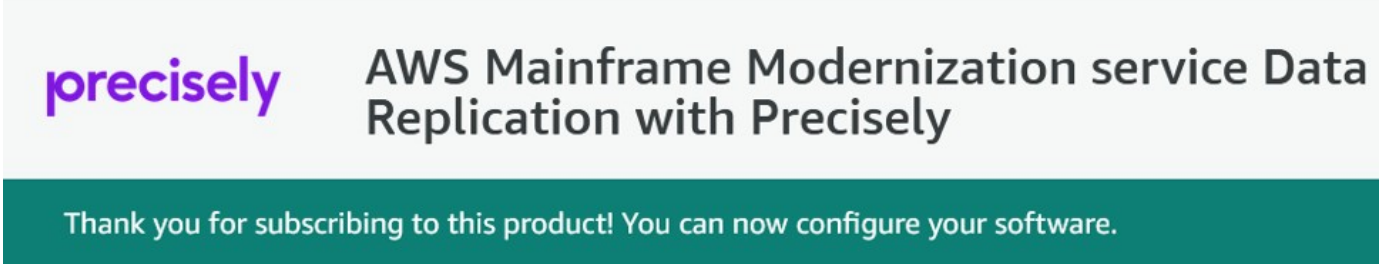
> **Note**
>
> Une réplication de données réussie nécessite que l' EC2 instance AWS dispose d'un accès de communication avec AWS Marketplace. En cas de problème de connectivité avec AWS Marketplace, le processus de réplication échouera.

Abonnez-vous à l'Amazon Machine Image

Lorsque vous vous abonnez à un produit AWS Marketplace, vous pouvez lancer une instance depuis l'AMI du produit.

1. Connectez-vous à la AWS Marketplace console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/marketket.

2. Sélectionnez Manage subscriptions (Gérer les abonnements).
3. Accédez à l'un des liens suivants en fonction de votre cas d'utilisation :
   - Réplication des données pour IBM z/OS : https://aws.amazon.com/marketplace/pp/prodview-doe2lroefogia
   - Réplication des données pour IBM i : https://aws.amazon.com/marketplace/pp/prodview-iqrkflccxf7ko
4. Choisissez Continue to Subscribe (Continuer pour s'abonner).
5. Si les termes et conditions sont acceptables, choisissez Accepter les termes. Le traitement de l'abonnement peut prendre quelques minutes.
6. Attendez que le message de remerciement apparaisse, comme indiqué ci-dessous. Ce message confirme que vous êtes bien abonné au produit.



7. Dans le volet de navigation de gauche, choisissez Gérer les abonnements. Cette vue affiche tous les abonnements auxquels vous êtes abonné.

Lancez la réplication des données de modernisation du AWS mainframe avec Precisely

1. Ouvrez la AWS Marketplace console à l'adresse https://console.aws.amazon.com/marketplace.
2. Dans le volet de navigation de gauche, choisissez Gérer les abonnements.
3. Recherchez l'AMI que vous souhaitez lancer, puis choisissez Launch new instance.
4. Sous Région, sélectionnez la région autorisée.
5. Choisissez Continuer pour lancer EC2. Cette action vous amène à la EC2 console Amazon.
6. Entrez un nom pour le serveur.
7. Sélectionnez un type d'instance qui correspond aux exigences de performance et de coût de votre projet. Le point de départ suggéré pour la taille de l'instance est `c5.2xLarge`.

8. Choisissez une paire de clés existante ou créez-en une nouvelle et enregistrez-en une nouvelle. Pour plus d'informations sur les paires de clés, consultez les paires de EC2 clés Amazon et les instances Linux dans le guide de EC2 l'utilisateur Amazon.
9. Modifiez les paramètres réseau et choisissez le VPC autorisé et le sous-réseau approprié.
10. Choisissez un groupe de sécurité existant ou créez-en un nouveau. En plus d'autoriser l'accès SSH (par défaut sur le port 22), pour la réplication des données avec une EC2 instance de serveur Precisely, il est courant d'autoriser le trafic TCP vers son port par défaut 2626.
11. Configurez le stockage pour l' EC2 instance Amazon.
12. Consultez le résumé et choisissez Launch instance. Pour que le lancement réussisse, le type d'instance doit être valide. Si le lancement échoue, choisissez Modifier la configuration de l'instance et choisissez un autre type d'instance.
13. Après avoir vu le message de réussite, choisissez Connect to instance.
14. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
15. Dans le volet de navigation de gauche, sous le menu Instances, sélectionnez Instances.
16. Dans le volet principal, vérifiez l'état de votre instance.

Créer une politique IAM

Pour exploiter avec succès les EC2 instances de modernisation du AWS mainframe déployées via notre AWS Marketplace liste, vous devez configurer un rôle et une politique IAM. Cette configuration IAM spécialement adaptée n'est pas facultative ; elle autorise vos instances EC2 Amazon à interagir avec le service. AWS Marketplace Le rôle et la politique IAM permettent à AWS Mainframe Modernisation d'enregistrer avec précision les données d'utilisation, ce qui est essentiel pour une facturation précise. L'échec de la mise en œuvre de cette configuration peut entraîner l'échec des tentatives de réplication des données et des interruptions du fonctionnement.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).
3. Si c'est la première fois que vous choisissez Politiques, la page Bienvenue dans Managed Policies apparaît. Sélectionnez Mise en route.
4. En haut de la page, sélectionnez Créer une politique.
5. Dans la section Éditeur de politique, choisissez l'option JSON.
6. Entrez la politique JSON suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": ["aws-marketplace:MeterUsage"],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Créer un rôle IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, sélectionnez Rôles, puis Créer un rôle.
3. Sous la section Type d'entité de confiance, sélectionnez Service AWS .
4. Dans la section Cas d'utilisation, sous Service ou cas d'utilisation, choisissez Amazon EC2.
5. Choisissez Suivant.
6. Dans la liste des politiques, sélectionnez Client géré dans le menu déroulant Filtrer par type et entrez le nom de la politique que vous avez créée. Cochez la case à côté du nom de la politique.
7. Choisissez Suivant.
8. Entrez un nom et, éventuellement, une description pour le rôle.
9. Passez en revue la politique de confiance et les autorisations, puis choisissez Create role.

Attachez le rôle IAM à l'instance Amazon EC2

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, choisissez Instances.
3. Sélectionnez votre EC2 instance Amazon.
4. Dans le menu Actions, choisissez Sécurité, puis Modifier le rôle IAM.
5. Sélectionnez le rôle à associer à votre instance, puis choisissez Mettre à jour le rôle IAM.

Pour plus d'informations sur la mise en route de la réplication de AWS données pour IBM i, consultez la section Présentation de la réplication de données.

AWS Mainframe Modernization Conversion de code avec mLogica

AWS Mainframe Modernization La conversion de code avec MLogica (conversion de code) est une AWS Mainframe Modernization fonctionnalité qui convertit automatiquement le code de l'assembleur du z/OS mainframe en COBOL. Vous pouvez utiliser la conversion de code pour extraire une image d'assembleur à l'aide du AWS CodeBuild service pour la conversion de code que vous souhaitez effectuer avec votre Compte AWS.

Rubriques

- Qu'est-ce que la conversion en assembleur avec mLogica ?
- Comprendre la facturation de la conversion de code pour la conversion d'un assembleur
- Concepts de conversion de code
- Comprendre les composants et les processus de conversion de code
- Tutoriel : Convertir le code d'Assembler en COBOL dans AWS Mainframe Modernization

Qu'est-ce que la conversion en assembleur avec mLogica ?

AWS Mainframe Modernization La conversion de code avec MLogica (conversion de code) convertit automatiquement le code de l'assembleur du z/OS mainframe en COBOL. Le service s'exécute dans votre environnement Compte AWS et ne transmet ni ne stocke le code source Assembler ou COBOL en dehors du Compte AWS. La conversion de code permet à votre compte autorisé d'extraire une image d'assembleur à l'aide du AWS CodeBuild service pour la conversion de code souhaitée.

AWS Mainframe Modernization vous permet de configurer des builds et des pipelines de integration/continuous livraison continue (CI/CD) pour vos applications migrées. Ces builds et pipelines utilisent AWS CodeBuild Amazon S3 pour fournir cette fonctionnalité. AWS CodeBuild est un service de génération entièrement géré qui compile votre code source, exécute des tests unitaires et produit des artefacts prêts à être déployés. Amazon S3 est un service de stockage d'objets qui offre une évolutivité, une disponibilité des données, une sécurité et des performances de pointe.

Rubriques

- Compilateurs de conversion de code
- Architecture de conversion de code

- Approche d'automatisation
- Sécurité
- Ressources supplémentaires

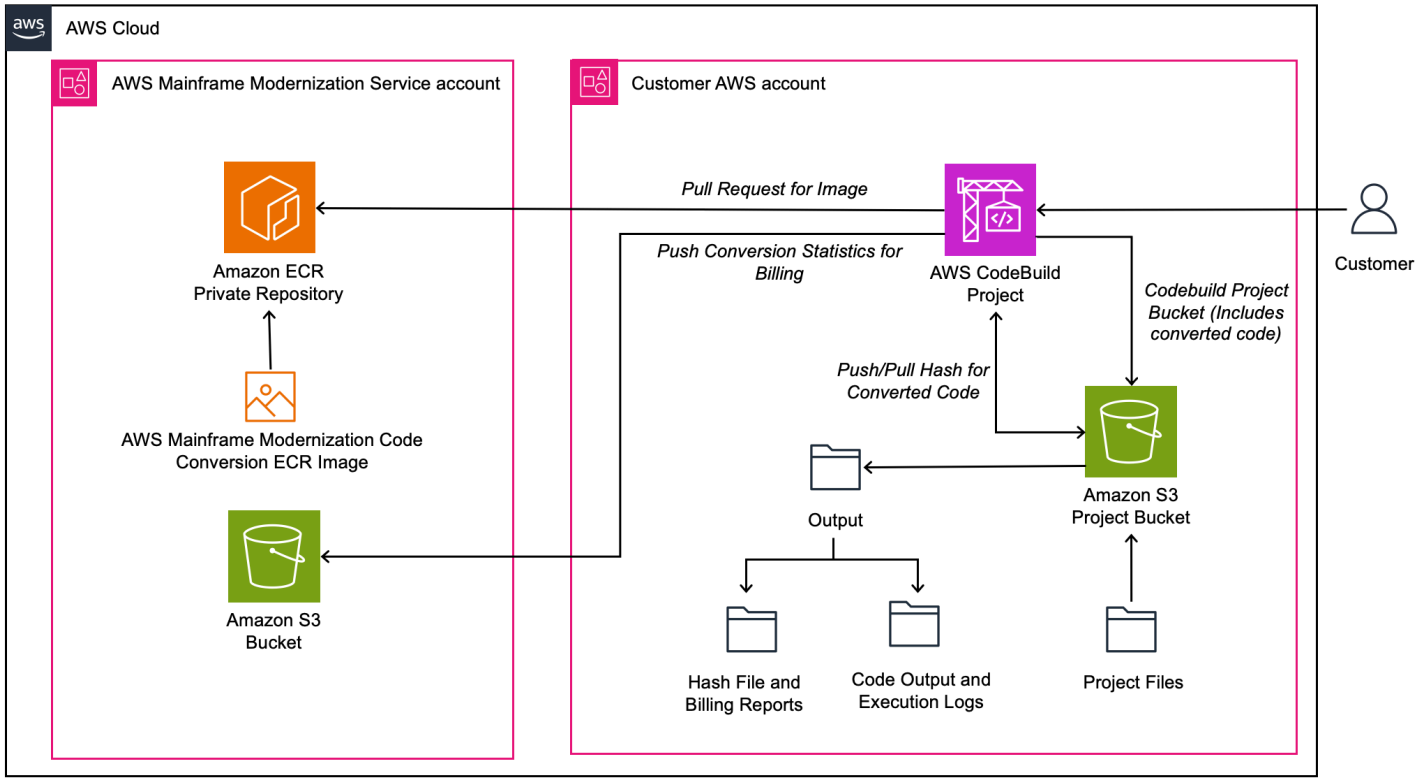
Compilateurs de conversion de code

La conversion de code peut être configurée pour émettre du COBOL adapté à la compilation et à l'exécution dans plusieurs environnements cibles avec différents compilateurs. Certains d'entre eux incluent :

- M2 Replateforme avec Rocket Software (anciennement Micro Focus) et d'autres environnements Rocket Enterprise Server
- Reconfiguration de la plateforme M2 avec NTT DATA Enterprise COBOL () UniKix
- MLogica LIBER*COBOL
- z/OSMainframe utilisant IBM Enterprise COBOL
- Iscobol Veryant

Architecture de conversion de code

Voici un schéma architectural du processus de conversion du code :



Approche d'automatisation

Pour utiliser la conversion de code avec CodeBuild, le code Assembler doit être chargé dans un compartiment Amazon S3, afin de configurer ultérieurement les paramètres de conversion et d'invoquer un CodeBuild projet pour effectuer chaque étape du processus de conversion. Le code COBOL cible est automatiquement stocké dans un chemin spécifié dans le compartiment Amazon S3.

Sécurité

AWS Mainframe Modernization La conversion de code permet la conversion tout en conservant tout le code source et cible dans votre Compte AWS. Le code source de l'assembleur, le code COBOL cible et les fichiers de configuration sont stockés dans votre compartiment Amazon S3. L'outil de conversion automatique fonctionne comme un conteneur dans l' CodeBuild environnement de votre Compte AWS. Le code reste dans votre compte à tout moment.

Pour permettre à l'outil de conversion d'accéder à votre compartiment Amazon S3, vous accordez des autorisations au compartiment à un Service AWS rôle. Lors de la configuration CodeBuild, vous définissez ce rôle de service afin de CodeBuild pouvoir accéder à l'image du conteneur et à votre compartiment Amazon S3.

Ressources supplémentaires

En plus de celathe section called "Tutoriel : Convertir le code d'Assembler en COBOL", voici quelques ressources supplémentaires où vous pouvez en savoir plus sur la création des AWS CloudFormation modèles et d'autres informations sur la conversion d'Assembler en COBOL.

- Lien vers l'atelier pour la conversion automatique du code d'Assembler en COBOL :https://catalog.workshops.aws/awsm2ccm-assembler-cobol/en-US.
- Article de blog :https://aws.amazon.com/blogs/migration-and-modernization/unlocking-new-potential-transform-your-assembler-programs-to-cobol-with-aws-mainframe-modernization/.

Comprendre la facturation de la conversion de code pour la conversion d'un assembleur

Vous allez consulter cette page pour comprendre le champ d'application et le processus de facturation liés à la conversion du code avant de procéder à la conversion proprement dite. La section de calcul de la facturation mentionne le processus par lequel la conversion d'Assembler en COBOL est facturée pour chaque ligne de code.

Conversion de code, facturation et champ d'application

La conversion du code assembleur génère des frais (rapports de facturation) sur votre compte Compte AWS uniquement une fois l'étape de conversion terminée. Les frais sont basés sur le nombre de lignes de code converties. Si vous effectuez plusieurs étapes de conversion, par exemple après avoir ajouté un nouveau code Assembler, modifié la configuration de conversion ou appliqué une nouvelle version du conteneur, seules les lignes modifiées et/ou les lignes nouvellement ajoutées sont utilisées pour calculer les frais. Nous ne vous facturerons pas deux fois pour la conversion de la même ligne de code dans le même programme.

> **Note**
>
> Les modules dont les lignes de code ont été modifiées et toutes les lignes de code des programmes nouveaux ou renommés seront facturés.

Pour éviter plusieurs frais, la conversion de code stocke un fichier binaire codé pour chaque module Assembler ou Macro dans le compartiment du projet dans`<Project_bucket>/awsm2ccm-do-`

not-delete/<AWS_account_number>/Hash. Ces fichiers codés ne contiennent aucun code client.

> **Important**
>
> Ne modifiez ou ne supprimez pas ces fichiers manuellement. Les modifications peuvent entraîner plusieurs facturations pour la conversion des mêmes composants.

Le rapport d'analyse de la conversion du AWS Mainframe Modernization code (« Rapport d'analyse ») fournit aux clients des détails sur l'étendue de la conversion prévue, le résultat et la facturation afin de garantir des attentes précises quant à la conversion réelle. La conversion peut entraîner la non-conversion de certaines lignes de code, la conversion partielle de certaines lignes de code et la conversion complète de certaines lignes de code. Le rapport d'analyse indique le nombre de lignes de code pour chaque catégorie. Vous devez exécuter et lire le rapport d'analyse avant de procéder à toute conversion de programmes, de macros et de cahiers. Une fois qu'un client a lu le rapport d'analyse et qu'il est d'accord avec le champ d'application indiqué, le résultat attendu et la facturation prévue, il peut poursuivre l'exécution de la conversion.

> **Note**
>
> En exécutant la **`Convert`** commande de conversion du code de modernisation du mainframe AWS, vous reconnaissez avoir exécuté et lu le rapport d'analyse, et vous acceptez le résultat attendu et le nombre facturable de lignes de code.

Étendue de la conversion

AWS Mainframe Modernization La conversion de code traite toutes les lignes de code de tous les composants assembleur, macro et copybook disponibles dans les répertoires scrlib et macrolib de l'emplacement source S3 configuré. Les programmes d'assemblage, ainsi que toutes les macros et tous les copybooks référencés dans un programme d'assemblage, sont concernés. Les composants de macro et de cahier qui ne sont pas référencés par un programme assembleur sont considérés comme hors de portée et ne sont pas convertis. Pendant le traitement, le convertisseur exécute des algorithmes avancés qui prennent en compte chaque composant du champ d'application de manière globale. Toutes les lignes de code de ces composants participent au traitement, qu'elles soient totalement, partiellement converties ou non converties. AWS Mainframe Modernization La conversion de code ignore les lignes vides et ne les considère pas comme des lignes de code. Les

lignes de commentaires et les lignes contenant tout autre texte (par exemple, les instructions JCL pour l'assembleur intégré à JCL) sont considérées comme des lignes de code pour la facturation.

Calcul de facturation

AWS Mainframe Modernization Frais de conversion de code pour l'ensemble des composants concernés. Cela signifie qu'il facture chaque ligne de code de chaque composant concerné, y compris les lignes qui n'ont pas pu être converties, qui ont été partiellement converties et qui ont été totalement converties. AWS Mainframe Modernization La conversion de code additionne toutes les lignes de code des composants fournis pour le traitement (y compris les programmes d'assemblage, les cahiers référencés et les macros référencées) et utilise le nombre total de lignes de code pour la facturation.

> **Note**
>
> Les copybooks et les macros non référencés par un programme Assembler ne sont pas considérés comme concernés.

Supposons par exemple qu'un programme comporte 1 000 lignes de code :

- 700 lignes sont entièrement converties
- 200 lignes sont partiellement converties
- 100 lignes ne sont pas converties

1 000 lignes de code seront traitées et seront facturables.

Améliorer la conversion

Si, en tant que client, vous recherchez un taux de conversion plus élevé pour les lignes de code ou si vous avez d'autres exigences spécifiques, vous pouvez contacter les AWS représentants pour obtenir des options d'engagement supplémentaires, telles qu'un effort de calibrage ou une assistance de services professionnels.

Concepts de conversion de code

Pour savoir comment se produit la conversion de code, il CodeBuild est important de comprendre certains concepts clés tels que la gestion des macros, les pages de code, etc.

Rubriques

- Gestion des macros
- Pages de codes (EBCDIC ou ASCII)
- CodeBuild

Gestion des macros

Le code Mainframe Assembler utilise fréquemment des macros pour encapsuler les fonctionnalités en vue de leur réutilisation. Le comportement des macros est généralement déterminé lors de l'exécution de l'application en fonction des paramètres transmis par un programme Assembler. La conversion de code fournit plusieurs mécanismes pour étendre les macros d'assemblage avant la conversion en COBOL.

Pages de codes (EBCDIC ou ASCII)

Mainframe Assembler contient souvent des littéraux de caractères exprimés sous forme de valeurs hexadécimales correspondant aux caractères EBCDIC. La conversion de code fournit une fonctionnalité configurable permettant de gérer automatiquement les littéraux de caractères en ASCII lors de l'émission de COBOL pour les environnements ASCII.

CodeBuild

La conversion de code est disponible via le AWS CodeBuild service. AWS CodeBuild est un outil d'automatisation de construction conçu à l'origine dans le cadre d'un CI/CD pipeline. In AWS Mainframe Modernization, AWS CodeBuild est utilisé pour automatiser l'outil de conversion MCCAC et d'autres outils tels que le compilateur COBOL Rocket Software (anciennement Micro Focus).

Comprendre les composants et les processus de conversion de code

AWS Mainframe Modernization Le processus de conversion de code inclut divers composants tels que le AWS Mainframe Modernization conteneur, le compartiment de projet S3 et l'emplacement des fichiers journaux.

Rubriques

- AWS Mainframe Modernization contenant

- compartiment de projet S3
- Emplacement des fichiers journaux
- Présentation du processus

AWS Mainframe Modernization contenant

AWS Mainframe Modernization Le conteneur de conversion de code s'exécute dans le AWS CodeBuild projet et fournit des commandes pour configurer les répertoires du projet et les fichiers de configuration, évaluer le code Assembler, développer les macros Assembler et convertir le code Assembler en COBOL.

Vous aurez accès au référentiel AWS ECR suivant : `381492161314.dkr.ecr.us-east-1.amazonaws.com/aws-mlogica-codebuild-prod`

Pour utiliser les images, vous pouvez suivre l'une des deux options suivantes :

- Utilisez la dernière balise lorsque vous consommez l'image via AWS CodeBuild. Lorsque vous utilisez l'image, vous utiliserez ce chemin :`381492161314.dkr.ecr.us-east-1.amazonaws.com/aws-mlogica-codebuild-prod`. Cela signifie que cela AWS CodeBuild récupérera la dernière image envoyée dans le référentiel.
- Lister la version et sélectionner parmi celles-ci. Pour ce faire, utilisez la commande suivante via la CLI pour répertorier les différentes versions du référentiel :

```
aws ecr describe-images \
  --registry-id 381492161314 \
  --repository-name aws-mlogica-codebuild-prod \
  --query 'imageDetails[*].{ImagePushedAt: imagePushedAt, ImageTags: imageTags}' \
  --output json | jq '[.[] | {ImageURI: (.ImageTags[] |
 "381492161314.dkr.ecr.us-east-1.amazonaws.com/aws-mlogica-codebuild-prod:" + .),
 ImagePushedAt: .ImagePushedAt}] | sort_by(.ImagePushedAt) | reverse'
```

Cela listera toutes les images avec le tag associé sur chaque image, ainsi que l'heure à laquelle une image particulière a été publiée dans le référentiel. Sur la base du code ci-dessus, vous obtiendrez une liste d'images où la balise sur l'image représente la version de l'utilitaire de conversion de code. Vous pouvez sélectionner l'image appropriée en fonction de vos besoins.

compartiment de projet S3

Le code d'entrée et de sortie, le code mis à jour avec des macros étendues et les rapports générés par la conversion de AWS Mainframe Modernization code sont stockés dans le bucket de projet que vous créez dans votre Gestion de compte AWS. Vous AWS Mainframe Modernization permettez à Code Conversion d'accéder au bucket en accordant des autorisations à un rôle AWS de service.

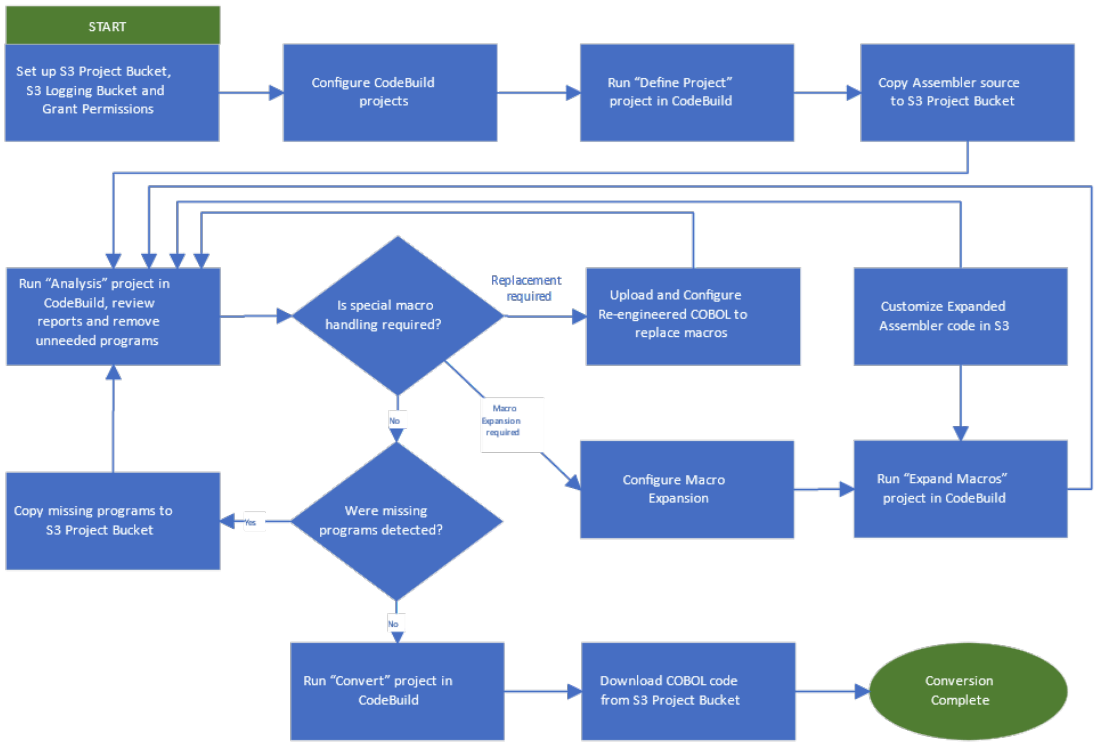
Emplacement des fichiers journaux

Les fichiers journaux sont écrits à deux endroits lors de l'exécution de chaque CodeBuild projet :

- Les fichiers journaux contenant les résultats de haut niveau de chaque CodeBuild étape sont écrits dans les fichiers journaux du compartiment de journalisation configuré dans le CodeBuild. Ces fichiers apparaissent sous forme d'archives gzip avec un nom de fichier de type GUID généré par le CodeBuild framework (par exemple,`0c03e183-ab40-4fe0-ba77- bc1d87e73b14.gz`). Chaque archive contient le journal généré par l'exécution d'un CodeBuild projet. Si l'exécution d'un CodeBuild projet échoue, ce fichier journal contiendra des informations de dépannage importantes.
- Les fichiers journaux contenant les résultats d'exécution détaillés au niveau du composant sont écrits dans les fichiers journaux situés dans le chemin du compartiment principal du projet avec le modèle de nom de fichier `<Project_Bucket_name>_.log` (par exemple`project-bucket_202406131200.log`). Ces journaux fournissent :
  - Un résumé de configuration indiquant les emplacements d'entrée et de sortie.
  - Un journal de chaque composant Assembler ou Macro traité avec le nom de fichier cible.
  - Liste des rapports générés avec l'emplacement des fichiers.
  - Pour les exécutions de conversion, liste des cahiers d'exécution fournis.

Présentation du processus

Le schéma suivant illustre le processus de conversion d'Assembler en COBOL :



Tutoriel : Convertir le code d'Assembler en COBOL dans AWS Mainframe Modernization

Vous pouvez utiliser ce document comme step-by-step guide pour comprendre comment convertir le code Assembler de modernisation du mainframe en COBOL. En outre, vous pouvez également consulter l'atelier sur la conversion automatique de code d'Assembler en COBOL pour en savoir plus sur le processus de conversion.

Rubriques

- Prérequis
- Étape 1 : Partagez les actifs de construction avec Compte AWS
- Étape 2 : créer des compartiments Amazon S3
- Étape 3 : Création d'une politique IAM
- Étape 4 : Création d'un rôle IAM
- Étape 5 : associer la politique IAM au rôle IAM
- Étape 6 : Création du CodeBuild projet
  - Étape 6.1 : Création du projet Define

  - Étape 6.2 : Création du projet d'analyse de code
  - Étape 6.3 : Création du projet de conversion de code
- Étape 7 : Définition du projet et téléchargement du code source
- Étape 8 : Exécuter l'analyse et comprendre les rapports
- Étape 9 : Exécuter la conversion du code
- Étape 10 : Vérifiez la conversion du code
- Étape 11 : Téléchargez le code converti
- Nettoyage des ressources

Prérequis

Lisez la Comprendre la facturation de la conversion de code pour la conversion d'un assembleur section pour comprendre comment la conversion du code Assembler génère des frais (rapports de facturation) sur votre Gestion de compte AWS compte et comment fonctionne la facturation.

Étape 1 : Partagez les actifs de construction avec Compte AWS

Au cours de cette étape, assurez-vous de partager les actifs de construction avec vous Compte AWS, en particulier dans la région où les actifs sont utilisés.

1. Ouvrez la AWS Mainframe Modernization console à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le volet de navigation de gauche, sélectionnez Outils.
3. Dans Conversion du code de modernisation du mainframe AWS avec mLogica, choisissez Share assets with my. Compte AWS

> **Important**
>
> Vous devez effectuer cette étape une fois dans chaque AWS région où vous avez l'intention d'effectuer des builds.

Étape 2 : créer des compartiments Amazon S3

Au cours de cette étape, vous créez des compartiments Amazon S3. Le premier compartiment est le compartiment de projet destiné AWS CodeBuild à contenir le code source, puis à pousser le compartiment de sortie pour contenir la AWS CodeBuild sortie (code converti). Pour plus d'informations, consultez la section Création, configuration et utilisation des compartiments Amazon S3 dans le guide de l'utilisateur Amazon S3.

1. Pour créer le compartiment de projet, connectez-vous à la console Amazon S3 et choisissez Create bucket.
2. Dans Configuration générale, donnez un nom au compartiment et spécifiez l' Région AWS endroit où vous souhaitez le créer. Un exemple de nom est*codebuild-regionId-accountId-bucket*, où :

   - `regionId`est le Région AWS produit du seau.
   - `accountId`est votre Compte AWS identifiant.

   > **Note**
   >
   > Si vous créez le bucket dans un pays différent Région AWS de celui de l'est des États-Unis (Virginie du Nord), spécifiez le `LocationConstraint` paramètre. Pour plus d'informations, consultez Create Bucket dans le manuel Amazon Simple Storage Service API Reference.

3. Conservez tous les autres paramètres, puis choisissez Create bucket.

Quels que soient les noms que vous choisissez pour ces compartiments, veillez à les utiliser tout au long de ce didacticiel.

Étape 3 : Création d'une politique IAM

Au cours de cette étape, vous allez créer une politique IAM. La politique IAM fournie accorde des autorisations spécifiques AWS CodeBuild pour interagir avec Amazon S3, Amazon Elastic Container Registry, les CloudWatch journaux Amazon qui les CodeBuild génèrent et les Amazon Elastic Compute Cloud ressources pour la conversion de code. Cette politique n'est pas personnalisée pour les clients. La politique accorde des autorisations AWS Mainframe Modernization pour interagir et récupérer les statistiques de conversion du code afin de facturer le client de manière appropriée.

Pour en savoir plus sur la création d'une stratégie IAM, consultez la section Création de politiques IAM dans le guide de l'utilisateur IAM.

Pour créer une politique

1. Connectez-vous à la console IAM, puis sélectionnez Policies dans le volet de navigation de gauche.
2. Choisissez Create Policy (Créer une politique).
3. Copiez et collez la politique JSON suivante dans l'éditeur de politiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:PutObjectAcl",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::codebuild-regionId-accountId-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-bucket/*",
                "arn:aws:s3:::aws-m2-repo-*" ],
                    "Effect": "Allow"
          },
          {
              "Action": [
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "logs:*",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DeleteNetworkInterface",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeVpcs",
```

```
                "ec2:CreateNetworkInterfacePermission"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

4. Vous pouvez éventuellement ajouter des balises à la politique. Les balises sont des paires clé-valeur qui peuvent vous aider à organiser, suivre ou contrôler l'accès à la politique.
5. Choisissez Next: Review (Suivant : vérifier).
6. Donnez un nom à la politique, par exemple,*`CodeBuildAWSM2CCMPolicy`*.
7. Vous pouvez éventuellement saisir une description de la politique et consulter le résumé de la politique pour vous assurer qu'il est correct.
8. Choisissez Create Policy (Créer une politique).

Étape 4 : Création d'un rôle IAM

Au cours de cette étape, vous créez un nouveau rôle IAM qui permet d'interagir avec les AWS ressources CodeBuild à votre place, après avoir associé les politiques IAM que vous avez créées précédemment à ce nouveau rôle IAM.

Pour plus d'informations sur la création d'un rôle de service, consultez la section Création d'un rôle pour déléguer des autorisations à un AWS service dans le guide de l'utilisateur IAM.

1. Connectez-vous à la console IAM, puis sélectionnez Rôles dans le volet de navigation de gauche.
2. Choisissez Créer un rôle.
3. Sous Type d'entité fiable, choisissez le service AWS.
4. Sous Cas d'utilisation pour d'autres services AWS CodeBuild, choisissez, puis choisissez CodeBuildà nouveau.
5. Choisissez Suivant.
6. Sur la page Add permissions (Ajouter des autorisations), sélectionnez Next (Suivant). Vous attribuez une politique au rôle ultérieurement.
7. Sous Détails du rôle, saisissez un nom pour le rôle, par exemple,`IAMRoleTaskExecutionRoleForCodeBuild`.

8. Sous Sélectionner les entités de confiance, vérifiez que le document de politique ressemble à ce qui suit :

```
{
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "codebuild.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
}
```

9. Choisissez Créer un rôle.

Étape 5 : associer la politique IAM au rôle IAM

Au cours de cette étape, vous associez la politique IAM que vous avez créée précédemment au rôle `IAMRoleTaskExecutionRoleForCodeBuild` IAM.

1. Connectez-vous à la console IAM, puis sélectionnez Rôles dans le volet de navigation de gauche.
2. Dans Rôles, choisissez le rôle que vous avez créé précédemment, par exemple`IAMRoleTaskExecutionRoleForCodeBuild`.
3. Dans Politiques d'autorisations, choisissez Ajouter des autorisations, puis Joindre des politiques.
4. Dans Autres politiques d'autorisation, choisissez les politiques que vous avez créées précédemment, par exemple*`CodeBuildAWSM2CCMPolicy`*.
5. Choisissez Attach Policies (Attacher des politiques).

Étape 6 : Création du CodeBuild projet

Au cours de cette étape, vous créez trois CodeBuild projets différents sur la base du `buildspec.yml` fichier mentionné ci-dessus.

Étape 6.1 : Création du projet Define

Pour créer le projet Define

1. Connectez-vous à la CodeBuild console et choisissez Create build project.
2. Dans la section Configuration du projet, donnez un nom au projet, par exemple,`1-awsm2ccm-define-project`.
3. Dans la section Source, pour Source provider, laissez la sélection par défaut.
4. Dans la section Environnement, choisissez Image personnalisée.
5. Dans le champ Type d'environnement, sélectionnez Linux.
6. Sous Registre d'images, choisissez Autre registre.
7. Dans le champ URL du registre externe, suivez la the section called "AWS Mainframe Modernization contenant" section.
8. Sous Rôle de service, choisissez Rôle de service existant, puis dans le champ ARN du rôle, choisissez le rôle de service que vous avez créé précédemment (par exemple,`IAMRoleTaskExecutionRoleForCodeBuild`).
9. Développez la section Configuration supplémentaire, procédez comme suit :
   a. VPC : configurez si nécessaire en fonction de votre configuration.
   b. Délai d'attente : réglé sur 60 minutes.
   c. Délai d'attente : défini sur 480 minutes.
   d. Chiffrement : Choisissez les paramètres de chiffrement appropriés (la valeur par défaut est correcte).
   e. Dans la section Variables d'environnement, ajoutez les éléments suivants un par un :
      - Nom : PROJECT_BUCKET. Valeur :**`codebuild-regionId-accountId- bucket`**. Type : texte brut
      - Nom : PROJECT_DIR. Valeur :**`prj_codebuild_01`**. Type : texte brut
      - Nom : AWSM2CCM_ACTION. Valeur :**`define_project`**. Type : texte brut
      - Nom : AWSM2CCM_LOGGING_BUCKET. Valeur :**`s3:// codebuild-regionId-accountId-bucket`**. Type : texte brut
10. Dans la section Buildspec, choisissez Insérer des commandes de construction, puis Basculer vers l'éditeur.
11. Remplacez les valeurs actuelles par les suivantes :

```
version: 0.2
phases:
    build:
        commands:
            - . /app/awsm2ccm_prod/bin/setup_env.sh
            - run_awsm2ccm.sh $PROJECT_DIR
artifacts:
    files:
        - '**/*'
    discard-paths: no
    base-directory: $PROJECT_DIR
```

où PROJECT_DIR sont des variables d'environnement disponibles à l'intérieur. CodeBuild Pour plus d'informations, consultez la section Variables d'environnement dans les environnements de génération.

12. Dans la section Artefacts, procédez comme suit :

    - sous Type, choisissez Amazon S3, puis choisissez votre compartiment de sortie, par exemple,`codebuild-regionId-accountId-bucket`.
    - pour Path, laissez ce champ vide.
    - pour Nom, entrez**`prj_codebuild_01`**.
    - pour l'emballage d'Artifact, sélectionnez Aucun.
    - pour Remplacer le nom de l'artefact, décochez cette option.
    - pour le chiffrement, laissez les paramètres par défaut.

13. Pour la section Logs, procédez comme suit :

    - CloudWatch journaux : Désactivé
    - Journaux S3 : activé

    - Compartiment: **`codebuild-regionId-account-bucket`**
    - Chemin du journal : **`CODEBUILD-LOGS`**

14. Choisissez Créer un projet de génération.

Étape 6.2 : Création du projet d'analyse de code

Pour créer le projet d'analyse de code

1. Connectez-vous à la CodeBuild console et choisissez Create build project.
2. Dans la section Configuration du projet, donnez un nom au projet, par exemple,`2-awsm2ccm-analysis`.
3. Dans la section Source, pour Source provider, choisissez Amazon S3, puis choisissez le bucket d'entrée que vous avez créé précédemment (par exemple,`codebuild-regionId-accountId-bucket`).
4. Dans le champ clé d'objet S3 ou dossier S3, entrez**`prj_codebuild_01`**.
5. Dans la section Environnement, choisissez Image personnalisée.
6. Dans le champ Type d'environnement, sélectionnez Linux.
7. Sous Registre d'images, choisissez Autre registre.
8. Dans le champ URL du registre externe, suivez la the section called "AWS Mainframe Modernization contenant" section.
9. Sous Rôle de service, choisissez Rôle de service existant, puis dans le champ ARN du rôle, choisissez le rôle de service que vous avez créé précédemment (par exemple,`IAMRoleTaskExecutionRoleForCodeBuild`).
10. Développez la section Configuration supplémentaire, procédez comme suit :
    a. VPC : configurez si nécessaire en fonction de votre configuration.
    b. Délai d'attente : réglé sur 60 minutes.
    c. Délai d'attente : défini sur 480 minutes.
    d. Chiffrement : Choisissez les paramètres de chiffrement appropriés (la valeur par défaut est correcte).
    e. Dans la section Variables d'environnement, ajoutez les éléments suivants un par un :
       - Nom : PROJECT_BUCKET. Valeur :**`codebuild-regionId-accountId-bucket`**. Type : texte brut
       - Nom : PROJECT_DIR. Valeur :**`prj_codebuild_01`**. Type : texte brut
       - Nom : AWSM2CCM_ACTION. Valeur :**`analysis`**. Type : texte brut
       - Nom : AWSM2CCM_LOGGING_BUCKET. Valeur :**`s3:// codebuild-regionId-accountId-bucket`**. Type : texte brut

11. Dans la section Buildspec, choisissez Insérer des commandes de construction, puis Basculer vers l'éditeur.
12. Remplacez les valeurs actuelles par les suivantes :

```
version: 0.2
phases:
    build:
        commands:
            - ln -s $CODEBUILD_SRC_DIR $PROJECT_DIR
            - . /app/awsm2ccm_prod/bin/setup_env.sh
            - run_awsm2ccm.sh $PROJECT_DIR
artifacts:
    files:
        - '*.log'
        - '_Converted/*/*'
        - '_Reports/*'
    secondary-artifacts:
        reports:
            files:
                - '_Reports/AWSM2CCM*'
    discard-paths: no
    base-directory: $PROJECT_DIR
```

où PROJECT_DIR sont des variables d'environnement disponibles à l'intérieur. CodeBuild Pour plus d'informations, consultez la section Variables d'environnement dans les environnements de génération.

13. Dans la section Artefacts, procédez comme suit :
    - sous Type, choisissez Amazon S3, puis choisissez votre compartiment de sortie (par exemple,`codebuild-regionId-accountId-bucket`).
    - dans le champ Path, saisissez ARTIFACTS.
    - pour Nom, entrez**`prj_codebuild_01`**.
    - pour l'emballage d'Artifact, sélectionnez Aucun.
    - pour Remplacer le nom de l'artefact, décochez cette option.
    - pour le chiffrement, laissez les paramètres par défaut.
14. Pour la section Logs, procédez comme suit :
    - CloudWatch journaux : Désactivé

- Journaux S3 : activé

- Compartiment: **`codebuild-regionId-account-bucket`**
- Chemin du journal : **`CODEBUILD-LOGS`**

15. Choisissez Créer un projet de génération.

Étape 6.3 : Création du projet de conversion de code

Pour créer le projet de conversion de code

1. Connectez-vous à la CodeBuild console et choisissez Create build project.
2. Dans la section Configuration du projet, saisissez un nom pour le projet (par exemple,`3-awsm2ccm-convert`).
3. Dans la section Source, pour Source provider, choisissez Amazon S3, puis choisissez le bucket d'entrée que vous avez créé précédemment (par exemple,`codebuild-regionId-accountId-bucket`).
4. Dans le champ clé d'objet S3 ou dossier S3, entrez**`prj_codebuild_01`**.
5. Dans la section Environnement, choisissez Image personnalisée.
6. Dans le champ Type d'environnement, sélectionnez Linux.
7. Sous Registre d'images, choisissez Autre registre.
8. Dans le champ URL du registre externe, suivez la the section called "AWS Mainframe Modernization contenant" section.
9. Sous Rôle de service, choisissez Rôle de service existant, puis dans le champ ARN du rôle, choisissez le rôle de service que vous avez créé précédemment, par exemple,`IAMRoleTaskExecutionRoleForCodeBuild`.
10. Développez la section Configuration supplémentaire, procédez comme suit :

    a. VPC : configurez si nécessaire en fonction de votre configuration.

    b. Délai d'attente : réglé sur 60 minutes.

    c. Délai d'attente : défini sur 480 minutes.

    d. Chiffrement : Choisissez les paramètres de chiffrement appropriés (la valeur par défaut est correcte).

    e. Dans la section Variables d'environnement, ajoutez les éléments suivants un par un :

- Nom : PROJECT_BUCKET. Valeur :**`codebuild-regionId-accountId-bucket`**. Type : texte brut
- Nom : PROJECT_DIR. Valeur :**`prj_codebuild_01`**. Type : texte brut
- Nom : AWSM2CCM_ACTION. Valeur :**`conversion`**. Type : texte brut
- Nom : AWSM2CCM_LOGGING_BUCKET. Valeur :**`s3:// codebuild-regionId-accountId-bucket`**. Type : texte brut

11. Dans la section Buildspec, choisissez Insérer des commandes de construction, puis Basculer vers l'éditeur.
12. Remplacez les valeurs actuelles par les suivantes :

```
version: 0.2
phases:
    build:
        commands:
            - export AWSM2CCM_PUSH_RUNTIME_COPYBOOKS=y
            - ln -s $CODEBUILD_SRC_DIR $PROJECT_DIR
            - . /app/awsm2ccm_prod/bin/setup_env.sh
            - run_awsm2ccm.sh $PROJECT_DIR
artifacts:
    files:
        - '*.log'
        - '_Converted/*/*'
        - '_Reports/*'
    discard-paths: no
    base-directory: $PROJECT_DIR
```

où PROJECT_DIR sont des variables d'environnement disponibles à l'intérieur. CodeBuild Pour plus d'informations, consultez la section Variables d'environnement dans les environnements de génération.

13. Dans la section Artefacts, procédez comme suit :

- sous Type, choisissez Amazon S3, puis choisissez votre compartiment de sortie (par exemple,`codebuild-regionId-accountId-bucket`).
- dans le champ Path, saisissez ARTIFACTS.
- pour Nom, entrez**`prj_codebuild_01`**.
- pour l'emballage d'Artifact, sélectionnez Aucun.

- pour Remplacer le nom de l'artefact, décochez cette option.
- pour le chiffrement, laissez les paramètres par défaut.

14. Pour la section Logs, procédez comme suit :

    - CloudWatch journaux : Désactivé
    - Journaux S3 : activé
    - Compartiment: **`codebuild-regionId-account-bucket`**
    - Chemin du journal : **`CODEBUILD-LOGS`**

15. Choisissez Créer un projet de génération.

Étape 7 : Définition du projet et téléchargement du code source

Le Define Project définit le dossier du projet et les fichiers de configuration, initialisés avec les configurations par défaut. Au cours de cette étape, vous lancez la construction. Pour cela :

1. Connectez-vous à la AWS CodeBuild console.
2. Dans le volet de navigation de gauche, choisissez Créer des projets.
3. Sélectionnez le projet créé précédemment (`1-awsm2ccm-define-project`) à construire
4. Choisissez Start build, puis Start now pour définir le projet. Une fois la construction démarrée, le statut passe à En cours.
5. Choisissez les détails de la phase pour voir la progression de chaque étape orchestrée par le AWS CodeBuild projet.
6. Attendez que le statut ait changé pour réussir toutes les étapes.
7. Accédez à la console Amazon S3.
8. Localisez et cliquez sur le compartiment Amazon S3 nommé `codebuild-regionId-accountId-bucket`

    - **`CODEBUILD-LOGS/`**le dossier contient les AWS CodeBuild journaux des AWS CodeBuild projets en cours d'exécution.
    - **`prj_codebuild_01/`**dossier contenant la structure du projet. Il est utilisé lors des étapes d'analyse, d'expand_macros et de conversion. Vous pouvez choisir `prj_codebuild_01/` d'explorer les détails

- **`cobol_reserved.rsw`**fichier de configuration (liste de mots COBOL) réservé au convertisseur. Il est utilisé lors de l'étape de conversion.
- **`Macro_Expansion/`**le dossier contient des macros à étendre aux programmes Assembler. Il est utilisé lors de l'étape expand_macros.
- **`macro_settings.json`**le fichier de configuration contient le remplacement personnalisé des macros. Il est utilisé lors de l'étape expand_macros.
- **`macrolib/`**le dossier contient les macros Assembler à convertir. Il est utilisé lors de l'étape d'analyse et de conversion.
  1. Sélectionnez `macrolib/`.
  2. Par défaut, une macro Assembler nommée `MACRO1.mac` est fournie sous forme de fichier d'exemple. Supprimez ce fichier car il n'est pas nécessaire pour l'analyse.
  3. Téléchargez vos macros dans ce répertoire.
- **`project_settings_aux.json`**le fichier de configuration contient les paramètres liés à la page de code. Il est utilisé lors de l'étape de conversion.
- **`project_settings.json`**le fichier de configuration contient les paramètres du convertisseur. Il est utilisé lors de l'étape de conversion.
- **`srclib/`**le dossier contient les programmes Assembler à convertir. Il est utilisé lors de l'étape d'analyse et de conversion.
  1. Sélectionnez `srclib/`.
  2. Par défaut, deux programmes Assembler sont nommés `SQtest01.asm` et `SQtest02.asm` sont fournis en tant qu'exemples. Supprimez ces fichiers car ils ne sont pas nécessaires à votre analyse et à votre conversion.
  3. Téléchargez vos programmes Assembler dans ce répertoire.

9. Vérifiez le statut de l'`1-awsm2ccm-define-project`étape. Cela aurait dû réussir sous l'onglet État de la dernière version.

Vous êtes prêt pour l'étape suivante : l'analyse du code.

Étape 8 : Exécuter l'analyse et comprendre les rapports

> **Note**
>
> AWS Mainframe Modernization L'étape d'analyse de la conversion du code est gratuite.

Au cours de cette étape, vous lancez un autre build :

1. Dans le volet de navigation de gauche, choisissez Créer des projets.
2. Choisissez le projet que vous avez créé à l'étape 6.2 pour créer :`2-awsm2ccm-analysis`.
3. Choisissez Start build, puis Start now pour générer des rapports d'analyse. Cela permettra de démarrer la construction et de passer à l'état en cours.
4. Choisissez les détails de la phase où vous pourrez voir la progression de chaque étape orchestrée par le AWS CodeBuild projet. Attendez que le changement de statut soit terminé pour que toutes les étapes soient terminées.
5. À partir du AWS Management Console, accédez à la console de service Amazon S3.
6. Localisez et cliquez sur le compartiment Amazon S3 : `codebuild-regionId-accountId-bucket`

    a. **`ARTIFACTS/`**le dossier contient les résultats des étapes d'analyse et de conversion.

    b. Sélectionnez `ARTIFACTS/prj_codebuild_01/_Reports/`.

    c. Les rapports suivants seront disponibles :

    - `AWSM2CCM-Analysis-Report-<timestamp>.pdf`est un rapport exécutif qui fournit la facturation et le champ d'application de la conversion du AWS Mainframe Modernization Code, l'amélioration de la conversion, le résumé des conversions et les statistiques de conversion détaillées. Il résume également le nombre de codes et le nombre de codes facturables au niveau du projet et fournit des mesures et des listes de membres référencés pour chaque composant. Il est essentiel d'exécuter et d'examiner ce rapport avant d'exécuter la conversion proprement dite.
    - `Conversion_Detailed_Statistics.txt`fournit la fréquence et le résultat de conversion attendu (affiché sous la forme « État de conversion ») pour chaque instruction trouvée dans chaque composant. Cela fournit un moyen rapide de déterminer si les instructions sont claires et que le convertisseur ne les prend pas en charge. Les résultats possibles de l'état de conversion sont les suivants :
        - Totalement convertie : l'instruction sera convertie avec précision en COBOL.
        - Partiellement convertie : l'instruction est prise en charge mais utilise un paramètre ou une expression non pris en charge. Des ajustements manuels sont probablement nécessaires après la conversion.
        - Non convertie : l'instruction n'est pas prise en charge par le convertisseur.

    - Instructions de pré-compilation pour vérifier : elles sont normalement incluses dans les macros et font référence à ce que l'on appelle probablement aussi les instructions du langage d'assemblage conditionnel (par exemple, AIF, AGO) sur le mainframe. Elles sont gérées par le précompilateur, qui est piloté par de telles instructions ou directives, sélectionne et produit du code ASM propre/statique. Ces instructions dépendent des valeurs réelles des paramètres de macro qui sont compilés. Ainsi, la même macro peut générer différents morceaux de code ASM, en fonction des valeurs des paramètres transmis. Cela est dû à la présence de telles instructions de pré-compilation. Dans ce cas, envisagez d'étendre ou de réorganiser la macro.
  - `Conversion_Global_Statistics.txt`fournit un résumé de l'état de la conversion au niveau d'un composant.
  - `CrossReference_PgmToCpyMacro.txt`rapports sur les dépendances du programme Assembler sur les macros. Il fournit un moyen rapide de déterminer si des macros sont absentes du code téléchargé.
  - `CrossReference_PgmToPgm.txt`rend compte des dépendances du programme Assembler par rapport à d'autres programmes Assembler. Il fournit un moyen rapide de déterminer si des programmes Assembler sont absents du code téléchargé.
7. Retournez à la console AWS CodeBuild de service.
8. Vérifiez l'état de l'étape 2-awsm2ccm-analysis. Cela aurait dû réussir sous l'onglet État de la dernière version.

Vous êtes prêt pour l'étape suivante : la conversion du code.

Étape 9 : Exécuter la conversion du code

> **Important**
>
> AWS Mainframe Modernization L'étape de conversion du code sera facturée en fonction de votre utilisation. Pour de plus amples informations sur la facturation, veuillez consulter the section called “Comprendre la facturation par conversion de code”.

Au cours de cette étape, vous allez configurer le processus de conversion, puis démarrer la génération.

1. À partir du AWS Management Console, accédez au service Amazon S3.

2. Localisez et cliquez sur le compartiment Amazon S3 :`codebuild-regionId-accountId-bucket`.

   a. Accédez à `prj_codebuild_01/`.

   b. Sélectionnez`project_settings.json`, puis choisissez Télécharger.

   c. Ouvrez le `project_settings.json` fichier pour voir la structure JSON suivante :

```
{
"Source programs directory":"srclib",
"Source copybooks/macros directory":"macrolib",
"Copybook/Macros Conversion":"Called_only",
"Do not regenerate the Copy/Macro if already exists":"false",
"Target Compiler":"IBM",
"Endianess":"Big",
"Converted programs extension":"",
"Converted CICS programs extension":"",
"Converted copies/macros extension":"",
"Trace Level":"STANDARD",
"Trace file open mode":"append",
"Data definition level":5,
"Start picture column":40,
"Generate Sync FILLER with name":"FILL-SYNC",
"Use SYNC clause":"yes",
"Decimal Point Comma":"true",
"Original Source Placement":"RIGHT"
}
```

   Où :

   - Répertoire du programme source : contient les programmes Assembler nécessaires à la conversion.
   - Répertoire source Copybooks/macros : contient les macros Assembler et les cahiers nécessaires à la conversion.
   - La conversion des copybooks/macros peut être soit :
     - Tout : Ce bouton radio indique que la conversion complète convertira toutes les copybook/Macros données disponibles dans le répertoire, qu'elles soient utilisées ou non par les programmes.
     - Called_only : Ce bouton radio indique que la conversion complète ne convertira que le contenu réellement utilisé par les copybook/Macros programmes.

- > **Important**
  >
  > Vous n'avez pas besoin de le régénérer Copy/Macro s'il existe déjà.

  Lorsque cela est vrai, l'outil ne convertira pas à nouveau le copybook/macro s'il est déjà converti (il existe dans le dossier de sortie).
- Cible : La conversion des programmes (code généré) dépend du compilateur COBOL cible. Les options suivantes sont prises en charge :
  - « IBM » pour mainframe IBM
  - « MF » pour Micro Focus COBOL
  - « VERYANT » pour Veryant iScobol
  - « NTT » pour NTT DATA Enterprise COBOL (Unikix)
- Endianess et bitness : La conversion des programmes (code généré) dépend de la plateforme cible (bits/endianess). Cette combinaison permet de sélectionner les options prises en charge suivantes :
  - Endianess : Big (pour Big-Endian) /Little (Little-Endian). Par exemple, le z/OS mainframe IBM est Big-Endian, Windows est Little-Endian, Linux varie selon les distributions (par exemple, Amazon Linux 2 est Little-Endian). EC2
  - Bitness : 32/64 (s'il n'est pas indiqué, la valeur par défaut sera 32). Le réglage recommandé est de 32 bits.
- Extension du programme converti : Il s'agit de définir l'extension de fichier pour les programmes COBOL générés. Vide (« ») : aucune extension. Pour les cibles COBOL de Rocket Software (anciennement Micro Focus), CBL est recommandé pour permettre à Rocket Enterprise Developer de reconnaître correctement les fichiers.
- Extension de programme CICS convertie : Ceci permet de définir l'extension de fichier pour les programmes CICS COBOL générés. Vide (« ») : aucune extension. Pour les cibles COBOL de Rocket Software, CBL est recommandé pour permettre à Rocket Enterprise Developer de reconnaître correctement les fichiers.
- Extension de copybooks/macros convertis : Ceci permet de définir l'extension de fichier pour les copybooks COBOL générés. Vide (« ») : aucune extension. Pour les cibles COBOL de Rocket Software, le CPY est recommandé pour permettre à Rocket Enterprise Developer de reconnaître correctement les fichiers.

- Niveau de trace : le traçage est l'information enregistrée CodeBuild lors de la conversion. L'utilisateur peut sélectionner le niveau de détail en sélectionnant l'une des options proposées.
  - ERREUR = ERREUR DE TRACE : seules les erreurs de conversion sont affichées.
  - STANDARD = TRACE STANDARD : les erreurs de conversion et les informations standard sont affichées. Il s'agit du paramètre recommandé.
  - TOUT = TRACE TOUT : niveau de traçage maximal
- Mode d'ouverture du fichier de trace : non utilisé. Le paramètre par défaut d'ajout est recommandé.
- Niveau de définition des données : Cela indique le niveau initial des sous-champs (après le niveau « 01 ») définis dans la section « work-storage » et « couplage ». Ça doit être un chiffre.
- Colonne d'image de départ : elle concerne le format du code COBOL généré et indique la colonne dans laquelle la clause PIC est placée (après les noms des champs). Ça doit être un chiffre.
- Emplacement de la source d'origine : cela indique la position où les commentaires sont placés dans le programme. Deux options s'offrent à vous :
  - DROITE : Cette option placera le commentaire ou les informations supplémentaires à la bonne position après la soixante-treizième (73) colonne. En COBOL, le code est écrit dans les soixante-douze premières (1-72) colonnes et tout ce qui se trouve dans la soixante-treizième colonne (**>=** 73) sera traité comme un commentaire.
  - CI-DESSUS : Cette option placera le commentaire au-dessus du contenu traduit.
- Générer un fichier de synchronisation avec le nom : Cette option est liée à l'alignement en mémoire des champs binaires (types de données Assembler « H », « F », « D », qui sont convertis en type de données COBOL « COMP »). Afin de garantir la bonne limite d'alignement, des champs de remplissage explicites seront ajoutés lors de la conversion. Il s'agit d'une option basée sur du texte, la valeur doit être une chaîne (comme FILL-SYNC).
- Utiliser la clause SYNC : cette option fait référence à l'alignement en mémoire des champs binaires. Oui = tous les champs sont convertis en COBOL. « COMP » sera défini avec la clause « SYNC » (par exemple, 05 WRKFLD PIC S9 (09) COMP SYNC).
- Virgule décimale : lorsque cela est vrai, la clause DECIMAL-POINT IS COMMA sera ajoutée au paragraphe COBOL « SPECIAL-NAMES ».

d. En fonction de vos besoins, modifiez les paramètres appropriés, puis enregistrez le`project_settings.json`.

e. Supprimez le `project_settings.json` fichier existant `prj_codebuild_01/` du compartiment Amazon S3, puis chargez la nouvelle version.

3. Retournez au AWS CodeBuild service.
4. Sélectionnez le projet à construire que vous avez créé précédemment : `3-awsm2ccm-convert`

   a. Choisissez Start build, puis Start now pour convertir les programmes Assembler et les macros en programmes COBOL et en copybooks.

   b. Attendez que le statut de construction passe à Succeeded pour ce projet. Il se trouvera sous l'onglet État de la dernière version.

Étape 10 : Vérifiez la conversion du code

1. À partir du AWS Management Console, accédez au service Amazon S3.
2. Localisez et cliquez sur le compartiment Amazon S3 :`codebuild-regionId-accountId-bucket`.
3. Naviguez vers **`awsm2ccm-do-not-delete`** . AWS Mainframe Modernization La conversion de code crée des fichiers binaires codés pour chaque module Assembler ou Macro pendant le processus de conversion. Ces fichiers sont essentiels pour éviter la double facturation aux clients et pour suivre la quantité de code Assembler fourni qui a été analysée et convertie. Les fichiers sont stockés à l'emplacement suivant :`codebuild-regionId-accountId- bucket/awsm2ccm-do-not-delete/<your_AWS_account_id>/Hash`. Les fichiers encodés ne contiennent aucun code assembleur et il n'est pas non plus possible d'extraire le code client de ces fichiers.

   > **Important**
   >
   > Vous ne devez ni modifier manuellement ces fichiers ni les supprimer. La modification ou la suppression de ces fichiers peut entraîner plusieurs facturations pour les mêmes composants.

   Traitez le **`awsm2ccm-do-not-delete/`**dossier comme un répertoire géré par le système. Consultez Support avant d'apporter des modifications à ce répertoire ou à son contenu.

4. Cliquez `codebuild-regionId-accountId-bucket` pour revenir au compartiment.
5. Choisissez **`ARTIFACTS/prj_codebuild_01/`**. Le dossier _Converted/ contient les sorties COBOL générées à la suite de l'étape de conversion du code. Il comportera les sous-répertoires suivants :
   - Le dossier copybooks/ contient les copybooks COBOL générés.
   - le dossier programs/ contient les programmes COBOL générés.
   - Le dossier runtime_lib/ contient des programmes COBOL et des copybooks supplémentaires fournis par la solution.
6. Si les rapports d'analyse et les autres rapports indiquent que la conversion a été réussie et que le AWS CodeBuild projet `3-awsm2ccm-convert` est marqué comme réussi, téléchargez le code COBOL et les cahiers depuis le répertoire _Converted/.

Étape 11 : Téléchargez le code converti

Au cours de cette étape, téléchargez le code COBOL et les cahiers depuis le répertoire _Converted/, puis compilez-les dans l'environnement COBOL cible.

1. À partir du AWS Management Console, accédez au service Amazon S3.
2. Localisez et cliquez sur le compartiment Amazon S3 :`codebuild-regionId-accountId-bucket`.
3. Accédez à l'emplacement : `ARTIFACTS/prj_codebuild_01/_Converted/`.
4. Téléchargez le code COBOL converti depuis tous les sous-répertoires sous _Converted/. Vous pouvez également utiliser la commande CLI suivante pour les télécharger en une seule fois :

```
aws s3 cp s3://codebuild-regionId-accountId-
bucket/ARTIFACTS/prj_codebuild_01/_Converted/ . --recursive
```

5. Analysez et compilez le COBOL converti dans l'environnement COBOL cible.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Supprimez les compartiments S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez Supprimer un compartiment dans le guide de l'utilisateur d'Amazon Simple Storage Service.
- Supprimez les politiques IAM que vous avez créées pour ce didacticiel. Pour plus d'informations, consultez la section Suppression des politiques IAM dans le guide de l'utilisateur IAM.
- Supprimez le rôle IAM que vous avez créé pour ce didacticiel. Pour plus d'informations, consultez la section Suppression de rôles ou de profils d'instance dans le guide de l'utilisateur IAM.
- Supprimez le CodeBuild projet que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un projet de construction CodeBuild dans le guide de AWS CodeBuild l'utilisateur.

Intégration de Charon

Présentation de Charon-SSP

En 1987, Sun Microsystems a lancé le processeur SPARC V7, un processeur RISC 32 bits. Le SPARC V8 a suivi en 1990, une révision du SPARC V7 original, avec notamment l'inclusion d'instructions matérielles de division et de multiplication. Les processeurs SPARC V8 ont constitué la base d'un certain nombre de serveurs et de stations de travail tels que les modèles SPARCstation 5, 10 et 20. En 1993, le SPARC V8 a été suivi par le processeur SPARC V9 64 bits. Cela est également devenu la base d'un certain nombre de serveurs et de stations de travail, tels que les Enterprise 250 et 450.

En raison de l'obsolescence du matériel et du manque de pièces de rechange ou remises à neuf, les logiciels et systèmes développés pour ces anciens postes de travail et serveurs basés sur SPARC sont devenus plus difficiles à entretenir. Pour répondre au besoin permanent de certains systèmes end-of-life basés sur SPARC, Stromasys S.A. a développé la gamme Charon-SSP d'émulateurs SPARC. Les produits suivants remplacent des machines virtuelles basées sur des logiciels pour les systèmes SPARC matériels natifs spécifiés. Vous trouverez ci-dessous un aperçu général des familles de matériel émulé.

Charon-SSP/4M émule le matériel SPARC suivant :

- Famille Sun-4m (représentée par le Sun SPARCstation 20) : à l'origine, une variante multiprocesseur Sun-4, basée sur le bus du module MBus processeur introduit dans la série 600MP. SPARCServer Plus tard, l'architecture Sun-4m a également inclus des systèmes non MBus monoprocesseurs tels que le SPAR Cstation 5, utilisant des processeurs d'architecture SPARC V8. Supporté à partir de SunOS 4.1.2 et de Solaris 2.1 à Solaris 9. SPARCServer Le support 600 MP a été abandonné après Solaris 2.5.1.

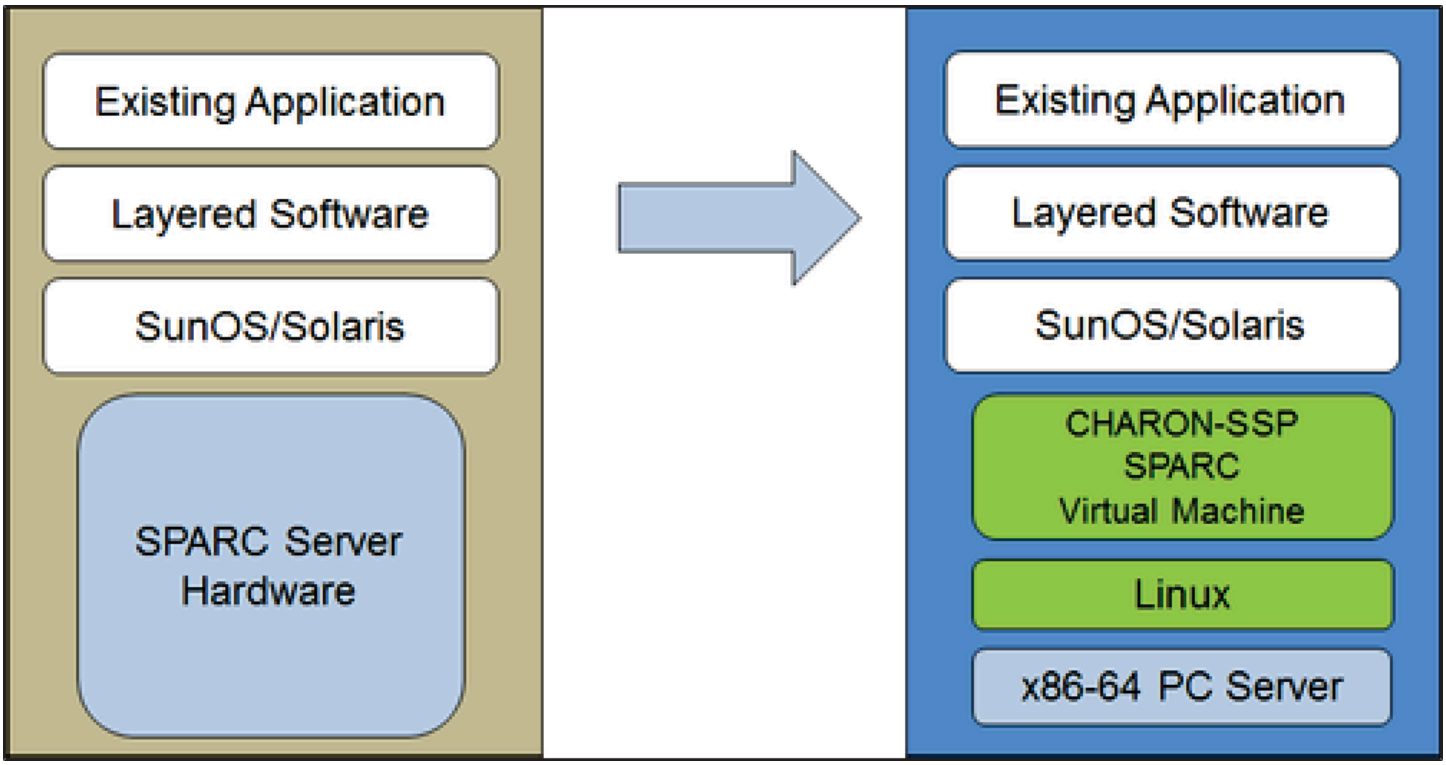
Charon-SSP/4U (+) émule le matériel SPARC suivant :

- Famille Sun-4u (représentée par le Sun Enterprise 450) : (U pour UltraSPARC) : cette variante a introduit l'architecture du processeur SPARC V9 64 bits et l'interconnexion des processeurs UPA utilisés pour la première fois dans la série Sun Ultra. Pris en charge par les versions 32 bits de Solaris à partir de la version 2.5.1. La première version 64 bits de Solaris pour Sun-4u était Solaris 7. Le support d'UltraSPARC I a été abandonné après Solaris 9. Solaris 10 prend en charge les implémentations Sun-4u d'UltraSP ARC II à UltraSPARC IV.

Charon-SSP/4V (+) émule le matériel SPARC suivant :

- Famille Sun-4v (représentée par les SPARC T2 et T4) : cette variante a ajouté la virtualisation des processeurs d'hyperviseur au Sun-4u, introduite dans le processeur multicœur Ultra SPARC T1. Le matériel sélectionné était pris en charge par Solaris version 10 à partir de la version 3/05 HW2 (la plupart des modèles, y compris le matériel émulé par Charon-SSP, nécessitent des versions plus récentes de Solaris 10). Plusieurs versions de Solaris 11 sont également prises en charge.

L'image suivante montre le concept de base de la migration du matériel physique vers un émulateur.



Les machines virtuelles Charon-SSP permettent aux utilisateurs d'ordinateurs basés sur Sun et Oracle SPARC de remplacer leur matériel natif d'une manière qui ne nécessite que peu ou pas de modification de la configuration système d'origine. Cela signifie que vous pouvez continuer à exécuter vos applications et vos données sans avoir à changer de plateforme ou à effectuer un portage sur une autre plateforme. Le logiciel Charon-SSP fonctionne sur des systèmes Intel 64 bits standard, garantissant ainsi la protection continue de votre investissement.

Charon-SSP/4U+ prend en charge les mêmes plateformes virtuelles SPARC que Charon-SSP/4U, et Charon-SSP/4v+ est identique à Charon-SSP/4V. Cependant, les versions 4U+ et 4V+ tirent parti de la technologie de virtualisation assistée par matériel VTx /EPT d'Intel et AMD-v/NPT d'AMD moderne pour offrir de meilleures performances du processeur virtuel. CPUs Charon-SSP/4U+ et Charon-

SSP/4v+ nécessitent le support VT-x/EPT ou AMD-V/NPT et doivent être installés sur un système hôte CPUs dédié. L'exécution de ces variantes de produit sur une machine virtuelle (par exemple, activée VMware) n'est pas prise en charge.

> **Note**
>
> Si vous envisagez d'exécuter Charon-SSP/4U+ ou 4V+ dans un environnement cloud, contactez Stromasys ou un VAR de Stromasys pour discuter de vos besoins.

Systèmes d'exploitation clients pris en charge

Les machines virtuelles Charon-SSP/4M prennent en charge les versions de système d'exploitation client suivantes :

- SunOS 4.1.3 - 4.1.4
- Solaris 2.3 à Solaris 9

Les machines virtuelles Charon-SSP/4U (+) prennent en charge les versions suivantes du système d'exploitation client :

- Solaris 2.5.1 à Solaris 10

Les machines virtuelles Charon-SSP/4V (+) prennent en charge les versions suivantes du système d'exploitation client :

- Solaris 10 (à partir de la mise à jour 4, 08/07) et Solaris 11.1 vers Solaris 11.4

Pour Charon-SSP/4V (+), notez ce qui suit :

- Pour le SPARC T4 émulé, les versions de Solaris 10 prises en charge sont les suivantes : Oracle Solaris 10 1/13, Oracle Solaris 10 8/11 et Solaris 10 9/10, ou Solaris 10 10/09 avec le jeu de correctifs Oracle Solaris 10 8/11.
- Le modèle SPARC T4 émulé est une condition préalable à l'exécution de Solaris 11.4 dans l'émulateur.
- Les zones du noyau Solaris ne sont pas prises en charge.

Conditions préalables à l'instance cloud Charon-SSP

En sélectionnant un type ou une forme d'instance, vous sélectionnez le matériel virtuel qui sera utilisé pour l'instance hôte Charon-SSP dans le cloud. Par conséquent, la sélection d'un type ou d'une forme d'instance détermine les caractéristiques matérielles du matériel hôte virtuel Charon-SSP (par exemple, le nombre de cœurs de processeur et la quantité de mémoire dont disposera votre système hôte virtuel Charon).

> **Note**
>
> Si vous utilisez une image du marché Charon-SSP pour lancer votre instance, toutes les exigences du système d'exploitation hôte Linux sont satisfaites.

La configuration matérielle minimale requise est décrite ci-dessous.

Points importants concernant les directives relatives aux tailles :

- Les directives de dimensionnement ci-dessous, en particulier en ce qui concerne le nombre de cœurs du processeur hôte et la mémoire hôte, indiquent les exigences minimales. Chaque situation de déploiement doit être revue et le dimensionnement réel de l'hôte doit être adapté si nécessaire. Par exemple, le nombre de cœurs de processeur disponibles pour les E/S doit être augmenté si les applications clientes génèrent une charge d'E/S élevée. En outre, un système comportant de nombreuses émulations CPUs est généralement capable de créer une charge d'E/S plus élevée et il peut donc être nécessaire d'augmenter le nombre de cœurs de processeur disponibles pour les E/S. Dans un environnement d'hyperthreading, pour de meilleures performances, le nombre de cœurs de processeur (réels ou physiques CPUs) doit être suffisant pour répondre aux exigences de processeur des émulateurs actifs, évitant ainsi que des threads à charge de travail élevée partagent un cœur de processeur physique.
- L'allocation des cœurs du processeur pour l'émulation CPUs et des cœurs du processeur pour le traitement des E/S est déterminée par la configuration. Voir Configuration du processeur dans le guide général de l'utilisateur de Charon-SSP pour plus d'informations à ce sujet et sur l'allocation par défaut des cœurs de processeur pour le traitement des E/S.

⚠ Informations générales importantes

- Pour faciliter le transfert rapide des données d'émulateur d'une instance cloud à une autre, il est vivement recommandé de stocker toutes les données pertinentes de l'émulateur sur un volume de disque distinct qui peut être facilement détaché de l'ancienne instance et attaché à une nouvelle instance.
- Assurez-vous de dimensionner correctement votre instance dès le début (vérifiez les exigences minimales ci-dessous). La licence Charon-SSP pour Charon-SSP AL est créée lors du premier lancement de l'instance. Le fait de changer ultérieurement de taille/type d'instance et donc de modifier le nombre de cœurs de processeur invalidera la licence et empêchera ainsi le démarrage des instances Charon (nouvelle instance requise). Si vous prévoyez d'utiliser l'instance Charon-SSP AL en mode AutoVE, veillez à inclure les informations du serveur AutoVE avant le premier lancement, sinon les serveurs de licences publics seront utilisés. La licence pour Charon-SSP VE est créée sur la base de l'empreinte digitale prise sur le serveur de licences. Si le serveur de licences est exécuté directement sur l'hôte de l'émulateur et que celui-ci nécessite ultérieurement, par exemple, une modification du nombre de cœurs de processeur, la licence sera invalidée (nouvelle licence et éventuellement nouvelle instance requises).

Prérequis pour l'instance

Exigences générales en matière de processeur : Charon-SSP prend en charge les processeurs d'architecture x86-64 modernes basés sur des instances Amazon. EC2

Exigences minimales pour Charon-SSP :

- Nombre minimal de cœurs de processeur du système hôte :
  - Au moins un cœur de processeur pour le système d'exploitation hôte, plus :
  - Pour chaque système SPARC émulé :
    - Un cœur de processeur pour chaque processeur émulé de l'instance, plus :
    - Au moins un cœur de processeur supplémentaire pour le traitement des E/S (au moins deux, si l'optimisation JIT du serveur est utilisée). Consultez la section Configuration du processeur mentionnée ci-dessus pour les options de configuration. Par défaut, Charon attribuera 1/3 (min. 1 ; arrondi au chiffre inférieur) du nombre de données CPUs visibles par l'hôte Charon au traitement des E/S.

- Mémoire minimale requise :
  - 4 Go ou plus de RAM pour le système d'exploitation hôte Linux. Les exigences réelles peuvent être plus élevées et dépendront des exigences des services non émulateurs exécutés sur l'hôte Linux. La recommandation précédente d'au moins 2 Go de RAM pour l'hôte Linux sera toujours valable pour de nombreux systèmes, mais les exigences croissantes du système d'exploitation et des applications Linux ont conduit à la mise à jour de la recommandation pour les nouvelles installations. De plus :
  - Pour chaque système SPARC émulé :
    - La mémoire configurée de l'instance émulée, plus :
    - 2 Go de RAM (6 Go de RAM si le serveur JIT est utilisé) pour permettre l'optimisation du DIT, les exigences de l'émulateur, les tampons d'exécution, le SMP et l'émulation graphique.
- Si l'hyperthreading est activé sur le x86-64 moderne CPUs, deux threads peuvent s'exécuter sur un cœur de processeur physique, fournissant deux fils logiques CPUs au système d'exploitation hôte. Si possible, désactivez l'hyperthreading sur l'hôte Charon-SSP. Cependant, cela n'est souvent pas possible dans VMware les environnements cloud, ou il n'est pas clair si l'hyperthreading est utilisé ou non. L'option d'hyperthreading Charon-SSP permet à Charon-SSP de s'adapter à de tels environnements. Consultez la section Configuration du processeur dans le guide d'utilisation général de Charon-SSP mentionné ci-dessus pour obtenir des informations de configuration détaillées. Remarque : pour de meilleures performances, les threads Charon-SSP ne doivent pas partager un cœur de processeur physique ; suffisamment de cœurs physiques doivent être disponibles sur le système hôte pour répondre aux exigences des émulateurs configurés.
- Une ou plusieurs interfaces réseau, selon les besoins du client.
- Charon-SSP/4U+ et Charon-SSP/4v+ doivent fonctionner sur du matériel physique compatible Intel VT-x/EPT ou AMD-V/NPT (instances baremetal) et ne peuvent donc pas fonctionner dans tous les environnements cloud. Consultez la documentation de votre fournisseur de cloud pour connaître la disponibilité de ce type de matériel. Notez également les points suivants :
  - Charon-SSP/4U+ et Charon-SSP/4v+ ne sont disponibles que si vous utilisez un noyau Linux pris en charge par Stromasys.
  - Si vous avez besoin de ce type de matériel SPARC émulé, contactez Stromasys ou votre VAR Stromasys pour discuter de vos besoins en détail.

Création et configuration d'une instance AWS cloud pour Charon (nouvelle interface graphique)

Cette section reflète la situation AWS Management Console au printemps 2022. Si vous utilisez toujours l'ancienne console, reportez-vous à l'annexe du guide de démarrage de Charon-SSP AWS .

Prérequis généraux

Cette description montre la configuration de base d'une instance Linux dans AWS. Il ne répertorie pas les prérequis spécifiques. Toutefois, en fonction de votre cas d'utilisation, tenez compte des conditions préalables suivantes :

- Compte Amazon et AWS Marketplace abonnements
  - Pour configurer une instance Linux dans AWS, vous avez besoin d'un AWS compte avec accès administrateur.
  - Identifiez la AWS région dans laquelle vous prévoyez de lancer votre instance. Assurez-vous que AWS les services que vous prévoyez d'utiliser sont disponibles dans cette région. Voir AWS Services par région.
  - Identifiez le VPC et le sous-réseau dans lesquels vous prévoyez de lancer votre instance.
  - Si votre instance nécessite un accès à Internet, assurez-vous que la table de routage associée à votre VPC dispose d'une passerelle Internet. Si votre instance nécessite un accès VPN à votre réseau local, assurez-vous qu'une passerelle VPN est disponible. La configuration exacte de votre VPC et de ses sous-réseaux dépend de la conception de votre réseau et des exigences de l'application.
  - Pour vous abonner à un AWS Marketplace service spécifique, sélectionnez AWS Marketplace Subscriptions dans le, AWS Management Console puis sélectionnez Gérer les abonnements.
  - Recherchez le service que vous comptez utiliser et abonnez-vous à celui-ci. Après un abonnement réussi, vous trouverez l'abonnement dans la section Gérer les abonnements. De là, vous pouvez directement lancer une nouvelle instance.
- Les prérequis matériels et logiciels de l'instance seront différents en fonction de l'utilisation prévue de l'instance :
  - Option 1 : l'instance doit être utilisée comme système hôte de l'émulateur Charon :
    - Reportez-vous aux sections relatives aux prérequis matériels et logiciels du guide de l'utilisateur et/ou du guide de démarrage de votre produit Charon pour déterminer les prérequis matériels et logiciels exacts qui doivent être remplis par l'instance Linux. L'image que vous

utilisez pour lancer votre instance et le type d'instance que vous avez choisi déterminent le logiciel et le matériel de votre instance cloud.

- Une licence de produit Charon est requise pour exécuter les anciens systèmes émulés. Reportez-vous aux informations de licence figurant dans la documentation de votre produit Charon, ou contactez votre représentant Stromasys ou le VAR de Stromasys pour plus d'informations.

- Option 2 : l'instance doit être utilisée comme serveur de licences VE dédié :
  - Consultez le guide du serveur de licences VE pour connaître les prérequis détaillés.

- Certains systèmes d'exploitation existants qui peuvent fonctionner dans les systèmes émulés fournis par les produits d'émulation Charon nécessitent une licence du fournisseur d'origine du système d'exploitation. L'utilisateur est responsable de toutes les obligations de licence liées à l'ancien système d'exploitation et doit fournir les licences appropriées.

Utilisation du AWS Management Console pour lancer une nouvelle instance

Pour créer une nouvelle instance

1. Connectez-vous à la EC2 console Amazon AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/ec2/.
2. Sélectionnez Launch instance (Lancer une instance).
3. Entrez un nom pour l'instance.
4. Sélectionnez une AMI. Une AMI est une image préemballée utilisée pour lancer des instances cloud. Il inclut le système d'exploitation et le logiciel d'application applicable. Le choix de l'AMI dépend de la manière dont vous prévoyez d'utiliser l'instance :

   - Si l'instance doit être utilisée comme système hôte de l'émulateur Charon, plusieurs choix d'AMI sont possibles :
     - Installation du système hôte Charon à partir d'une image de marché Charon prépackagée : ils contiennent le système d'exploitation sous-jacent et le logiciel Charon préinstallé.
       - Vérifiez auprès de votre représentant Stromasys quelles options sont actuellement disponibles sur le marché de vos fournisseurs de cloud.
       - Selon le fournisseur de cloud et les plans de lancement des produits Stromasys, il peut y avoir deux variantes :
         - Licences automatiques (AL) à utiliser avec un serveur de licences public géré par Stromasys ou avec un serveur de licences AutoVE privé géré par le client

    - Environnement virtuel (VE) à utiliser avec un serveur de licences VE privé géré par le client
  - Installation du système hôte Charon à l'aide d'une installation d'émulateur Charon classique avec les packages RPM d'installation de l'émulateur Charon pour Linux :
    - Choisissez une AMI Linux d'une distribution prise en charge par le produit et la version de Charon que vous avez sélectionnés. Consultez le guide d'utilisation de votre produit sur le site de documentation de Stromasys.
  - Si l'instance doit être utilisée comme serveur de licences VE dédié, consultez le guide du serveur de licences VE dans la documentation des licences pour connaître les exigences de l'instance Linux.

  Après avoir déterminé quelle AMI est requise, sélectionnez une AMI de produit Linux ou Charon correspondante. Si l'AMI dont vous avez besoin ne s'affiche pas, choisissez Parcourir davantage AMIs. Choisissez l'AMI Linux qui correspond à la manière dont vous prévoyez d'utiliser l'instance. Il peut s'agir de l'un des périphériques suivants :

  - Une image du marché Charon VE préemballée. Le nom de l'AMI inclura la chaîne « ve ».
  - Une image du marché Charon AL préemballée pour Automatic Licensing ou AutoVE.
  - Version Linux prise en charge pour l'installation d'un produit RPM.
  - Version Linux prise en charge pour le serveur de licences VE.
5. Sélectionnez un type d'instance. Amazon EC2 propose des types d'instances avec différentes combinaisons de capacité de processeur, de mémoire, de stockage et de réseau. Sélectionnez un type d'instance qui répond aux exigences du produit Charon que vous souhaitez utiliser. Certaines images du site de vente comportent une sélection limitée de types d'instances.
6. Sélectionnez une paire de clés existante ou créez-en une nouvelle et enregistrez-en une nouvelle. Si vous sélectionnez une paire de clés existante, assurez-vous de disposer de la clé privée correspondante. Dans le cas contraire, vous ne pourrez pas vous connecter à votre instance.

   > **Note**
   >
   > Si votre système de gestion le prend en charge, pour RHEL 9.x, Rocky Linux 9.x et Oracle Linux 9.x, utilisez une clé SSH de type ECDSA ou. ED25519 Ces types vous permettent de vous connecter à ces systèmes Linux hôtes Charon à l'aide d'un tunnel SSH sans avoir à modifier les paramètres de politique cryptographique par

défaut sur l'hôte Charon pour des paramètres moins sécurisés. Par exemple, cela est important pour le gestionnaire Charon-SSP. Consultez la section Utilisation de politiques cryptographiques à l'échelle du système dans la documentation Red Hat.

7. Dans la section Paramètres réseau, choisissez Modifier. Choisissez les paramètres qui correspondent à votre environnement.

   - Spécifiez un VPC.
   - Spécifiez un sous-réseau existant ou créez-en un nouveau.
   - Activez ou désactivez l'attribution automatique d'une adresse IP publique à l'interface principale. L'attribution automatique n'est possible que si l'instance possède une seule interface réseau.
   - Attribuez un groupe de sécurité personnalisé existant ou nouveau. Le groupe de sécurité doit au moins autoriser SSH à accéder à l'instance. Tous les ports requis par les applications que vous prévoyez d'exécuter sur l'instance doivent également être autorisés. Vous pouvez modifier le groupe de sécurité à tout moment après avoir créé l'instance.

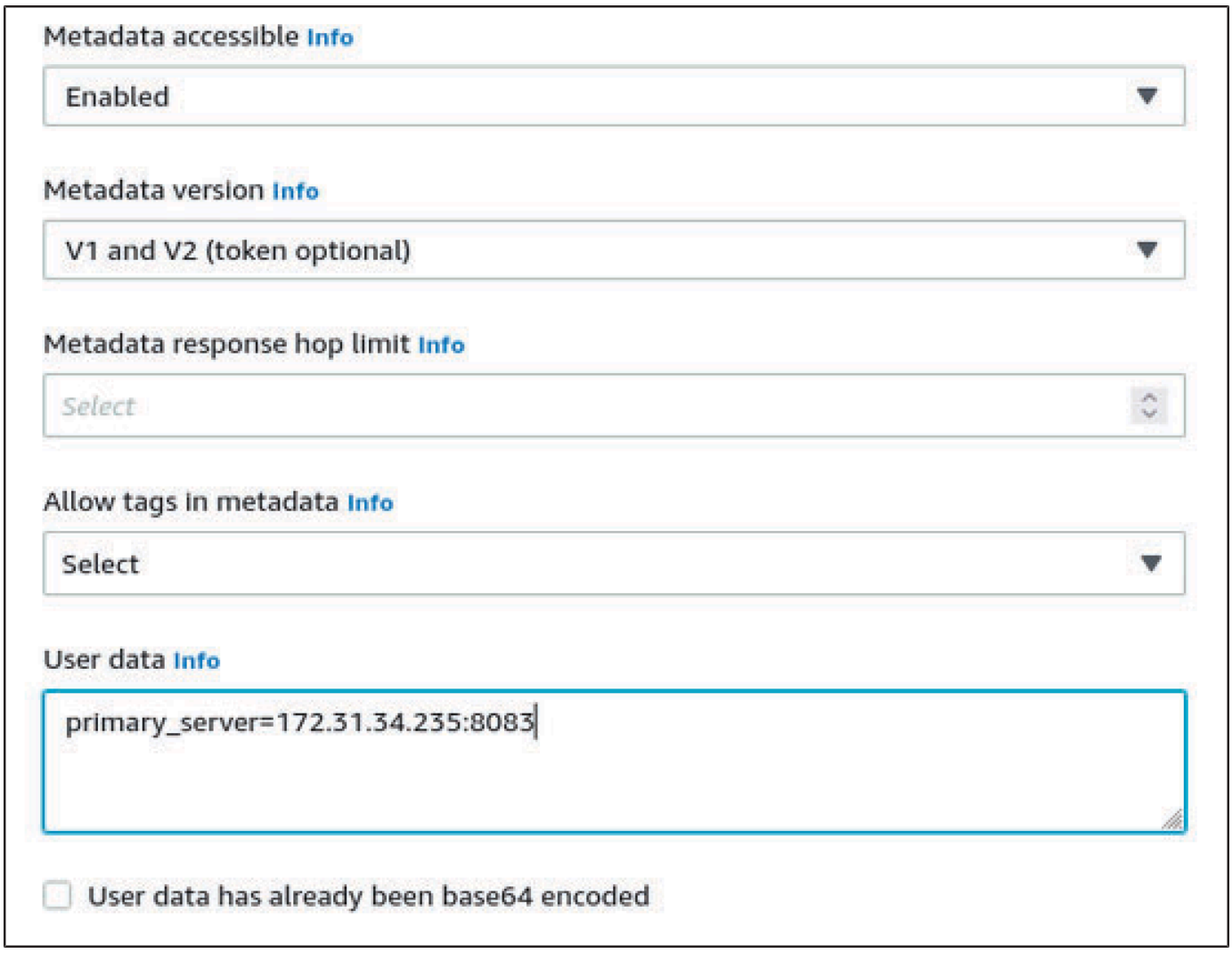
8. Dans la section Stockage, pour le volume racine (le disque système), choisissez une taille adaptée à votre environnement. La taille minimale du disque système recommandée pour le système Linux est de 30 GiB. Pour fournir de l'espace pour les conteneurs de disques virtuels et pour d'autres besoins de stockage, vous pouvez ajouter de l'espace de stockage maintenant ou après le lancement de l'instance. Toutefois, la taille du disque système doit couvrir les exigences du système Linux, y compris les applications et les utilitaires que vous prévoyez d'installer.

   > **Note**
   >
   > Nous vous recommandons de créer des volumes de stockage distincts pour les données de l'application Charon (par exemple, les images de disque). Si nécessaire, vous pouvez ultérieurement migrer ces volumes vers une autre instance.

9. Développez la section Détails avancés, faites défiler l'écran vers le bas et sélectionnez Spécifier les options du processeur. Trois des exemples les plus susceptibles d'être utiles à un environnement d'émulateur Charon sont présentés dans l'image suivante.

10. Pour un système de serveur de licences VE dont la version est antérieure à 1.1.23, vous devez attribuer le rôle IAM requis à l'instance. Ce doit être un rôle qui permet l'`ListUsers`action. Pour attribuer un rôle, dans la section détaillée des détails avancés, sélectionnez un rôle dans le profil d'instance IAM ou choisissez Créer un nouveau profil IAM. Pour plus d'informations, consultez la section Rôles IAM pour Amazon EC2.
11. Si votre instance est basée sur une AWS Marketplace image Charon AL et que vous prévoyez d'utiliser les serveurs de licences publics gérés par Stromasys, vous devez ajouter les informations correspondantes à la configuration de l'instance avant de lancer l'instance.

    Entrez les informations relatives au serveur de licences AutoVE comme indiqué dans l'image suivante.



Les options de configuration des données utilisateur valides sont les suivantes :

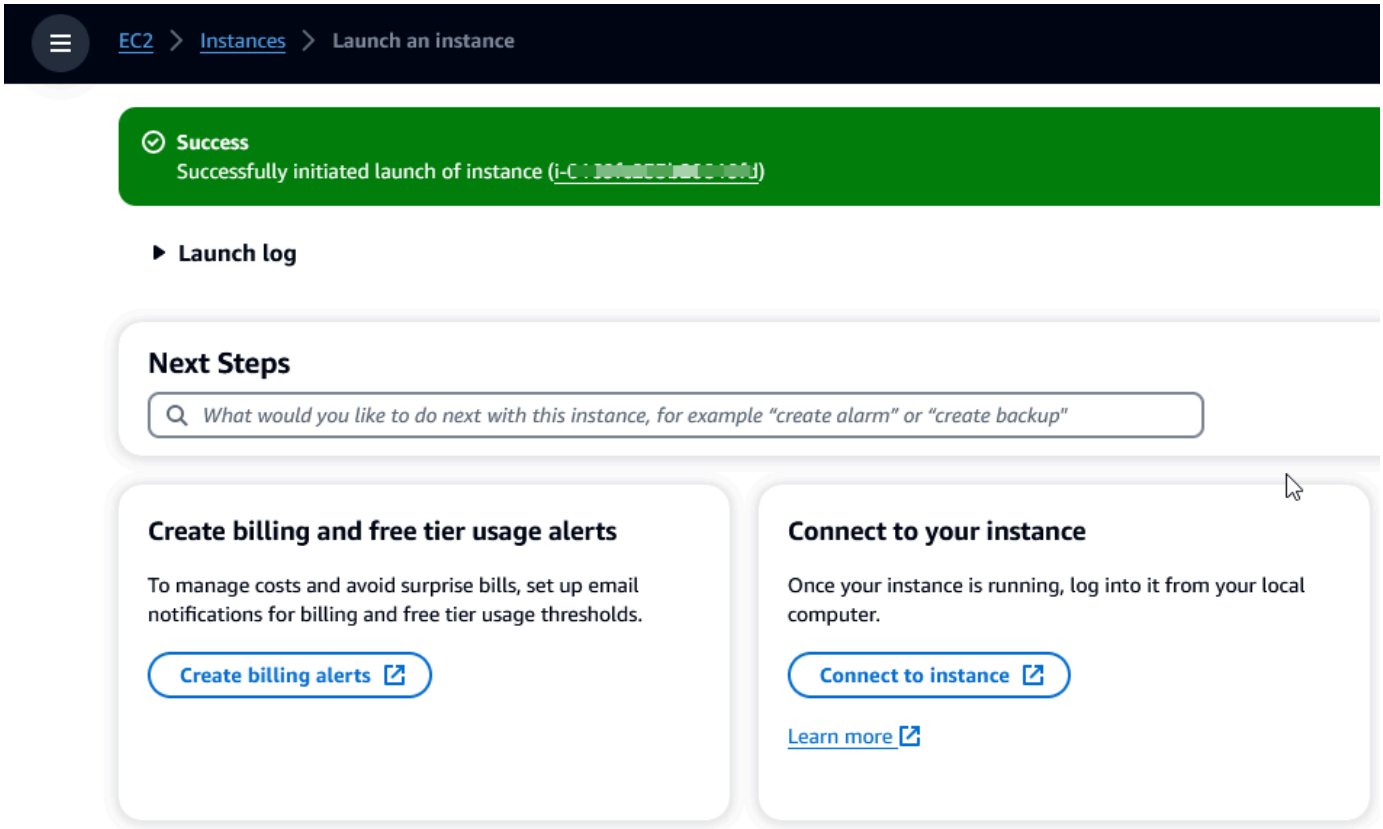
- **`primary_server=`***`<ip-address>`*`[:`*`<port>`*`]`
- **`backup_server=`***`<ip-address>`*`[:`*`<port>`*`]`

Où

- <ip-address>représente l'adresse IP du serveur principal et du serveur de sauvegarde, le cas échéant.
- <port>représente un port TCP autre que celui par défaut utilisé pour communiquer avec le serveur de licences (par défaut : TCP/8083).

> **Note**
>
> Au moins un serveur de licences doit être configuré lors du lancement initial pour activer le mode AutoVE. Sinon, l'instance sera liée à l'un des serveurs de licences publics exploités par Stromasys.

12. Dans la section Résumé, choisissez Launch instance. Au bout d'un moment, le message de réussite suivant s'affichera :



13. Dans le coin inférieur droit de l'écran, choisissez Afficher toutes les instances.
14. Pour voir les détails de votre instance, cochez la case située à gauche de la ligne qui représente l'instance dans le tableau Instances. Les détails de votre instance apparaîtront dans la moitié inférieure de l'écran. Pour plus d'informations sur la connexion à votre instance, consultez Connect dans le guide de EC2 l'utilisateur Amazon.

AWS Modernisation du mainframe et replateforme avec NTT DATA

AWS Mainframe Modernization propose une variété d'images Amazon Machine (AMIs). Ils AMIs facilitent le provisionnement rapide des EC2 instances Amazon, en créant un environnement sur mesure pour le réhébergement et la replateforme des applications mainframe à AWS l'aide de NTT Data. Ce guide fournit les étapes nécessaires pour y accéder et les utiliser AMIs.

Prérequis

- Assurez-vous de disposer d'un accès administrateur à un AWS compte sur lequel vous pouvez créer des EC2 instances Amazon.
- Vérifiez que le service de modernisation du AWS mainframe est disponible dans la région où vous prévoyez de créer les EC2 instances Amazon. Consultez la liste des services AWS disponibles par région.
- Identifiez le VPC Amazon sur lequel vous souhaitez créer les instances Amazon EC2 .

Abonnez-vous à l'Amazon Machine Image

Lorsque vous vous abonnez à un produit AWS Marketplace, vous pouvez lancer une instance depuis l'AMI du produit.

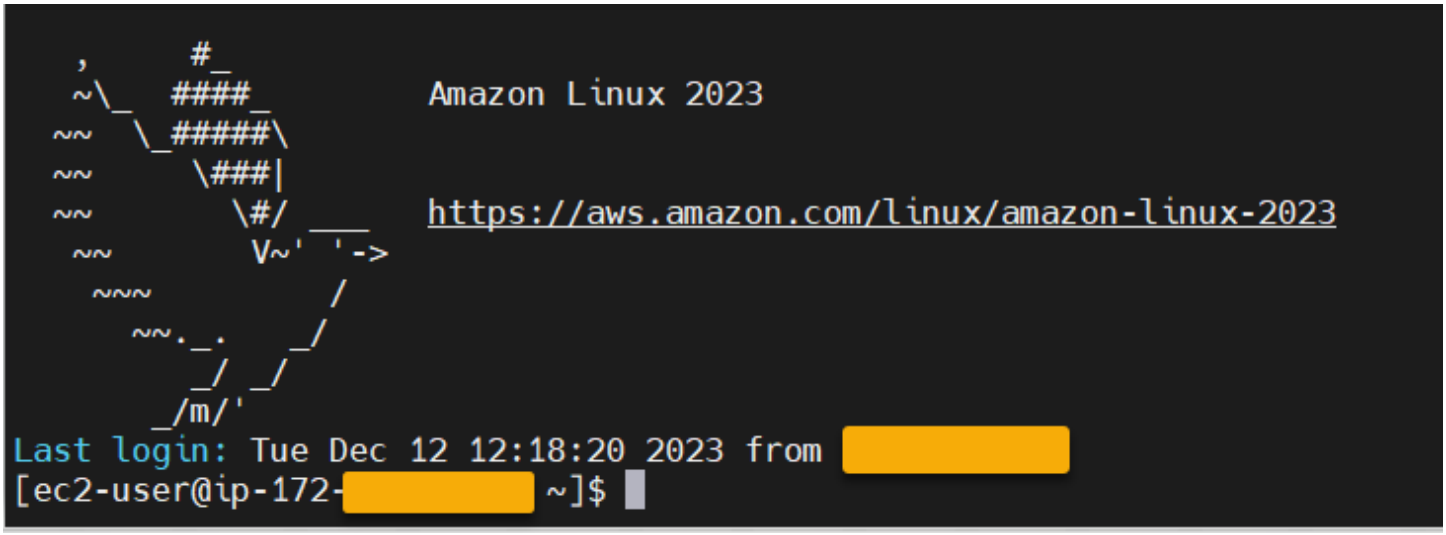
1. Connectez-vous à la AWS Marketplace console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/marketket.
2. Sélectionnez Manage subscriptions (Gérer les abonnements).
3. Copiez et collez le lien suivant dans la barre d'adresse du navigateur : https://aws.amazon.com/marketplace/pp/prodview-eg227ymldsnx2
4. Choisissez Continue to Subscribe (Continuer pour s'abonner).
5. Si les termes et conditions sont acceptables, choisissez Accepter les termes. Le traitement de l'abonnement peut prendre quelques minutes.
6. Attendez qu'un message de remerciement apparaisse. Ce message confirme que vous êtes bien abonné au produit.
7. Dans le volet de navigation de gauche, choisissez Gérer les abonnements. Cette vue affiche tous vos abonnements.

Lancez la replateforme de modernisation du AWS mainframe avec l'instance NTT DATA

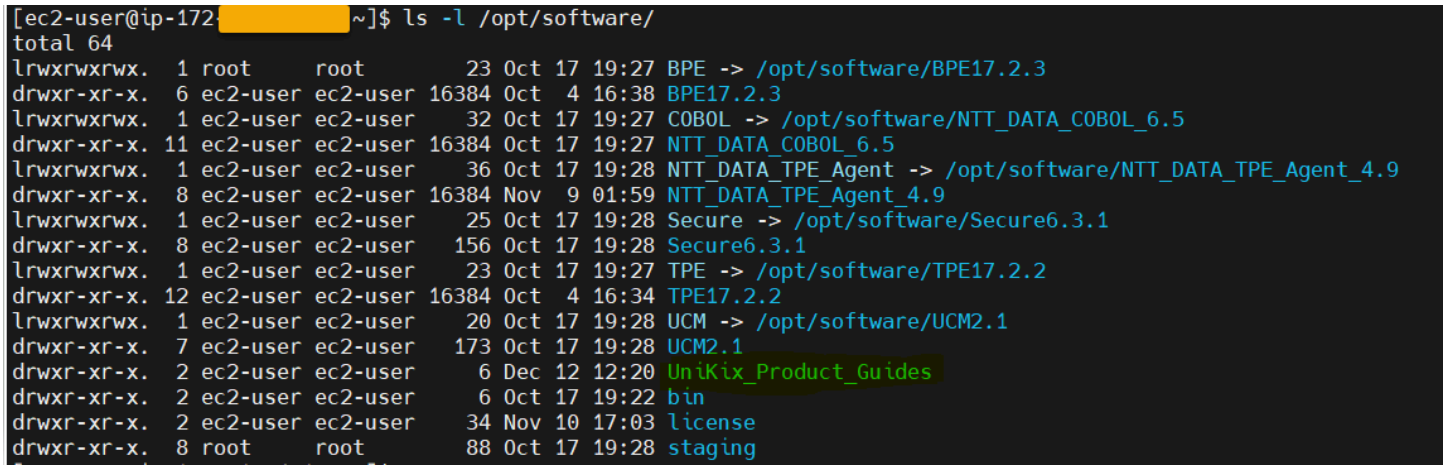
1. Ouvrez la AWS Marketplace console à l'adresse https://console.aws.amazon.com/marketplace.
2. Dans le volet de navigation de gauche, choisissez Gérer les abonnements.
3. Recherchez l'AMI que vous souhaitez lancer, puis choisissez Launch new instance.
4. Sous Région, sélectionnez la région autorisée.
5. Choisissez Continuer pour lancer EC2. Cette action vous amène à la EC2 console Amazon.
6. Entrez un nom pour le serveur.
7. Sélectionnez un type d'instance qui correspond aux exigences de performance et de coût de votre projet. Le point de départ suggéré pour la taille de l'instance est`c5.2xLarge`.
8. Choisissez une paire de clés existante ou créez-en une nouvelle et enregistrez-en une nouvelle. Pour plus d'informations sur les paires de clés, consultez la section relative aux paires de EC2 clés Amazon et aux instances Linux dans le guide de EC2 l'utilisateur Amazon.
9. Modifiez les paramètres réseau et choisissez le VPC autorisé et le sous-réseau approprié.
10. Choisissez un groupe de sécurité existant ou créez-en un nouveau. S'il s'agit d'une EC2 instance Amazon Enterprise Server, il est courant d'autoriser le trafic TCP vers les ports 86 et 10086 pour administrer la configuration du logiciel Rocket (anciennement Micro Focus).
11. Configurez le stockage pour l' EC2 instance Amazon.
12. Consultez le résumé et choisissez Launch instance. Pour que le lancement réussisse, le type d'instance doit être valide. Si le lancement échoue, choisissez Modifier la configuration de l'instance et choisissez un autre type d'instance.
13. Une fois le message de réussite affiché, choisissez Connect to instance.
14. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
15. Dans le volet de navigation de gauche, sous le menu Instances, sélectionnez Instances.
16. Dans le volet principal, vérifiez l'état de votre instance.

Commencer à utiliser NTT Data

Après avoir approvisionné l' EC2 instance Amazon, connectez-vous par SSH avec le nom `ec2-user` d'utilisateur. L'écran ressemblera à l'image suivante.



Sous le `/opt/software/` dossier se trouve un dossier nomméUniKix_Product_Guides, comme illustré dans l'image suivante.



Le `UniKix_Product_Guides` dossier contient la documentation des composants suivants installés sur cette EC2 instance Amazon :

- TYPE DE DONNÉES NTT
- BPE DE DONNÉES NTT
- NTT DATA Enterprise COBOL
- NTT DATA Sécurisé UniKix
- Directeur UniKix central de NTT DATA

Le `software` dossier qui apparaît dans l'image précédente contient les fichiers binaires des composants répertoriés ci-dessus.

Après avoir validé avec succès l' EC2 instance Amazon, commencez à utiliser AWS Mainframe Modernization Replatform avec NTT DATA en suivant la documentation de NTT Data.

Tutoriel : Déployer CardDemo une application sur NTT DATA

Cette page explique le step-by-step processus de déploiement de l' CardDemo exemple d'application sur la replateforme de modernisation du AWS mainframe avec le moteur d'exécution NTT DATA Unikix.

L' CardDemo exemple d'application est une application centrale simplifiée conçue et développée pour tester et présenter des technologies partenaires pour les cas d'utilisation de migration AWS et de modernisation des mainframes.

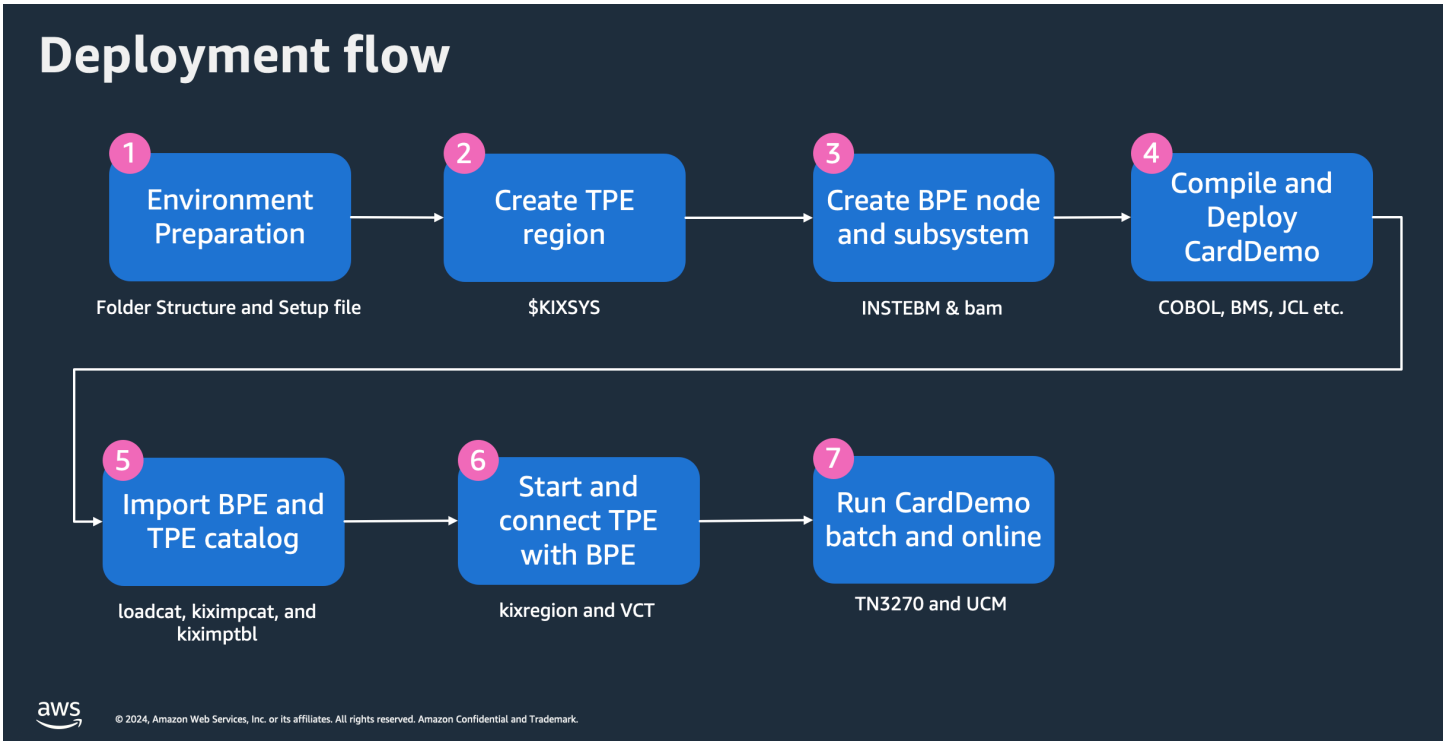
Pour plus d'informations sur cette application, voir GitHubréférentiel pour CardDemo.

Rubriques

- Schéma du flux de déploiement
- Prérequis
- Étape 1 : Préparation de l'environnement
- Étape 2 : Création d'une région TPE
- Étape 3 : Création du nœud et du sous-système BPE
- Étape 4 : Compiler et déployer CardDemo l'application
- Étape 5 : Importer le catalogue BPE et TPE
- Étape 6 : démarrer et connecter le TPE au BPE
- Étape 7 : Exécutez l' CardDemoapplication
- Résolution des problèmes

Schéma du flux de déploiement

Le schéma suivant montre chaque étape du processus de déploiement d'une application sur le runtime NTT DATA Unikix.



Prérequis

- Suivez les instructions fournies dans l'AMI Replateforme avec NTT DATA d'utilisation de l'AMI NTT DATA UniKix Marketplace.
- Modifiez l'option de métadonnées de l'instance IMDSv2sur Facultative, comme indiqué dans Restaurer l' IMDSv1utilisation du guide de EC2 l'utilisateur Amazon.
- Téléchargez les composants CardDemo d'exécution pour NTT DATA UniKix à partir du GitHub référentiel.
- Connectez-vous à l' EC2 instance d' UniKix exécution en tant que`ec2-user`.
- Extrayez les composants CardDemo d'exécution téléchargés à l'aide de ce lien : UniKix_ CardDemo _runtime_v1.zip.
  - Le répertoire extrait doit contenir `bin` des `migrated_app` répertoires.
  - Déplacez les deux `migrated_app` dossiers `bin` et les répertoires sous votre `$HOME` répertoire. Le chemin ressemblerait à ça`/home/ec2-user`.
  - Vous devriez avoir les répertoires suivants dans votre `$HOME` :
    - `/home/ec2-user/bin`
    - `/home/ec2-user/migrated_app`

- Déplacez tous les fichiers dans le répertoire $HOME/bin à l'aide de la commande suivante :
- • `chmod +x $HOME/bin/*`

Étape 1 : Préparation de l'environnement

Une fois les prérequis remplis, la première étape consiste à préparer l'environnement dans lequel vous souhaitez déployer l' CardDemo application.

1. Connectez-vous à l' EC2 instance d' UniKix exécution en tant que`ec2-user`.
2. Observez la liste des UniKix logiciels préemballés dans l'AMI, tels que TPE, BPE et COBOL, ainsi que d'autres logiciels provenant de l'emplacement du UniKix produit NTT DATA en utilisant la commande suivante dans votre instance : EC2

   ```
   ls -l /opt/software/
   ```

3. Examinez l' CardDemo application migrée. Vous verrez tout le code source, y compris les cartes BMS, les programmes COBOL, les copybooks COBOL et. JCLs Vous trouverez également l'exportation des catalogues BPE et TPE, des définitions de ressources CICS et des données migrées telles que les fichiers séquentiels et les fichiers VSAM en procédant comme suit :

   ```
   ls  $HOME/migrated_app/*/*
   ```

4. Créez une structure de dossiers en exécutant le `create_project` script à l'aide de la commande suivante :

   ```
   sh $HOME/bin/create_project
   ```

5. Activez l' CardDemo environnement en vous procurant le fichier de `carddemo.env` configuration à l'aide de :

   ```
   source $HOME/bin/carddemo.env
   ```

Étape 2 : Création d'une région TPE

Une fois que vous avez activé l'environnement dans lequel vous souhaitez déployer l'application, vous devez créer une région TPE.

1. Créez une région TPE à l'aide de la `kixregion createRegion` commande qui nécessite des entrées telles que `$KIXSYS$JAVA_HOME`, et`$KIXLICDIR`. Ces variables d'environnement sont déjà configurées dans le fichier de `carddemo.env` configuration.

```
kixregion createRegion $KIXSYS $JAVA_HOME $KIXLICDIR
```

2. Configurez la région TPE à l'aide de la `kixregion setAttr` commande.

```
kixregion setAttr $KIXSYS server.tx.languages.cobol.enabled true
kixregion setAttr $KIXSYS server.tx.languages.cobol.flavor vcobol
kixregion setAttr $KIXSYS server.tx.languages.cobol.home $VCOBOL
kixregion setAttr $KIXSYS maps.location $PROJECT_ROOT/maps
kixregion setAttr $KIXSYS programs.location $PROJECT_ROOT/loadlib
kixregion setAttr $KIXSYS environment.KIXDATA $KIXDATA
kixregion setAttr $KIXSYS td.jobq.submission.node $EBMHOME
kixregion setAttr $KIXSYS td.jobq.submission.subsys $EBMSYS
```

3. Générez le fichier d'environnement utilisateur spécifique à cette région TPE en exécutant la `kixregion createScript` commande. Cette commande crée ou met à jour `$KIXSYS/bin/userenv` en fonction de la configuration de la région TPE.

```
kixregion createScript $KIXSYS
```

4. Activez la région TPE en recherchant le fichier d'environnement utilisateur (`$KIXSYS/bin/userenv`).

```
source $KIXSYS/bin/userenv
```

5. Créez la région TPE en exécutant la `kixinstall2` commande.

```
kixinstall2
```

Étape 3 : Création du nœud et du sous-système BPE

Après avoir créé une région TPE, vous devez créer le nœud et le sous-système BPE en suivant ces étapes.

1. Modifiez le propriétaire et les autorisations de`INSTEBM`.

```
sudo chown root $INSTEBM
```

```
sudo chmod 4755 $INSTEBM
```

2. Créez un nœud BPE à l'aide de la `INSTEBM` commande. Le répertoire des nœuds BPE est fourni en tant que paramètre d'entrée.

```
$INSTEBM $EBMHOME
```

3. Activez l'environnement par lots en recherchant le `batchenv` fichier depuis le nœud BPE nouvellement créé.

```
source $EBMHOME/batchenv
```

4. Créez le sous-système BPE au sein de ce nœud à l'aide du Batch Administration Manager (bam). La `bam` commande ouvre l'interface du Batch Administration Manager.

```
bam
```

   a. Démarrez le nœud BPE à l'aide de l'interface BAM. Choisissez l'option 2, Environnements système dans le menu principal.

   b. Choisissez l'option 2, Start/ (Stop) Batch Node pour démarrer le nœud BPE.

```
                System Environments                          2024/09/17 21:27:03




             1   Report System Status
             2   Start/(Stop) Batch Node
             3   Assign the Console
             4   Change the Date
             5   Redirect Job Output
             6   Inter-Node Communications
             7   Job Accounting
             8   Assign the Initial Job Number
             9   Enable/Disable Duplicate Name Execution Delay : Enabled

             H - Help
             R - Return to Main Menu

--------------------------------------------------------------------------------
Type an option and press Return: 2
```

c. Une fois démarré, appuyez deux fois sur la touche Retour pour revenir au menu principal du BAM.

```
                System Environments: Start Batch Node        2024/09/17 21:28:28









              Batch Node Startup Completed.









--------------------------------------------------------------------------------
Press Return to Continue
```

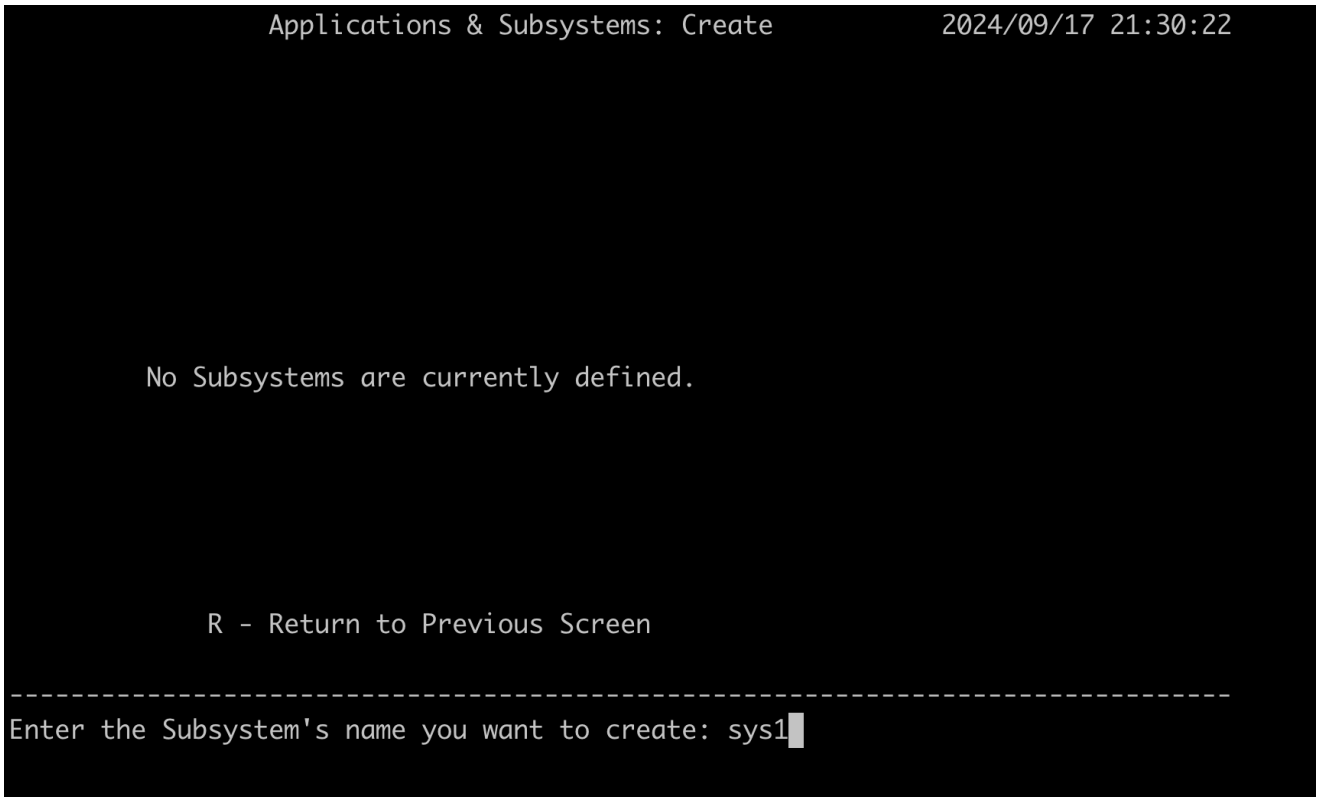
d. Pour créer le sous-système BPE, choisissez l'option 3, Applications et sous-systèmes.

```
                Batch Administration Manager                2024/09/17 21:29:03




             1   Software License Management
             2   System Environments
             3   Applications & Subsystems
             4   Security & Users
             5   Classes & Activities
             6   Problem Determination



             H - Help
             Q - Quit

--------------------------------------------------------------------------------
Type an option and press Return: 3
```

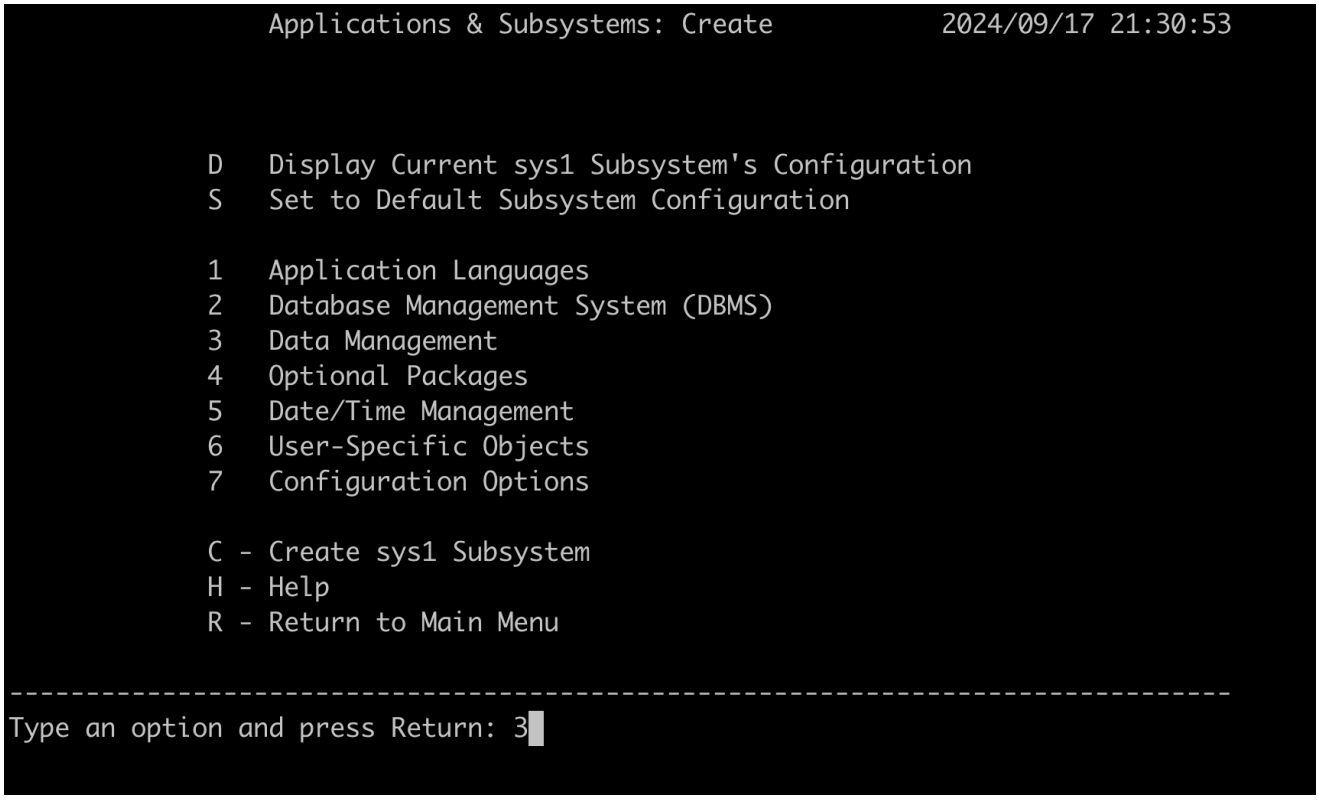
e. Choisissez ensuite l'option 3, Créer un sous-système.

```
                Applications & Subsystems                   2024/09/17 21:29:57




             1   List All Subsystems
             2   Query a Subsystem
             3   Create a Subsystem
             4   Update a Subsystem
             5   Delete a Subsystem
             6   Change/Show Default Subsystem
             7   Import a Subsystem (from another batch node)
             8   Create a BPESUB Project
             9   Export Subsystem bldsub Config File

             H - Help
             R - Return to Main Menu

--------------------------------------------------------------------------------
Type an option and press Return: 3
```

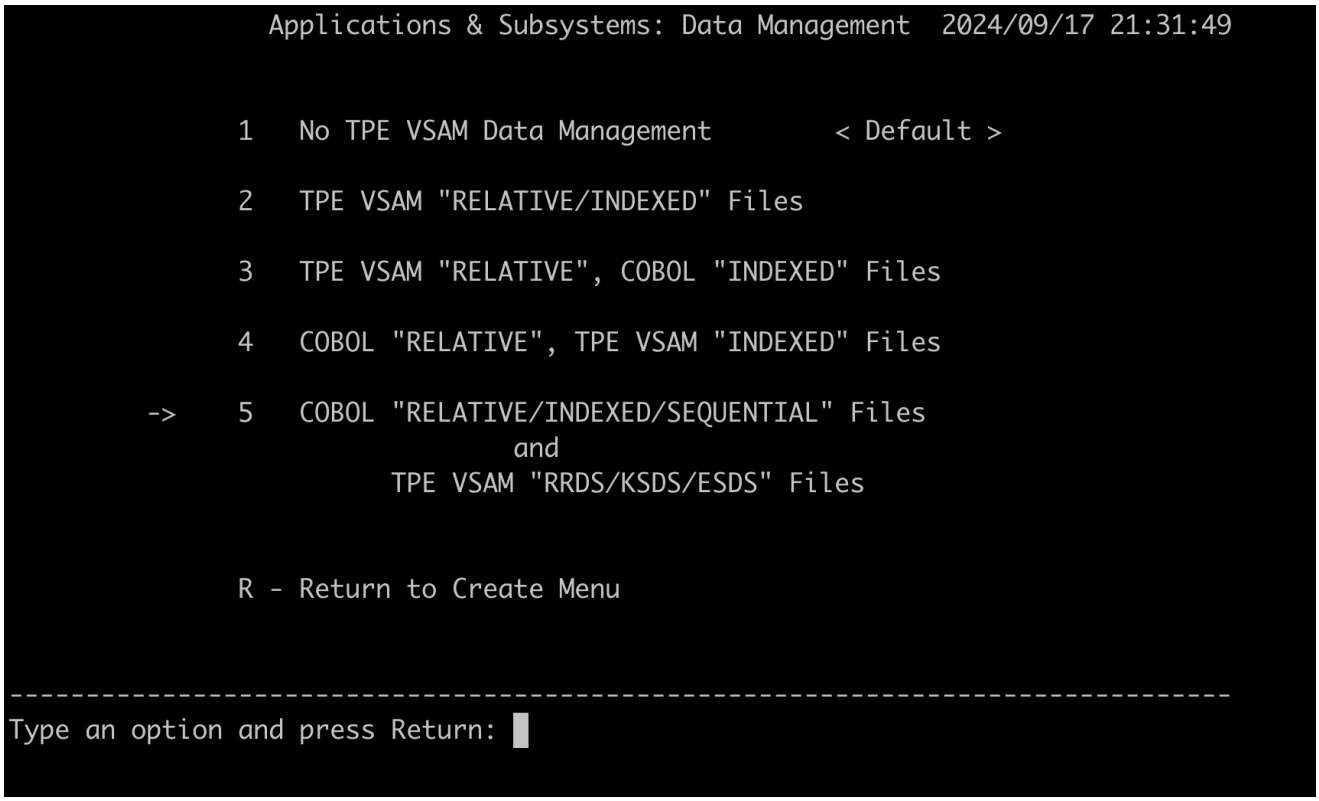
f. Entrez le nom du sous-système sous la forme`sys1`.



g. Choisissez l'option 3, Gestion des données.

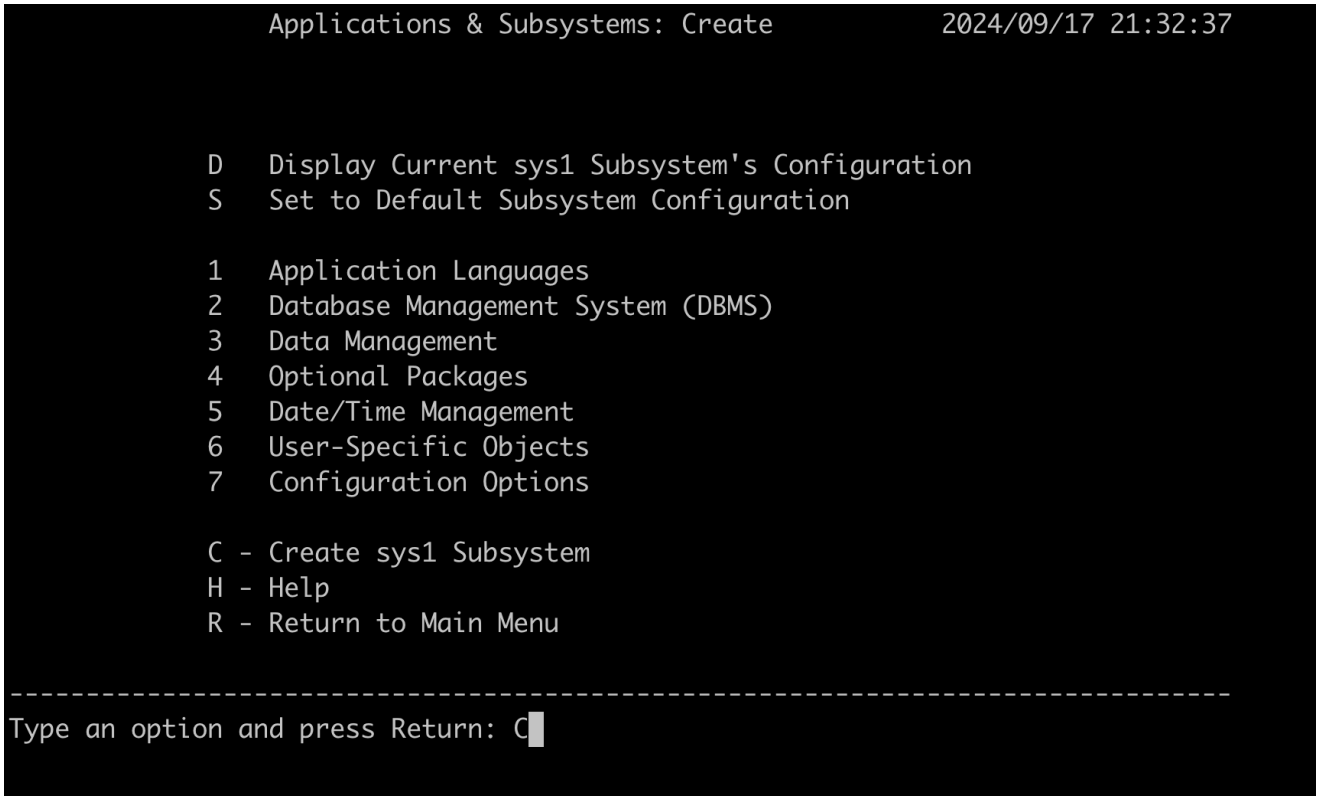


h. Choisissez l'option 5, car l' CardDemo application implique à la fois des fichiers séquentiels et des fichiers VSAM.



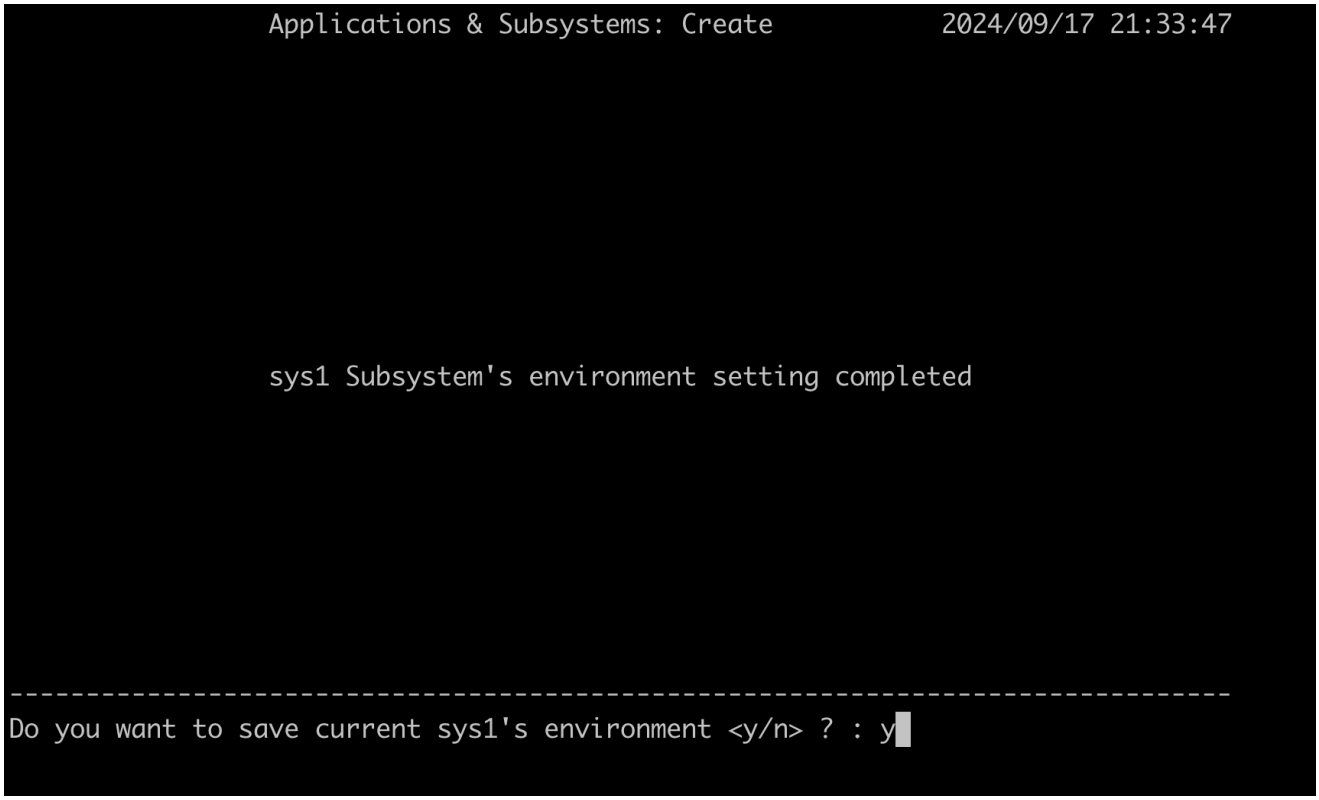
i. (Facultatif) Appuyez sur « R » pour revenir à la page Créer le menu, passez en revue les différentes options de configuration disponibles.

j. Sur la page Créer, entrez « C » pour créer le sous-système. `sys1`

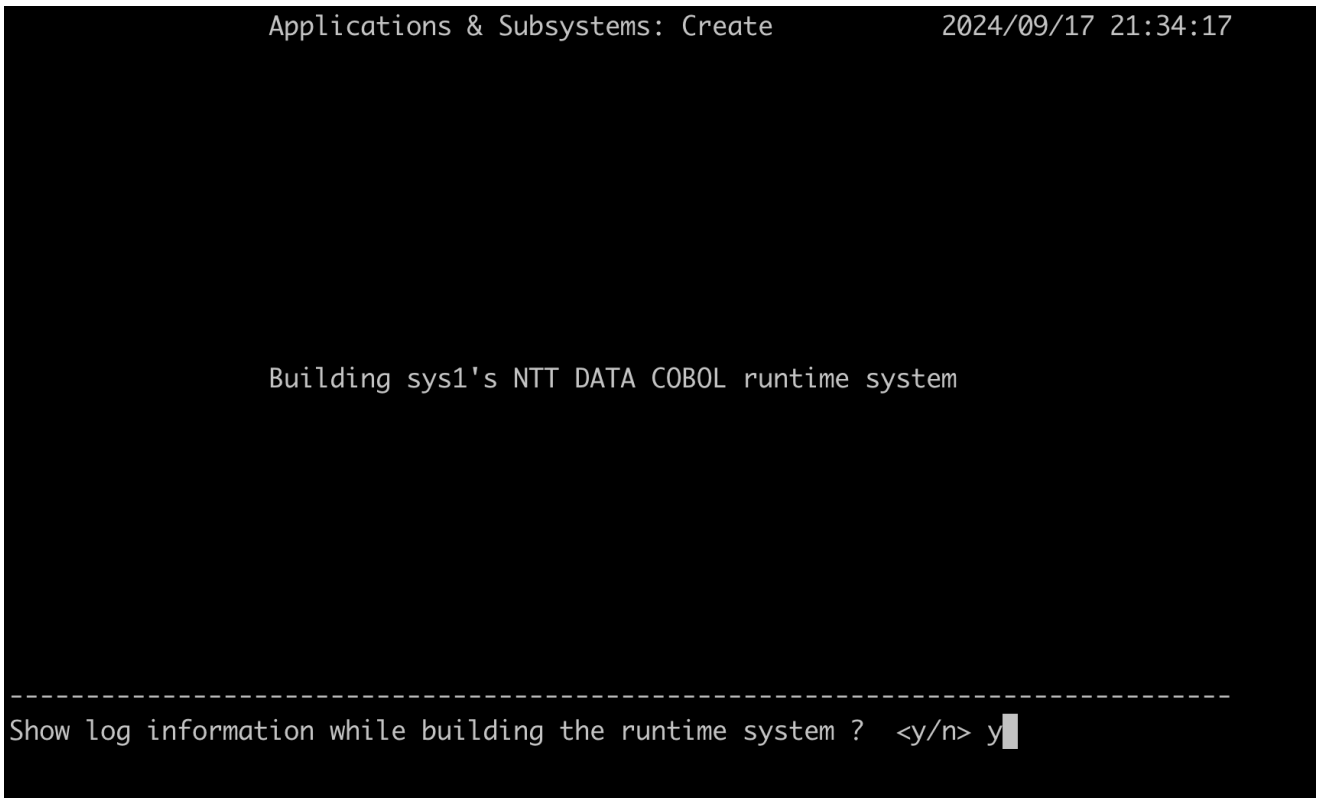


k. Passez en revue les paramètres et entrez « C » pour continuer avec les autres paramètres d'environnement. Ces paramètres d'environnement sont préremplis en fonction des variables d'environnement nécessaires définies dans le fichier de `carddemo.env` configuration et de la mise en place de la structure de dossiers recommandée.

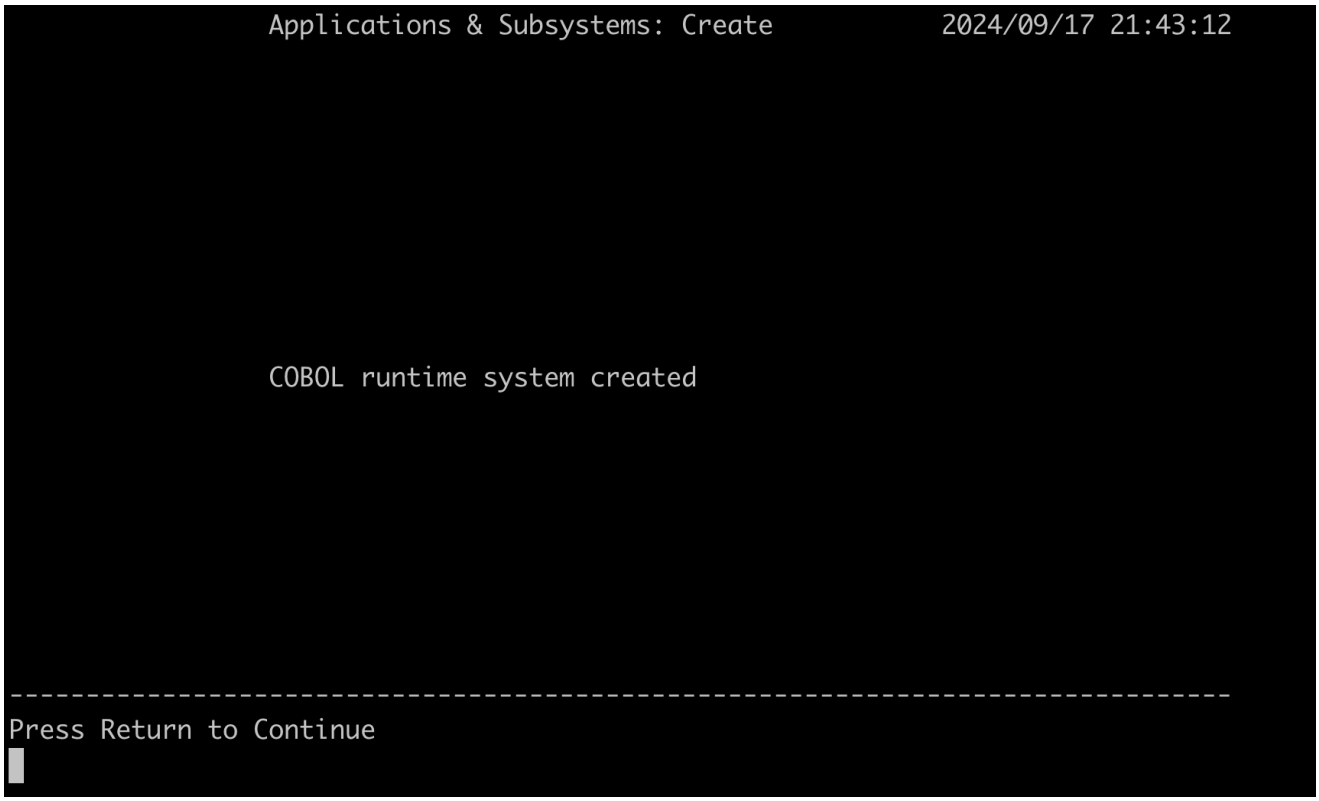
l. Entrez « y » pour confirmer et enregistrer les paramètres d'environnement actuels.

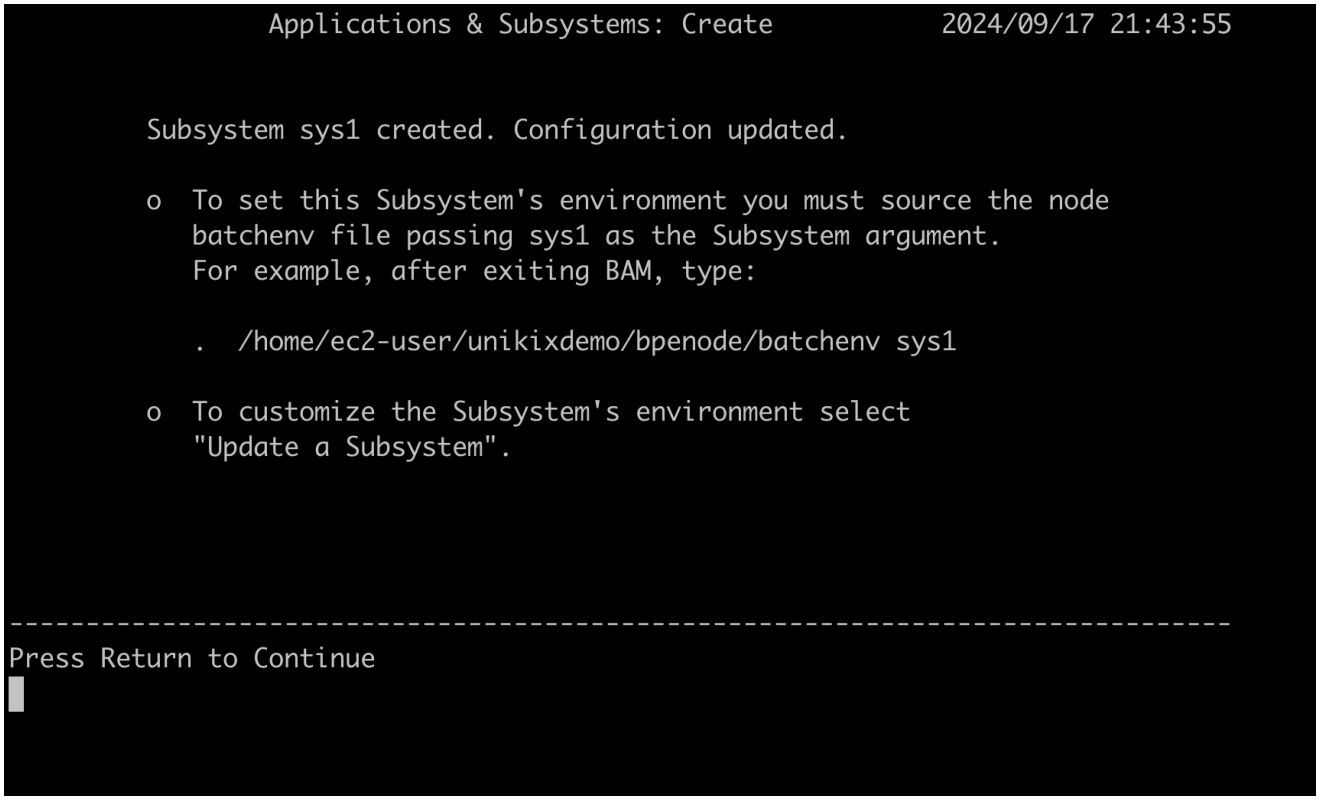


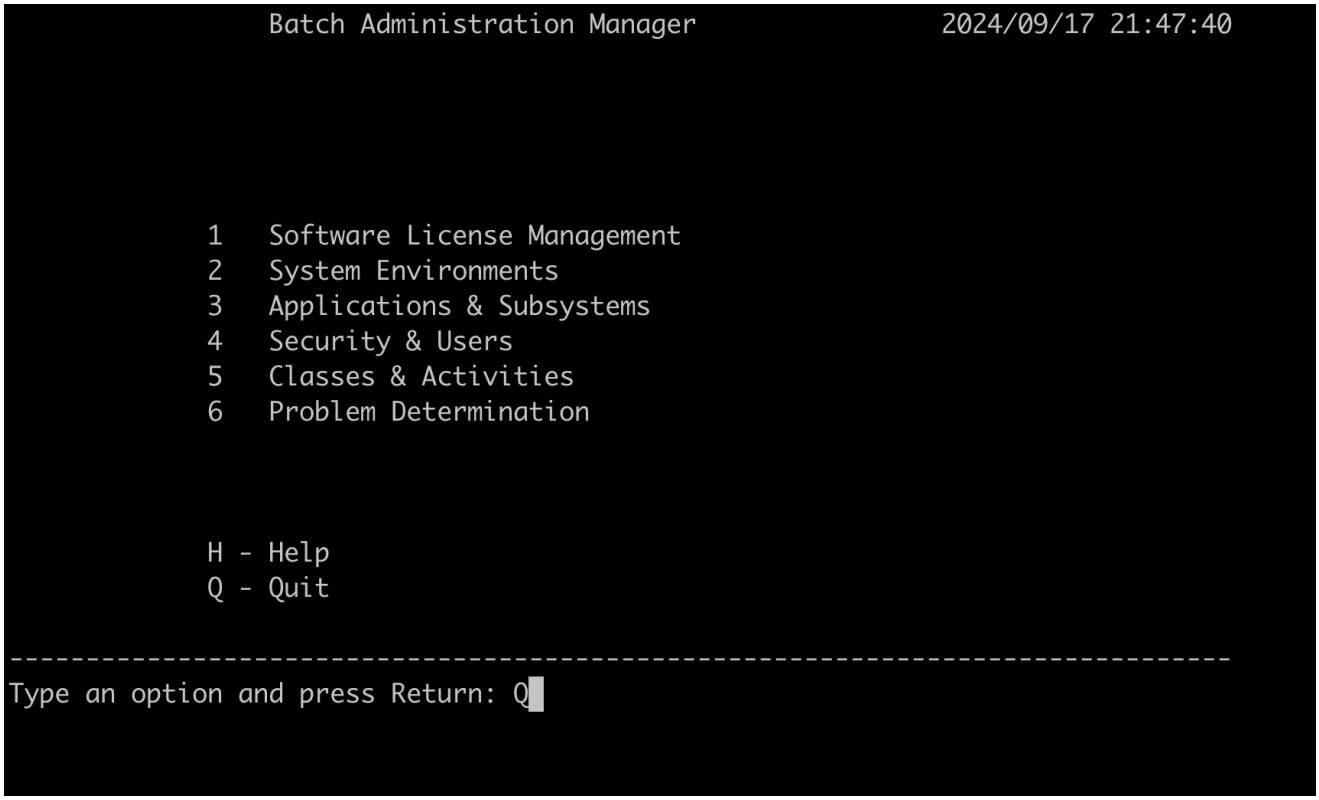
m. Entrez « y » pour afficher le journal lors de la création du sous-système.



n. Appuyez sur la touche Retour jusqu'à ce que vous reveniez au menu principal et que vous quittiez l'interface BAM en sélectionnant l'option Quitter.







5. Activez le sous-système BPE en le fournissant `batchenv` avec le nom du sous-système. `sys1`

```
source $EBMHOME/batchenv sys1
```

Étape 4 : Compiler et déployer CardDemo l'application

Au cours de cette étape, vous compilez les programmes COBOL et déployez des artefacts d'application tels que JCL, des procédures, des fichiers de données et des définitions de ressources CICS.

1. Activez à nouveau l' CardDemo environnement en recherchant le fichier de `carddemo.env` configuration.

```
source $HOME/bin/carddemo.env
```

2. Accédez au répertoire des sources COBOL.

```
cd $MIGAPP_DIR/cbl
```

3. Compilez le programme Cobol à `CBACT01C.cbl` l'aide d'un `compile` script.

```
compile CBACT01C.cbl
```

4. Compilez tous les programmes Cobol à l'aide d'un `compile.all` script.

```
compile.all
```

5. Accédez au répertoire source des cartes BMS.

```
cd $MIGAPP_DIR/bms
```

6. Compilez la carte BMS à `COACTUP.bms` l'aide d'un `compbms` script.

```
compbms COACTUP.bms
```

7. Compilez toutes les cartes BMS à l'aide d'un `compbms.all` script.

```
compbms.all
```

8. Vérifiez les fichiers binaires compilés pour les cartes COBOL et BMS.

```
ls $PROJECT_ROOT/loadlib
ls $PROJECT_ROOT/maps
```

9. Déployez d'autres artefacts d'application tels que JCL, des procédures, des fichiers de données et des définitions de ressources CICS à l'aide du `deploy_app` script.

```
deploy_app
```

10. Accédez au répertoire JCL du projet.

```
cd $PROJECT_ROOT/jcl
```

11. Translate JCL ACCTFILE en macro BPE JCL. Utilisez la `mvstrans` commande, en utilisant l'option « -v » pour la vérification JCL, et l'option « -f » pour créer la macro.

```
mvstrans ACCTFILE -v
mvstrans ACCTFILE -f
```

12. Translate la procédure JCL REPROC en macro de procédure BPE JCL. Utilisez la `mvstrans` commande avec l'option « -p » ainsi que l'option « -v » pour la vérification, et l'option « -f » pour créer la macro.

```
mvstrans REPROC -v -p
mvstrans REPROC -f -p
```

13. Translate all JCLs et les procédures JCL.

```
for file in "./jmvs/*"; do mvstrans $file -f; done > jmvs.out
for file in "./mvsp/*"; do mvstrans $file -p -f; done > mvsp.out
```

Étape 5 : Importer le catalogue BPE et TPE

Au cours de cette étape, vous importez le catalogue BPE et TPE à l'aide de différentes commandes.

1. Importez le catalogue BPE à l'aide de `loadcat` la commande.

```
loadcat $MIGAPP_DIR/catlg/bpe/BPECAT*
```

2. Accédez au répertoire $KIXSYS.

```
cd $KIXSYS
```

3. Importez le catalogue TPE à l'aide de `kiximpcat` la commande.

```
kiximpcat -c CATALOG -l CATALOG.lst
```

4. Importez les définitions de ressources CICS à l'aide de la commande kiximptbl.

```
kiximptbl
```

Étape 6 : démarrer et connecter le TPE au BPE

Au cours de cette étape, vous devez démarrer la région TPE créée précédemment avec le gestionnaire BPE et les connecter pour pouvoir exécuter l'exemple d' CardDemo application.

1. Exécutez la `kixverify` commande sur tous les fichiers VSAM pour vous assurer qu'ils sont réinitialisés et que tous les fichiers ouverts précédemment sont fermés.

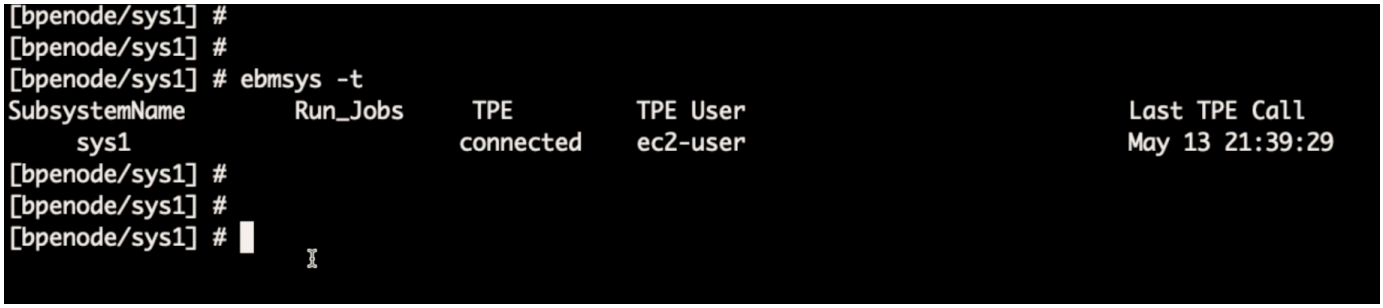
```
kixverify -r ALL
```

2. Démarrez la région TPE.

```
kixregion start $KIXSYS
```

3. Assurez-vous que le BPE et le TPE sont connectés. Cela est crucial car les fichiers VSAM appartiennent à TPE, et toute opération par lots permettant d'accéder au VSAM nécessitera une connexion au TPE.

```
ebmsys -t
```



Étape 7 : Exécutez l' CardDemoapplication

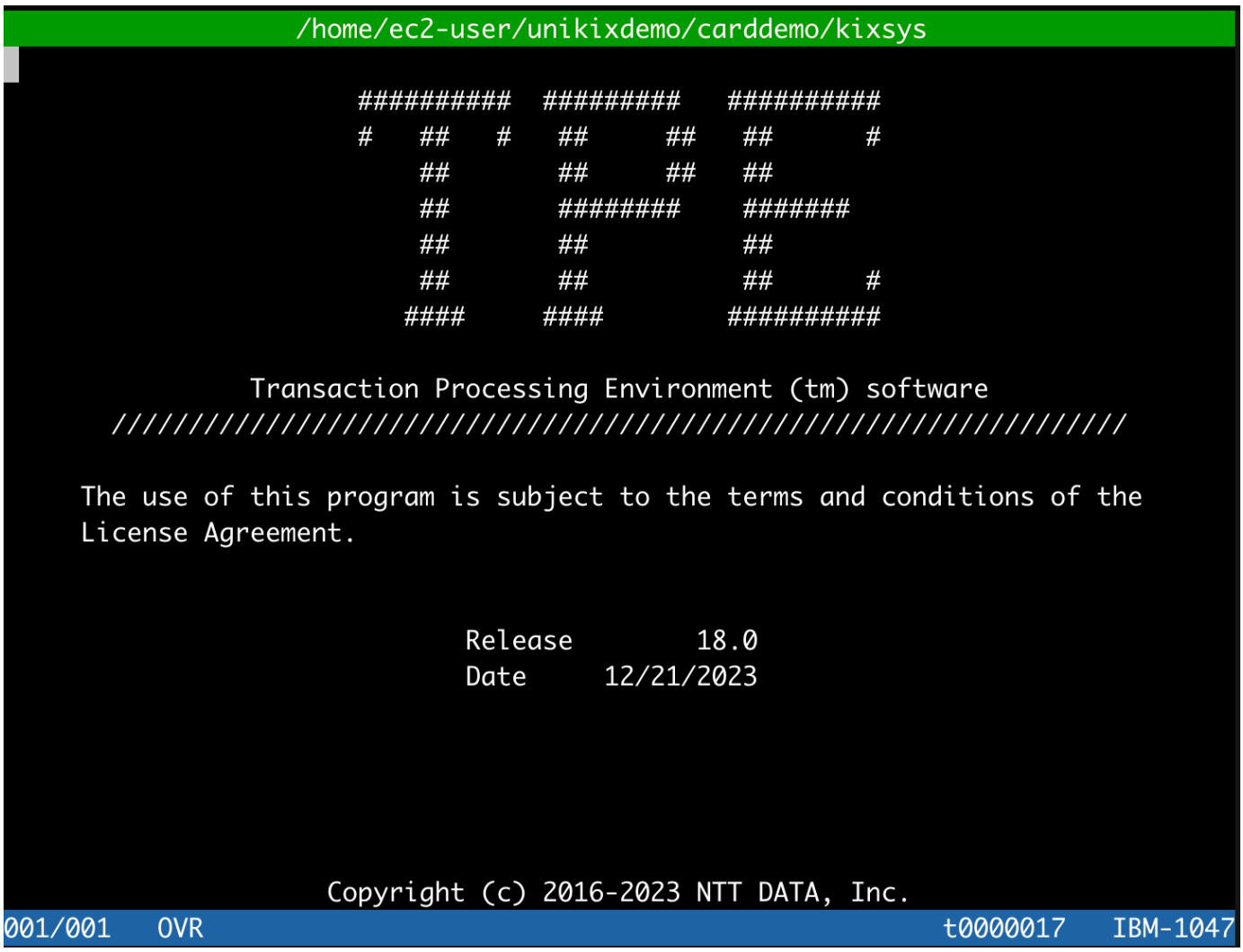
Au cours de cette étape, vous exécutez l' CardDemo application dans l'émulateur de terminal TN327 0.

L'AMI UniKix d'exécution est fournie avec un émulateur de terminal TN327 0 que vous pouvez lancer directement depuis l' UniKix EC2 instance.

Connectez-vous au TPE à l'aide d'un émulateur de terminal TN327 0

- Lancez le terminal TN327 0 à l'aide de la `kixterm` commande.

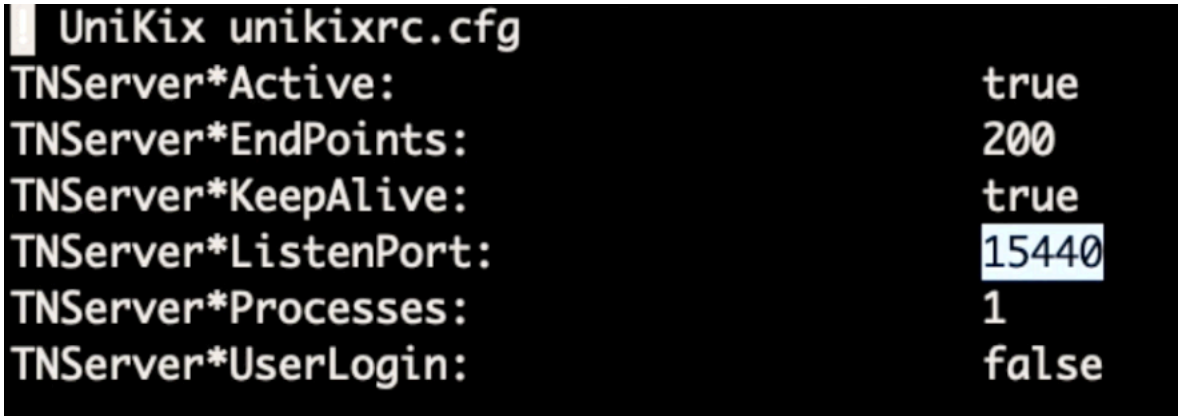
```
kixterm
```



(Facultatif) Si vous souhaitez utiliser votre propre émulateur de terminal :

1. Obtenez l'adresse IP de l'instance UniKix d'exécution depuis la EC2 console Amazon.
2. Obtenez le numéro de port pour la connexion à la région TPE à l'aide de l'émulateur de terminal TN327 0. Vous pouvez le trouver à TNServer ListenPort partir du fichier unikixrc.cfg.

```
cat $KIXSYS/unikixrc.cfg
```

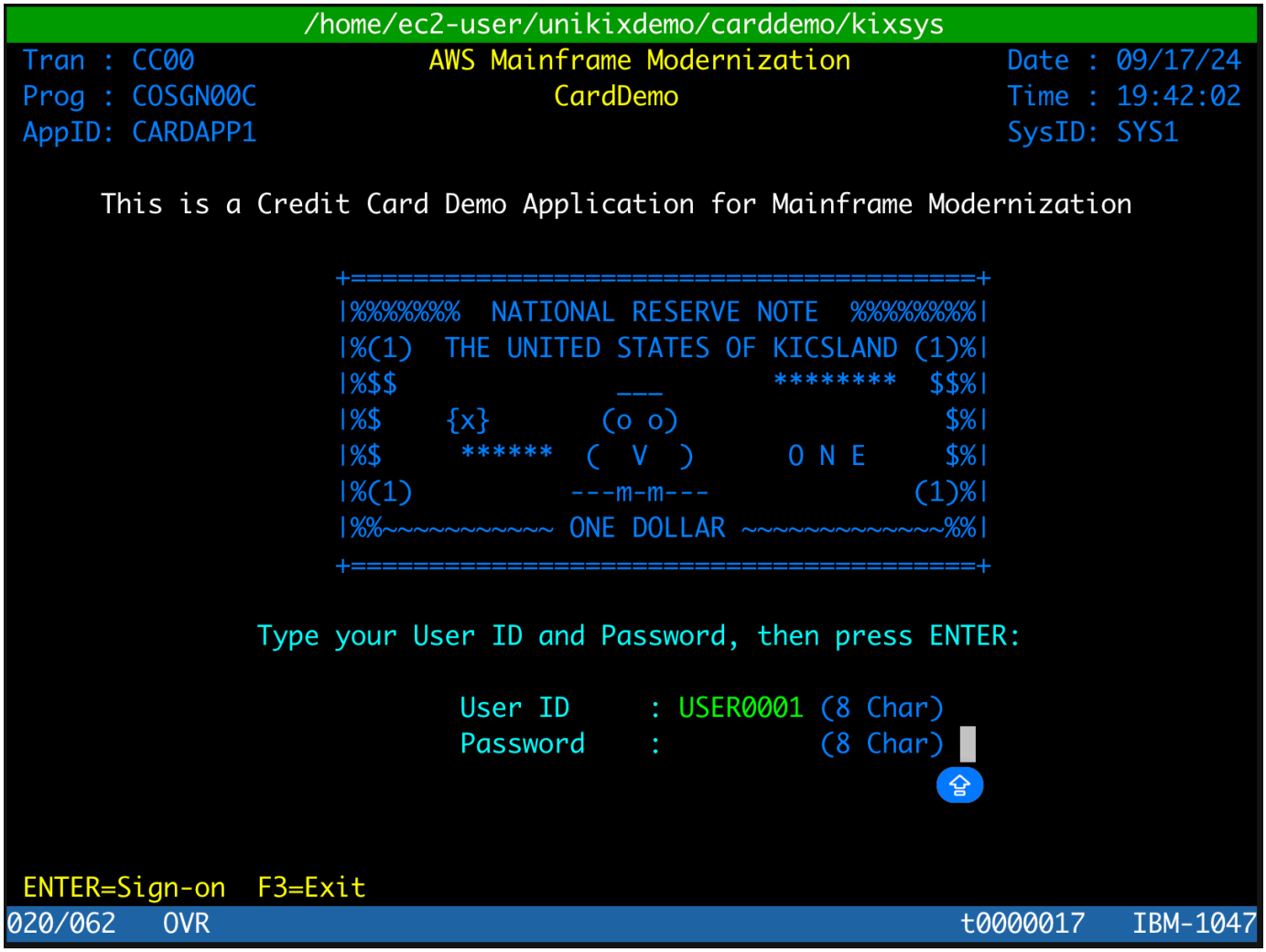


3. Configurez votre émulateur de terminal TN327 0 pour utiliser l'adresse IP de l'instance d' UniKix exécution et le numéro de port 15440.

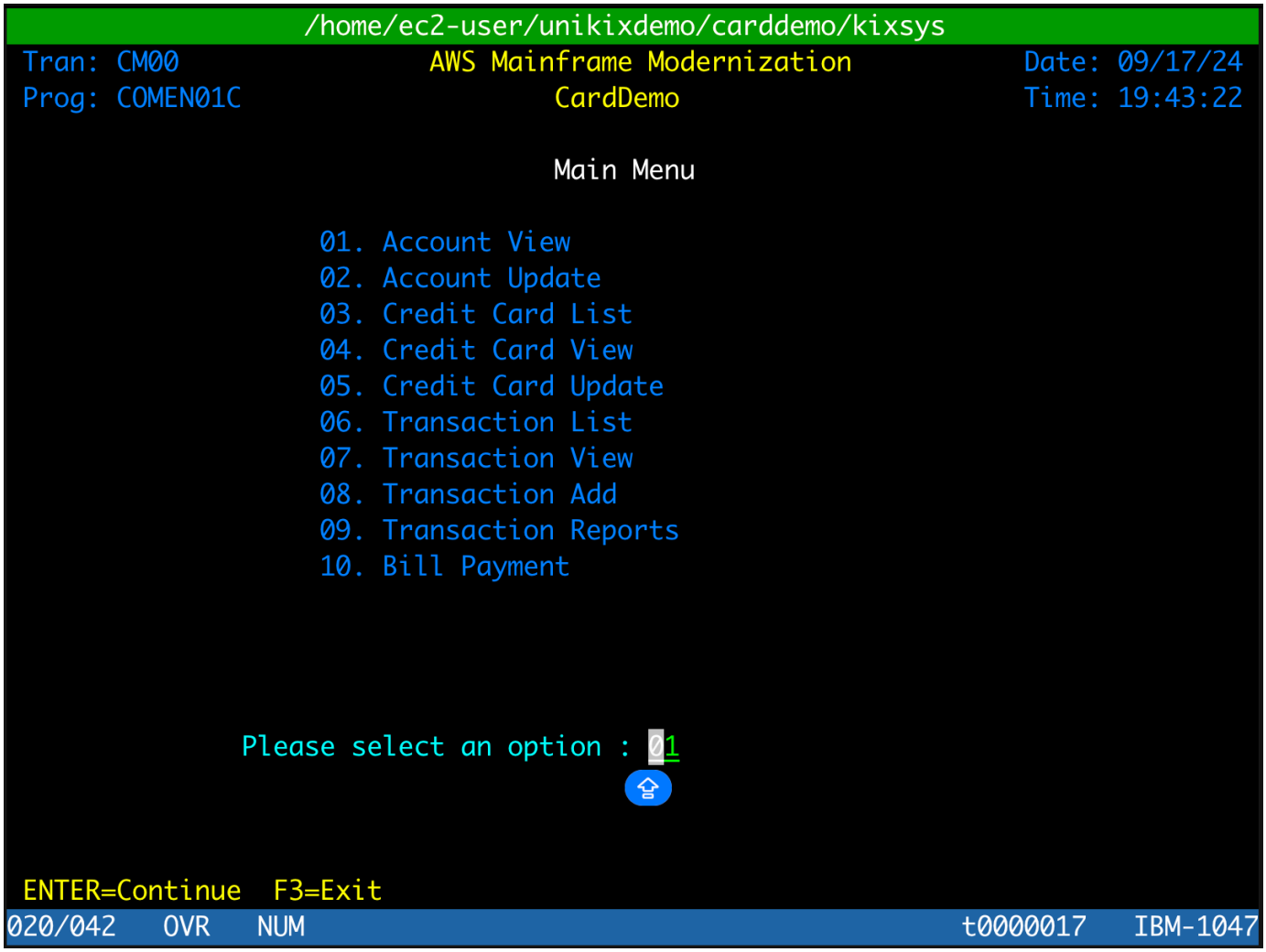
Transactions en ligne

Cette section suppose que vous vous êtes connecté à l'émulateur de terminal TN327 0 à l'aide de la `kixterm` commande.

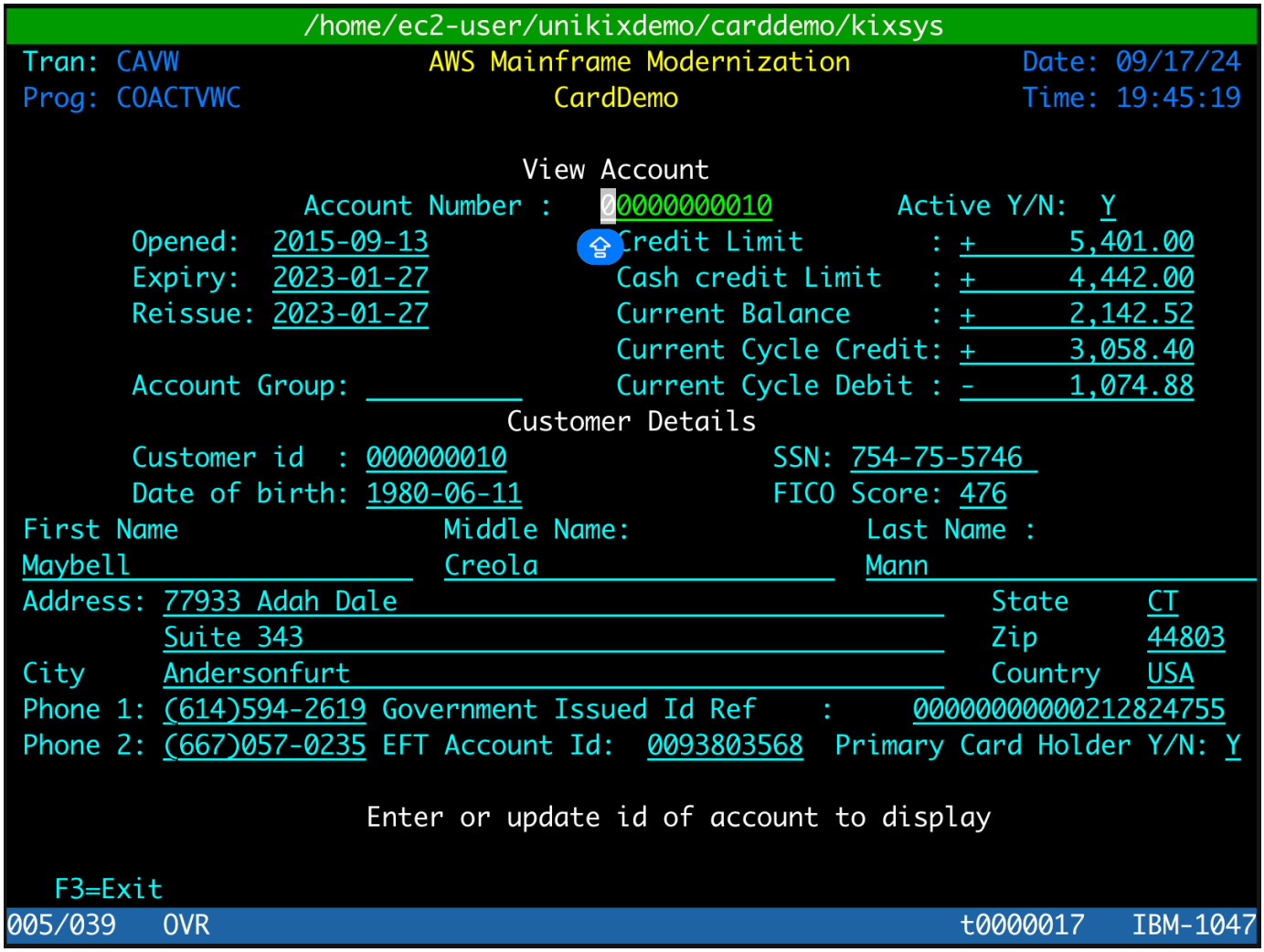
1. Après vous être connecté à partir de l'émulateur de terminal TN327 0, appuyez sur la touche « Entrée » pour effacer l'écran TPE et saisir la transaction initiale.
2. Sur la transaction initiale CC00 (écran de connexion), entrez le nom d'`USER001`utilisateur et le mot de `PASSWORD` passe.



3. Choisissez l'option « 01 » dans le menu principal pour afficher les comptes.



4. Dans l'écran Afficher le compte, entrez un numéro de compte (par exemple, 00000000010). Vous devriez voir les informations du compte renseignées à partir des données migrées.



5. Appuyez deux fois sur la touche « PF03 » pour revenir à l'écran de connexion et quittez le terminal TN327 0 en appuyant sur « Ctrl+C » (Windows) ou « Cmd+C » (Macbook).

Tâches par lots

1. Accédez au répertoire JCL.

```
cd $MBMSUB
```

2. Soumettez la tâche `MFCATGL1` et observez le résultat du journal des tâches.

```
BPESUB READCARD
```

3. Vous pouvez éventuellement consulter les journaux des tâches depuis le `$SUBSYS_OUTDIR` répertoire.

```
ls -lrt $SUBSYS_OUTDIR/*
```

Vous avez maintenant déployé avec succès l' CardDemo application sur l' UniKix environnement d'exécution NTT DATA et vérifié l'application en cours d'exécution en parcourant quelques écrans en ligne CICS et en effectuant des tâches par lots.

Résolution des problèmes

Voici quelques erreurs courantes que vous pouvez rencontrer lors de la configuration de l' CardDemo application.

Erreur : erreur de licence

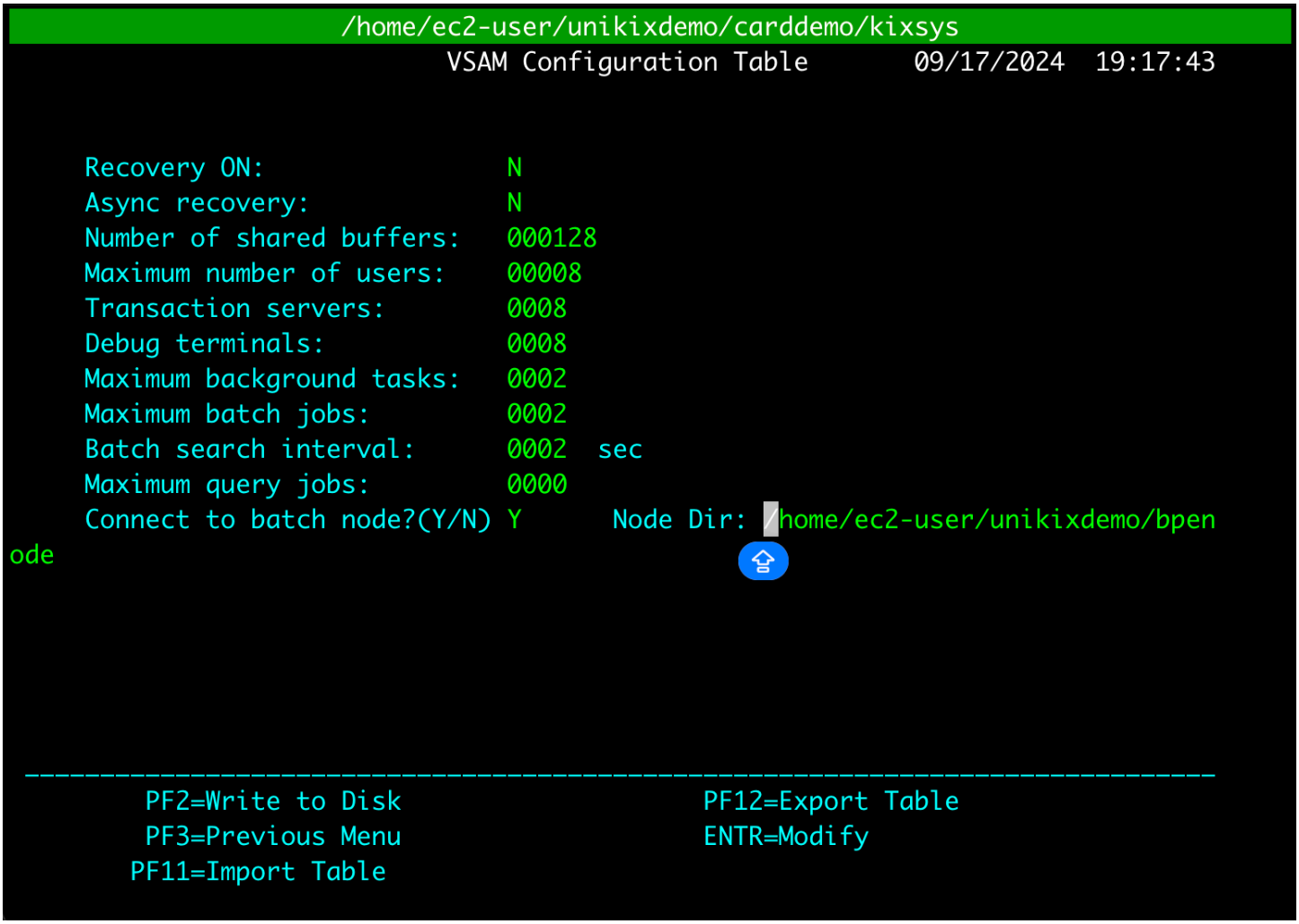
Si vous recevez un message d'erreur de licence lorsque vous suivez ce didacticiel, il se peut que le IMDSv2soit activé dans votre EC2. Vous pouvez résoudre ce problème en modifiant l'option IMDSv2des métadonnées de l'instance sur Facultative, comme indiqué dans la section Restaurer l'utilisation du IMDSv1 guide de EC2 l'utilisateur Amazon.

Erreur : le TPE n'est pas connecté au BPE

Si le TPE n'est pas connecté à BPE, assurez-vous que la table de configuration VSAM est correctement configurée avec le répertoire des nœuds BPE. Pour accéder à la table de configuration VSAM, lancez l'émulateur de terminal TN327 0 à l'aide de la commande suivante :

```
kixterm
```

1. Entrez le nom de la transaction en tant queCTBL.
2. Dans le menu du gestionnaire de tables, choisissez l'option Tableaux standard.
3. Sur l'écran des tables standard, choisissez l'option Table de configuration VSAM.
4. Vérifiez si Connect to batch node ? est défini sur « Y » et le répertoire des nœuds est correct.


/home/ec2-user/unikixdemo/carddemo/kixsys
VSAM Configuration Table 09/17/2024 19:17:43
Recovery ON: N
Async recovery: N
Number of shared buffers: 000128
Maximum number of users: 00008
Transaction servers: 0008
Debug terminals: 0008
Maximum background tasks: 0002
Maximum batch jobs: 0002
Batch search interval: 0002 sec
Maximum query jobs: 0000
Connect to batch node?(Y/N) Y Node Dir: /home/ec2-user/unikixdemo/bpen
ode
PF2=Write to Disk
PF3=Previous Menu
PF11=Import Table
PF12=Export Table
ENTR=Modify

La sécurité dans le cadre de la modernisation des AWS mainframes

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité applicables à la modernisation des AWS mainframes, consultez la section Services AWS concernés par programme de conformité .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris la sensibilité de vos données, les exigences de votre entreprise et la législation et la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation de la modernisation des AWS mainframes. Il vous explique comment configurer la modernisation du AWS mainframe pour atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser les ressources de modernisation de votre AWS mainframe.

AWS La modernisation du mainframe fournit ses propres ressources protégées par l'IAM (application, environnement, déploiement, etc.), qui sont les ressources administratives de modernisation du AWS mainframe, sur lesquelles toute action doit être autorisée par les politiques IAM.

AWS La modernisation du mainframe pour le replatforming est également sécurisée par IAM. L'IAM accorde ou refuse à un mandant l'autorisation d'effectuer une action spécifique sur une ressource définie, dérivée de l'environnement mainframe d'origine, également par le biais de politiques IAM standard. Le moteur d'exécution de replateforme AWS Mainframe Modernization appelle le service d'autorisation IAM lorsqu'une application tente une telle action sur une ressource protégée. IAM

renverra l'autorisation ou le refus en fonction des mécanismes d'évaluation des politiques IAM standard.

Table des matières

- Protection des données dans le cadre de la modernisation des AWS mainframes
- Identity and Access Management pour la modernisation des AWS mainframes
- Validation de conformité pour la AWS modernisation du mainframe
- Résilience dans la AWS modernisation des mainframes
- Sécurité de l'infrastructure dans AWS Mainframe Modernization
- Accès AWS Mainframe Modernization via un point de terminaison AWS PrivateLink d'interface

Protection des données dans le cadre de la modernisation des AWS mainframes

Le modèle de responsabilité AWS partagée Le modèle de s'applique à la protection des données dans le AWS cadre de la modernisation des ordinateurs centraux. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- SSL/TLS À utiliser pour communiquer avec AWS les ressources. Nous exigeons TLS 1.2 et recommandons TLS 1.3.

- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour capturer AWS des activités, consultez la section Utilisation des CloudTrail sentiers dans le guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3.

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Nom. Cela inclut lorsque vous travaillez avec AWS Mainframe Modernization ou autre à Services AWS l'aide de la console, de l'API ou AWS SDKs. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Données collectées par AWS Mainframe Modernization

AWS La modernisation du mainframe collecte plusieurs types de données auprès de vous :

- `Application configuration`: il s'agit d'un fichier JSON que vous créez pour configurer votre application. Il contient vos choix pour les différentes options proposées par AWS Mainframe Modernization. Le fichier contient également des informations relatives aux AWS ressources dépendantes, telles que les chemins Amazon Simple Storage Service où sont stockés les artefacts de l'application ou le nom de ressource Amazon (ARN) indiquant AWS Secrets Manager où sont stockées les informations d'identification de votre base de données.
- `Application executable (binary)`: il s'agit d'un fichier binaire que vous compilez et que vous avez l'intention de déployer dans le cadre de la modernisation AWS du mainframe.
- `Application JCL or scripts`: Ce code source gère les tâches par lots ou d'autres traitements pour le compte de votre application.

- `User application data`: Lorsque vous importez des ensembles de données, AWS Mainframe Modernization les stocke dans la base de données relationnelle afin que votre application puisse y accéder.
- `Application source code`: via Amazon AppStream 2.0, AWS Mainframe Modernization fournit un environnement de développement dans lequel vous pouvez écrire et compiler du code.

AWS La modernisation du mainframe stocke ces données de manière native dans. AWS Les données que nous collectons auprès de vous sont stockées dans un compartiment Amazon S3 géré par AWS Mainframe Modernization. Lorsque vous déployez une application, AWS Mainframe Modernization télécharge les données sur une instance Amazon Elastic Compute Cloud basée sur Amazon Elastic Block Store. Lorsque le nettoyage est déclenché, les données sont supprimées du volume Amazon EBS et d'Amazon S3. Les volumes Amazon EBS sont réservés à un locataire unique, ce qui signifie qu'une instance est utilisée pour un client. Les instances ne sont jamais partagées. Lorsque vous supprimez un environnement d'exécution, le volume Amazon EBS est également supprimé. Lorsque vous supprimez une application, les artefacts et la configuration sont supprimés d'Amazon S3.

Les journaux des applications sont stockés sur Amazon CloudWatch. Les messages du journal des applications clientes sont également CloudWatch exportés vers. Les CloudWatch journaux peuvent contenir des données sensibles pour le client, telles que des données commerciales ou des informations de sécurité (dans les messages de débogage). Pour de plus amples informations, veuillez consulter Surveillance de la modernisation des AWS mainframes avec Amazon CloudWatch.

En outre, si vous choisissez d'associer un ou plusieurs systèmes de fichiers Amazon Elastic FSx File System ou Amazon à votre environnement d'exécution, les données de ces systèmes seront stockées dans AWS. Vous devrez nettoyer ces données si vous décidez de ne plus utiliser les systèmes de fichiers.

Vous pouvez utiliser toutes les options de chiffrement Amazon S3 disponibles pour sécuriser vos données lorsque vous les placez dans le compartiment Amazon S3 utilisé par AWS Mainframe Modernization pour le déploiement d'applications et les importations de jeux de données. En outre, vous pouvez utiliser les options de FSx chiffrement Amazon EFS et Amazon si vous associez un ou plusieurs de ces systèmes de fichiers à votre environnement d'exécution.

Chiffrement des données arrêté pour le service de modernisation des AWS mainframes

AWS La modernisation du mainframe s'intègre AWS Key Management Service pour fournir un chiffrement transparent côté serveur (SSE) sur toutes les ressources dépendantes qui stockent les données de façon permanente, à savoir Amazon Simple Storage Service, Amazon DynamoDB et Amazon Elastic Block Store. AWS La modernisation du mainframe crée et gère des AWS KMS clés de chiffrement symétriques pour vous. AWS KMS

Le chiffrement des données au repos par défaut permet de réduire les frais opérationnels et la complexité liés à la protection des données sensibles. Dans le même temps, il vous permet de migrer des applications qui nécessitent une conformité stricte en matière de chiffrement et des exigences réglementaires.

Vous ne pouvez pas désactiver cette couche de chiffrement ni sélectionner un autre type de chiffrement lorsque vous créez des environnements d'exécution et des applications.

Vous pouvez utiliser votre propre clé gérée par le client pour les applications de modernisation des AWS mainframes et les environnements d'exécution afin de chiffrer les ressources Amazon S3 et Amazon EBS.

Pour vos applications de modernisation du AWS mainframe, vous pouvez utiliser cette clé pour chiffrer la définition de votre application ainsi que d'autres ressources applicatives, telles que les fichiers JCL, qui sont enregistrées dans le compartiment Amazon S3 créé dans le compte du service. Pour de plus amples informations, veuillez consulter Création d'une application .

Pour vos environnements d'exécution AWS Mainframe Modernization, AWS Mainframe Modernization utilise votre clé gérée par le client pour chiffrer le volume Amazon EBS qu'il crée et attache à votre EC2 instance Amazon Mainframe AWS Modernization, qui figure également dans le compte du service. Pour de plus amples informations, veuillez consulter Création d'un environnement d'exécution.

> **Note**
>
> Les ressources DynamoDB sont toujours chiffrées à l'aide d' Clé gérée par AWS un compte de service intégré au Mainframe AWS Modernization. Vous ne pouvez pas chiffrer les ressources DynamoDB à l'aide d'une clé gérée par le client.

AWS La modernisation du mainframe utilise votre clé gérée par le client pour les tâches suivantes :

- Redéploiement d'une application.
- Remplacement d'une EC2 instance Amazon de modernisation du AWS mainframe.

AWS Mainframe Modernization n'utilise pas votre clé gérée par le client pour chiffrer les bases de données Amazon Relational Database Service ou Amazon Aurora, les files d'attente Amazon Simple Queue Service et les caches ElastiCache Amazon créés pour prendre en charge AWS une application de modernisation du mainframe, car aucun d'entre eux ne contient de données client.

Pour plus d'informations, consultez Clés gérées par le client dans le Guide du développeur AWS Key Management Service (langue française non garantie).

Le tableau suivant résume la façon dont la modernisation AWS du mainframe chiffre vos données sensibles.

Type de données	Clé gérée par AWS chiffrement	Chiffrement des clés géré par le client
`Definition` Contient la définition d'une application particulière.	Activées	Activées
`EnvironmentSummary` Contient des informations sur l'environnement d'exécution.	Activées	Activées
`ApplicationSummary` Contient des informations sur l'application AWS Mainframe Modernization.	Activées	Activées
`DeploymentSummary` Contient des informations sur le déploiement d'une applicati	Activées	Activées

Type de données	Clé gérée par AWS chiffrement	Chiffrement des clés géré par le client
on de modernisation AWS du mainframe.		

> **Note**
>
> AWS La modernisation du mainframe active automatiquement le chiffrement au repos Clés gérées par AWS afin de protéger gratuitement vos données sensibles. Toutefois, AWS KMS des frais s'appliquent pour l'utilisation d'une clé gérée par le client. Pour plus d'informations sur la tarification, consultez Tarification d'AWS Key Management Service.

Pour plus d'informations sur AWS KMS, voir AWS Key Management Service.

Comment la modernisation AWS du mainframe utilise les subventions dans AWS KMS

AWS La modernisation du mainframe nécessite une subvention pour utiliser votre clé gérée par le client.

Lorsque vous créez une application ou un environnement d'exécution, ou que vous déployez une application dans AWS Mainframe Modernization chiffrée à l'aide d'une clé gérée par le client, AWS Mainframe Modernization crée une subvention en votre nom en envoyant une CreateGrantdemande à. AWS KMS Les subventions AWS KMS sont utilisées pour donner à AWS Mainframe Modernization l'accès à une clé KMS dans un compte client.

AWS La modernisation du mainframe nécessite l'autorisation d'utiliser votre clé gérée par le client pour les opérations internes suivantes :

- Envoyez DescribeKeydes demandes AWS KMS à pour vérifier que l'ID de clé symétrique géré par le client saisi lors de la création d'une application, d'un environnement d'exécution ou du déploiement d'une application est valide.
- Envoyez GenerateDataKeydes demandes AWS KMS à chiffrer le volume Amazon EBS attaché aux EC2 instances Amazon hébergeant les environnements d'exécution AWS Mainframe Modernization.

- Envoyez des demandes de déchiffrement AWS KMS à pour déchiffrer le contenu chiffré sur Amazon EBS.

AWS La modernisation du mainframe utilise des AWS KMS subventions pour déchiffrer vos secrets stockés dans Secrets Manager et lors de la création d'un environnement d'exécution, de la création ou du redéploiement d'une application et de la création d'un déploiement. Les subventions créées par la modernisation AWS du mainframe soutiennent les opérations suivantes :

- Créez ou mettez à jour une subvention d'environnement d'exécution :
  - Decrypt (Déchiffrer)
  - Encrypt (Chiffrer)
  - ReEncryptFrom
  - ReEncryptTo
  - GenerateDataKey
  - DescribeKey
  - CreateGrant
- Créez ou redéployez une subvention de candidature :
  - GenerateDataKey
- Créez une subvention de déploiement :
  - Decrypt

Vous pouvez révoquer l'accès à l'octroi ou supprimer l'accès du service à la clé gérée par le client à tout moment. Si vous le faites, AWS Mainframe Modernization ne pourra accéder à aucune des données chiffrées par la clé gérée par le client, ce qui affectera les opérations qui dépendent de ces données. Par exemple, si AWS Mainframe Modernization essayait d'accéder à une définition d'application chiffrée par une clé gérée par le client sans accorder cette clé, l'opération de création de l'application échouerait.

AWS Mainframe Modernization collecte les configurations des applications utilisateur (fichiers JSON) et les artefacts (fichiers binaires et exécutables). Il crée également des métadonnées qui suivent les différentes entités utilisées pour le fonctionnement de la modernisation du AWS mainframe, et crée des journaux et des métriques. Les journaux et statistiques visibles par le client incluent :

- CloudWatch journaux qui reflètent l'application et le moteur d'exécution (AWS Blu Age ou Rocket Software (anciennement Micro Focus)).

- CloudWatch métriques pour les tableaux de bord des opérations.

En outre, AWS Mainframe Modernization collecte des données d'utilisation et des mesures pour les mesures, les rapports d'activité, etc. concernant les services. Ces données ne sont pas visibles par le client.

AWS La modernisation du mainframe stocke ces données à différents endroits en fonction du type de données. Les données client que vous chargez sont stockées dans un compartiment Amazon S3. Les données de service sont stockées à la fois dans Amazon S3 et DynamoDB. Lorsque vous déployez une application, vos données et les données de service sont téléchargées sur les volumes Amazon EBS. Si vous choisissez d'associer le FSx stockage Amazon EFS ou Amazon à votre environnement d'exécution, les données stockées dans ces systèmes de fichiers sont également téléchargées sur le volume Amazon EBS.

Le chiffrement au repos est configuré par défaut. Vous ne pouvez ni le désactiver ni le modifier. Actuellement, vous ne pouvez pas non plus modifier sa configuration.

Création d'une clé gérée par le client

Vous pouvez créer une clé symétrique gérée par le client en utilisant le AWS Management Console ou le AWS KMS APIs.

Pour créer une clé symétrique gérée par le client

Suivez les étapes de la rubrique Création d'une clé symétrique gérée par le client dans le Guide du développeur AWS Key Management Service .

Stratégie de clé

Les stratégies de clé contrôlent l'accès à votre clé gérée par le client. Chaque clé gérée par le client doit avoir exactement une stratégie de clé, qui contient des instructions qui déterminent les personnes pouvant utiliser la clé et comment elles peuvent l'utiliser. Lorsque vous créez votre clé gérée par le client, vous pouvez spécifier une stratégie de clé. Pour plus d'informations, consultez Gestion de l'accès aux clés gérées par le client dans le Guide du développeur AWS Key Management Service .

Pour utiliser votre clé gérée par le client avec vos ressources de modernisation du AWS mainframe, les opérations d'API suivantes doivent être autorisées dans la politique clé :

- `kms:CreateGrant` : ajoute une attribution à une clé gérée par le client. Accorde un accès de contrôle à une clé KMS spécifiée, ce qui permet d'accéder aux opérations d'octroi requises par la modernisation AWS du mainframe. Pour plus d'informations sur l'utilisation des subventions, consultez le guide du AWS Key Management Service développeur.

  Cela permet à AWS Mainframe Modernisation d'effectuer les opérations suivantes :

  - Appelez `GenerateDataKey` pour générer une clé de données chiffrée et la stocker, car la clé de données n'est pas immédiatement utilisée pour chiffrer.
  - Appelez `Decrypt` pour utiliser la clé de données chiffrée stockée afin d'accéder aux données chiffrées.
  - Configurez un directeur partant à la retraite pour permettre au service de`RetireGrant`.
- `kms:DescribeKey`— Fournit les informations clés gérées par le client pour permettre à AWS Mainframe Modernization de valider la clé.

AWS La modernisation du mainframe nécessite `kms:CreateGrant` des `kms:DescribeKey` autorisations et des autorisations dans la politique clé du client. AWS La modernisation du mainframe utilise cette politique pour créer une subvention pour elle-même.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "Enable IAM User Permissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::AccountId:role/ExampleRole"
        },
        "Action": [
            "kms:CreateGrant",
            "kms:DescribeKey"
        ],
        "Resource": "*"
    }]
}
```

> **Note**
>
> Le rôle illustré `Principal` dans l'exemple précédent est celui que vous utilisez pour les opérations de modernisation AWS du mainframe telles que `CreateApplication` et`CreateEnvironment`.

Pour plus d'informations sur la spécification d'autorisations dans une politique, consultez le Guide du développeur AWS Key Management Service .

Pour plus d'informations sur le dépannage des clés d'accès, consultez le Guide du développeur AWS Key Management Service .

Spécification d'une clé gérée par le client pour la modernisation AWS du mainframe

Vous pouvez spécifier une clé gérée par le client pour les ressources suivantes :

- Application
- Environnement

Lorsque vous créez une ressource, vous pouvez spécifier la clé en saisissant un ID KMS, que AWS Mainframe Modernization utilise pour chiffrer les données sensibles stockées par la ressource.

- ID KMS : identifiant de clé pour une clé gérée par le client. Saisissez un ID de clé, un ARN de clé, un nom d'alias ou un ARN d'alias.

Vous pouvez spécifier une clé gérée par le client à l'aide du AWS Management Console ou du AWS CLI.

Pour spécifier votre clé gérée par le client lors de la création d'un environnement d'exécution dans le AWS Management Console, voirCréation d'un environnement d'exécution pour la modernisation du AWS mainframe. Pour spécifier votre clé gérée par le client lors de la création d'une application dans le AWS Management Console, voirCréation d'une AWS Mainframe Modernization application.

Pour ajouter votre clé gérée par le client lorsque vous créez un environnement d'exécution avec le AWS CLI, spécifiez le `kms-key-id` paramètre comme suit :

```
aws m2 create-environment –engine-type microfocus –instance-type M2.m5.large
--publicly-accessible –engine-version 7.0.3 –name test
--high-availability-config desiredCapacity=2
--kms-key-id myEnvironmentKey
```

Pour ajouter votre clé gérée par le client lorsque vous créez une application avec le AWS CLI, spécifiez le `kms-key-id` paramètre comme suit :

```
aws m2 create-application –name test-application –description my description
--engine-type microfocus
--definition content="$(jq -c . raw-template.json | jq -R)"
--kms-key-id myApplicationKey
```

AWS Contexte de chiffrement de la modernisation du mainframe

Un contexte de chiffrement est un ensemble facultatif de paires clé-valeur qui contient des informations contextuelles supplémentaires sur les données.

AWS KMS utilise le contexte de chiffrement comme données authentifiées supplémentaires pour prendre en charge le chiffrement authentifié. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, AWS KMS lie le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez inclure le même contexte de chiffrement dans la demande.

AWS Contexte de chiffrement de la modernisation du mainframe

AWS La modernisation du mainframe utilise le même contexte de chiffrement dans toutes les opérations AWS KMS cryptographiques liées à une application (création d'une application et création d'un déploiement), où la clé se trouve `aws:m2:app` et la valeur est l'identifiant unique de l'application.

Example

```
"encryptionContextSubset": {
        "aws:m2:app": "a1bc2defabc3defabc4defabcd"
}
```

Utilisation du contexte de chiffrement pour la surveillance

Lorsque vous utilisez une clé symétrique gérée par le client pour chiffrer vos applications ou vos environnements d'exécution, vous pouvez également utiliser le contexte de chiffrement dans les

enregistrements d'audit et les journaux pour identifier la manière dont la clé gérée par le client est utilisée.

Utilisation du contexte de chiffrement pour contrôler l'accès à votre clé gérée par le client

Vous pouvez utiliser le contexte de chiffrement dans les stratégies de clé et les politiques IAM comme `conditions` pour contrôler l'accès à votre clé symétrique gérée par le client. Vous pouvez également utiliser des contraintes de contexte de chiffrement dans un octroi.

AWS La modernisation du mainframe utilise une contrainte de contexte de chiffrement dans les autorisations afin de contrôler l'accès à la clé gérée par le client dans votre compte ou votre région. La contrainte d'octroi exige que les opérations autorisées par l'octroi utilisent le contexte de chiffrement spécifié. L'exemple suivant est une subvention que AWS Mainframe Modernization utilise pour chiffrer un artefact d'application lors de la création d'une application.

```
//This grant is retired immediately after create application finish
{
   "grantee-principal": m2.us-west-2.amazonaws.com,
   "retiring-principal": m2.us-west-2.amazonaws.com,
   "operations": [
       "GenerateDataKey"
   ]
   "condition": {
        "encryptionContextSubset": {
            “aws:m2:app”: “a1bc2defabc3defabc4defabcd”
   }
}
```

Surveillance de vos clés de chiffrement pour la modernisation AWS du mainframe

Lorsque vous utilisez une clé gérée par le AWS KMS client avec vos ressources de modernisation du AWS mainframe, vous pouvez utiliser AWS CloudTrailAmazon CloudWatch Logs pour suivre les demandes envoyées par AWS Mainframe Modernization. AWS KMS

Exemples d'environnements d'exécution

Les exemples suivants sont des AWS CloudTrail événements destinés à`DescribeKey`, `CreateGrantGenerateDataKey`, et `Decrypt` à surveiller les opérations KMS appelés par AWS Mainframe Modernization pour accéder aux données chiffrées par votre clé gérée par le client :

DescribeKey

AWS La modernisation du mainframe utilise cette `DescribeKey` opération pour vérifier si la clé gérée par le AWS KMS client associée à votre environnement d'exécution existe dans le compte et dans la région.

L'exemple d'événement suivant enregistre l'opération `DescribeKey` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T19:40:26Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:43Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.182",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
```

```
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

CreateGrant

Lorsque vous utilisez une clé gérée par le AWS KMS client pour chiffrer votre environnement d'exécution, AWS Mainframe Modernization envoie plusieurs `CreateGrant` demandes en votre nom pour effectuer les opérations KMS nécessaires. Certaines des subventions créées par AWS Mainframe Modernization sont supprimées immédiatement après leur utilisation. Les autres sont retirés lorsque vous supprimez l'environnement d'exécution.

L'exemple d'événement suivant enregistre l'`CreateGrant`opération pour le rôle d'exécution Lambda associé au flux de travail Create Environment.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
```

```
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:11:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
```

```
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

L'exemple d'événement suivant enregistre l'`CreateGrant`opération pour le rôle lié au service du groupe Auto Scaling. Le rôle d'exécution Lambda associé au flux de travail Create Environment appelle cette `CreateGrant` opération. Il autorise le rôle d'exécution à créer une sous-subvention pour le rôle lié au service du groupe Auto Scaling.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65MZFUPM4JO:EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
        "arn": "arn:aws:sts::111122223333:assumed-role/EnvironmentWorkflow-
alpha-CreateEnvironmentLambdaS-1AU4A8VNQEEKN/EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN",
                "accountId": "111122223333",
                "userName": "EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:22:28Z",
```

```
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "54.148.236.160",
    "userAgent": "aws-sdk-java/2.18.21 Linux/4.14.255-276-224.499.amzn2.x86_64
 OpenJDK_64-Bit_Server_VM/11.0.14.1+10-LTS Java/11.0.14.1 vendor/Amazon.com_Inc. md/
internal exec-env/AWS_Lambda_java11 io/sync http/Apache cfg/retry-mode/legacy",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
```

```
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
}
```

GenerateDataKey

Lorsque vous activez une clé gérée par le AWS KMS client pour votre ressource d'environnement d'exécution, Auto Scaling crée une clé unique pour chiffrer le volume Amazon EBS associé à l'environnement d'exécution. Il envoie une `GenerateDataKey` demande AWS KMS qui spécifie la clé gérée par le AWS KMS client pour la ressource.

L'exemple d'événement suivant enregistre l'opération `GenerateDataKey` :

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65EEXVIEH7D:AutoScaling",
        "arn": "arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForAutoScaling/
AutoScaling",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAutoScaling"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:23:16Z",
```

```
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "autoscaling.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:18Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "autoscaling.amazonaws.com",
    "userAgent": "autoscaling.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        },
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "numberOfBytes": 64
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


Decrypt

Lorsque vous accédez à un environnement d'exécution chiffré, Amazon EBS appelle l'`Decrypt`opération pour utiliser la clé de données cryptée stockée afin d'accéder aux données chiffrées.

L'exemple d'événement suivant enregistre l'opération `Decrypt` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ebs.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:22Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ebs.amazonaws.com",
    "userAgent": "ebs.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
}
```

Exemples d'applications

Les exemples suivants sont des AWS CloudTrail événements pour `CreateGrant` et `GenerateDataKey` pour surveiller les opérations KMS appelés par AWS Mainframe Modernization afin d'accéder aux données chiffrées par votre clé gérée par le client :

CreateGrant

Lorsque vous utilisez une clé gérée par le AWS KMS client pour chiffrer les ressources de votre application, le rôle d'exécution Lambda envoie `CreateGrant` une demande en votre nom pour accéder à la clé KMS de votre compte. AWS La subvention permet au rôle d'exécution Lambda de télécharger les ressources de l'application client sur Amazon S3 à l'aide de votre clé gérée par le client. Cette subvention est retirée immédiatement après la création de la demande.

L'exemple d'événement suivant enregistre l'opération `CreateGrant` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T22:47:04Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
```

```
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
        "operations": [
            "GenerateDataKey"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


GenerateDataKey

Lorsque vous activez une clé gérée par le AWS KMS client pour votre ressource d'application, le rôle d'exécution Lambda crée une clé qu'il utilise pour chiffrer et télécharger les données

des clients sur Amazon Simple Storage Service. Le rôle d'exécution Lambda envoie une `GenerateDataKey` demande AWS KMS qui spécifie la clé gérée par le AWS KMS client pour la ressource.

L'exemple d'événement suivant enregistre l'opération `GenerateDataKey` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65CLCEKKC7Z:ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "arn": "arn:aws:sts::111122223333:assumed-role/ApplicationWorkflow-
alpha-CreateApplicationVersion-1IZRBZYDG20B/ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B",
                "accountId": "111122223333",
                "userName": "ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:28:32Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:29:08Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
```

```
            "aws:m2:app": "a1bc2defabc3defabc4defabcd",
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcd/1/cics-transaction/ZBNKE35.so"
        },
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Exemples de déploiements

Les exemples suivants sont des AWS CloudTrail événements pour `CreateGrant` et `Decrypt` pour surveiller les opérations KMS appelés par AWS Mainframe Modernization afin d'accéder aux données chiffrées par votre clé gérée par le client :

CreateGrant

Lorsque vous utilisez une clé gérée par le AWS KMS client pour chiffrer vos ressources de déploiement, AWS Mainframe Modernization envoie deux `CreateGrant` demandes en votre nom. La première subvention est liée au rôle d'exécution Lambda actuel à appeler ListBatchJobScriptFiles et est retirée immédiatement après la fin du déploiement. La deuxième subvention va à l'encontre du rôle EC2 d'instance défini par Amazon afin qu'Amazon EC2 puisse télécharger les ressources des applications clients depuis Amazon S3. Cette autorisation est retirée lorsque l'application est supprimée de l'environnement d'exécution.

L'exemple d'événement suivant enregistre l'opération `CreateGrant` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:40:07Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
```

```
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Lorsque vous accédez à un déploiement, Amazon EC2 appelle l'`Decrypt`opération pour utiliser la clé de données cryptée stockée afin de déchiffrer et de télécharger les données client chiffrées depuis Amazon S3.

L'exemple d'événement suivant enregistre l'opération `Decrypt` :

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65BSPZ37E6G:m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "arn": "arn:aws:sts::111122223333:assumed-role/
SupernovaEnvironmentInstanceScopeDownRole/m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
```

```
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/
SupernovaEnvironmentInstanceScopeDownRole",
                "accountId": "111122223333",
                "userName": "SupernovaEnvironmentInstanceScopeDownRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:19:29Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:40:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:m2:app": "a1bc2defabc3defabc4defabcdm",
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcdm/1/cics-transaction/BBANK40P.so"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
```

```
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

En savoir plus

Les ressources suivantes fournissent plus d'informations sur le chiffrement des données au repos.

- Pour plus d'informations sur les concepts de base AWS Key Management Service, consultez le Guide du développeur AWS Key Management Service .
- Pour plus d'informations sur les Bonnes pratiques de sécurité pour AWS Key Management Service, consultez le Guide du développeur AWS Key Management Service .

Chiffrement en transit

Pour les applications interactives faisant partie de charges de travail transactionnelles, les échanges de données entre l'émulateur de terminal et le protocole AWS Mainframe Modernization Service Endpoint for TN327 0 ne sont pas chiffrés en transit. Si l'application nécessite un chiffrement pendant le transit, vous souhaiterez peut-être implémenter des mécanismes de tunneling supplémentaires.

AWS La modernisation du mainframe utilise le protocole HTTPS pour chiffrer le service. APIs Toutes les autres communications au sein de AWS Mainframe Modernization sont protégées par le VPC de service ou le groupe de sécurité, ainsi que par le protocole HTTPS. AWS La modernisation du mainframe transfère les artefacts, les configurations et les données des applications. Les artefacts d'application sont copiés depuis un compartiment Amazon S3 que vous possédez, tout comme les données d'application. Vous pouvez fournir des configurations d'applications à l'aide d'un lien vers Amazon S3 ou en téléchargeant un fichier localement.

Le chiffrement de base en transit est configuré par défaut, mais ne s'applique pas au protocole TN327 0. AWS La modernisation du mainframe utilise le protocole HTTPS pour les points de terminaison des API, qui sont également configurés par défaut.

Identity and Access Management pour la modernisation des AWS mainframes

AWS Identity and Access Management (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources de modernisation du AWS mainframe. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Rubriques

- Public ciblé
- Authentification par des identités
- Gestion des accès à l'aide de politiques
- Comment fonctionne la modernisation AWS du mainframe avec IAM
- Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe
- Résolution des problèmes liés à la modernisation AWS du mainframe (identité et accès)
- Utilisation des rôles liés aux services pour AWS Mainframe Modernization

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez dans le cadre de la modernisation du AWS mainframe.

Utilisateur du service : si vous utilisez le service AWS Mainframe Modernization pour effectuer votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Au fur et à mesure que vous utilisez de plus en plus de fonctionnalités de modernisation du AWS mainframe pour effectuer votre travail, vous aurez peut-être besoin d'autorisations supplémentaires. En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur. Si vous ne parvenez pas à accéder à une fonctionnalité dans AWS Mainframe Modernization, consultezRésolution des problèmes liés à la modernisation AWS du mainframe (identité et accès).

Administrateur de service — Si vous êtes responsable des ressources de modernisation des AWS mainframes dans votre entreprise, vous avez probablement un accès complet à la modernisation des AWS mainframes. C'est à vous de déterminer les fonctionnalités et ressources de modernisation

du AWS mainframe auxquelles les utilisateurs du service doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM. Pour en savoir plus sur la manière dont votre entreprise peut utiliser l'IAM avec la modernisation du AWS mainframe, consultez. Comment fonctionne la modernisation AWS du mainframe avec IAM

Administrateur IAM : si vous êtes administrateur IAM, vous souhaiterez peut-être en savoir plus sur la manière dont vous pouvez rédiger des politiques pour gérer l'accès à la modernisation des AWS mainframes. Pour consulter des exemples de politiques basées sur l'identité de modernisation du AWS mainframe que vous pouvez utiliser dans IAM, consultez. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié (connecté à AWS) en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en AWS tant qu'identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une identité fédérée, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS l'aide de la fédération, vous assumez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter au portail AWS Management Console ou au portail AWS d'accès. Pour plus d'informations sur la connexion à AWS, consultez la section Comment vous connecter à votre compte Compte AWS dans le guide de Connexion à AWS l'utilisateur.

Si vous y accédez AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes à l'aide de vos informations d'identification. Si vous n'utilisez pas d' AWS outils, vous devez signer vous-même les demandes. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer des demandes vous-même, consultez AWS Signature Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, il vous AWS recommande d'utiliser l'authentification multifactorielle (MFA) pour renforcer la sécurité de votre compte. Pour plus d'informations, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center et Authentification multifactorielle AWS dans IAM dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur racine, consultez Tâches nécessitant des informations d'identification d'utilisateur racine dans le Guide de l'utilisateur IAM.

Identité fédérée

La meilleure pratique consiste à obliger les utilisateurs humains, y compris ceux qui ont besoin d'un accès administrateur, à utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide Services AWS d'informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, d'un fournisseur d'identité Web AWS Directory Service, du répertoire Identity Center ou de tout utilisateur qui y accède à l'aide des informations d'identification fournies Services AWS par le biais d'une source d'identité. Lorsque des identités fédérées y accèdent Comptes AWS, elles assument des rôles, qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous pouvez vous connecter et synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre source d'identité afin de les utiliser dans toutes vos applications Comptes AWS et applications. Pour obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou une seule application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme telles que des mots de passe et des clés d'accès. Toutefois, si certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM, nous vous recommandons d'effectuer une rotation des clés d'accès. Pour plus d'informations, consultez Rotation régulière des clés d'accès pour les cas d'utilisation nécessitant des informations d'identification dans le Guide de l'utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez nommer un groupe IAMAdminset lui donner les autorisations nécessaires pour administrer les ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Cas d'utilisation pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Pour assumer temporairement un rôle IAM dans le AWS Management Console, vous pouvez passer d'un rôle d'utilisateur à un rôle IAM (console). Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré : pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour

obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.
  - Rôle de service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.
  - Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés

à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une EC2 instance et qui envoient des demandes AWS CLI d' AWS API. Cela est préférable au stockage des clés d'accès dans l' EC2 instance. Pour attribuer un AWS rôle à une EC2 instance et le rendre disponible pour toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes exécutés sur l' EC2 instance d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utiliser un rôle IAM pour accorder des autorisations aux applications exécutées sur des EC2 instances Amazon dans le guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur root ou session de rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur la structure et le contenu des documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui autorise l'action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre Compte AWS. Les politiques gérées incluent les politiques AWS gérées et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d'accès (ACLs)

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et AWS WAF Amazon VPC sont des exemples de services compatibles. ACLs Pour en savoir plus ACLs, consultez la présentation de la liste de contrôle d'accès (ACL) dans le manuel Amazon Simple Storage Service Developer Guide.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations en résultant représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCPs) : SCPs politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans AWS Organizations. AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de votre entreprise. Si vous activez toutes les fonctionnalités d'une organisation, vous pouvez appliquer des politiques de contrôle des services (SCPs) à l'un ou à l'ensemble de vos comptes. Le SCP limite les autorisations pour les entités figurant dans les comptes des membres, y compris chacune Utilisateur racine d'un compte AWS d'entre elles. Pour plus d'informations sur les Organizations et consultez SCPs les politiques de contrôle des services dans le Guide de AWS Organizations l'utilisateur.
- Politiques de contrôle des ressources (RCPs) : RCPs politiques JSON que vous pouvez utiliser pour définir le maximum d'autorisations disponibles pour les ressources de vos comptes sans mettre à jour les politiques IAM associées à chaque ressource que vous possédez. Le RCP limite les autorisations pour les ressources des comptes membres et peut avoir un impact sur les autorisations effectives pour les identités, y compris Utilisateur racine d'un compte AWS, qu'elles appartiennent ou non à votre organisation. Pour plus d'informations sur les Organizations RCPs, y compris une liste de ces Services AWS supports RCPs, consultez la section Resource control policies (RCPs) dans le guide de AWS Organizations l'utilisateur.

- Politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de séance en résultant sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations, consultez Politiques de session dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Comment fonctionne la modernisation AWS du mainframe avec IAM

Avant d'utiliser l'IAM pour gérer l'accès à la modernisation du AWS mainframe, découvrez quelles fonctionnalités IAM peuvent être utilisées dans le cadre de la modernisation du mainframe. AWS

Fonctionnalités IAM que vous pouvez utiliser dans le cadre de la modernisation du AWS mainframe

Fonctionnalité IAM	AWS Assistance à la modernisation du mainframe
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Non
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition de politique	Oui
ACLs	Non
ABAC (étiquettes dans les politiques)	Oui

Fonctionnalité IAM	AWS Assistance à la modernisation du mainframe
Informations d'identification temporaires	Oui
Transmission des sessions d'accès (FAS)	Oui
Rôles de service	Oui
Rôles liés à un service	Oui

Pour obtenir une vue d'ensemble de la façon dont la modernisation du AWS mainframe et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez la section AWS Services compatibles avec IAM dans le Guide de l'utilisateur IAM.

Politiques basées sur l'identité pour AWS la modernisation du mainframe

Prend en charge les politiques basées sur l'identité : oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité, car celle-ci s'applique à l'utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Politiques basées sur les ressources dans le cadre de la modernisation du mainframe AWS

Prend en charge les politiques basées sur les ressources : non

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal intercompte à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont différents Comptes AWS, un administrateur IAM du compte sécurisé doit également accorder à l'entité principale (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions stratégiques pour la modernisation des AWS ordinateurs centraux

Prend en charge les actions de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de stratégie portent généralement le même nom que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations

nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour consulter la liste des actions de modernisation du AWS mainframe, voir Actions définies par la modernisation du AWS mainframe dans la référence d'autorisation de service.

Dans le cadre de la modernisation AWS du mainframe, les actions stratégiques utilisent le préfixe suivant avant l'action :

```
m2
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "m2:StartApplication",
      "m2:StopApplication"
         ]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `List`, incluez l'action suivante :

```
"Action": "m2:List*"
```

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Ressources relatives aux politiques relatives à la AWS modernisation des ordinateurs centraux

Prend en charge les ressources de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Vous pouvez restreindre l'accès à des ressources spécifiques de modernisation du AWS mainframe en les utilisant ARNs pour identifier la ressource à laquelle s'applique la politique IAM. Pour plus d'informations sur le format de ARNs, consultez Amazon Resource Names (ARNs) dans le Références générales AWS.

Par exemple, un environnement de modernisation de AWS mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:m2:regionId:accountId:env/service-generated-unique-identifier"
```

Une application de modernisation AWS du mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:m2:regionId:accountId:app/service-generated-unique-identifier"
```

Les actions de modernisation du AWS mainframe ne prennent pas toutes en charge les autorisations au niveau des ressources. Pour les actions qui ne prennent pas en charge les autorisations au niveau des ressources, vous devez utiliser le caractère générique (*).

Les actions de modernisation du AWS mainframe suivantes ne prennent pas en charge les autorisations au niveau des ressources.

```
ListApplications
            ListApplicationVersions
            ListBatchJobDefinitions
            ListBatchJobExecutions
            ListDataSetImportHistory
            ListDataSets
```

```
            ListDeployments
            ListEngineVersions
            ListEnvironments
            ListTagsForResource
```

Pour consulter la liste des types de ressources de modernisation du AWS mainframe et de leurs caractéristiques ARNs, voir Ressources définies par la modernisation du AWS mainframe dans la référence d'autorisation de service. Pour savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, voir Actions définies par la modernisation du AWS mainframe.

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

AWS Autorisations de l'API de modernisation du mainframe : référence aux actions, aux ressources et aux conditions

Lorsque vous rédigez des politiques d'autorisation que vous pouvez associer à une identité IAM (politiques basées sur l'identité), vous pouvez utiliser le tableau suivant comme référence. Le tableau inclut les éléments suivants :

- Chaque opération de l'API de modernisation du AWS mainframe.
- Les actions correspondantes pour lesquelles vous pouvez accorder des autorisations pour effectuer l'action.
- La AWS ressource pour laquelle vous pouvez accorder les autorisations.

Vous spécifiez les actions dans le champ `Action` de la politique, ainsi que la valeur des ressources dans le champ `Resource` de la politique.

Vous pouvez utiliser des clés de condition AWS globales dans vos politiques de modernisation AWS du mainframe pour exprimer des conditions. Pour obtenir la liste complète des AWS clés, consultez la section Clés de condition globale disponibles dans le guide de l'utilisateur IAM.

> **Note**
>
> Pour indiquer une action, utilisez le préfixe `m2:` suivi du nom de l'opération d'API (par exemple, `m2:CreateApplication`).

AWS API de modernisation du mainframe et autorisations requises pour les actions

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
CancelBatchJobExecution		Application

CreateApplication	`iam:PassRole`
	`kms:DescribeKey`
	`kms:CreateGrant`
	`s3:GetObject`
	`s3:ListBucket`
		Application

CreateDataSetImportTask	`s3:GetObject`	Application
CreateDataSetExportTask	`kms:DescribeKey` `s3:PutObject`	Application

CreateDeployment	`elasticloadbalancing:AddTags`
	`elasticloadbalancing:CreateListener`
	`elasticloadbalancing:CreateTargetGroup`
	`elasticloadbalancing:RegisterTargets`
	`elasticloadbalancing:DeleteListener`
		Application

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources

	`elasticloadbalancing:DeleteTargetGroup`
	
	`elasticloadbalancing:DeregisterTargets`
	
	`elasticloadbalancing:DeleteLoadBalancer`
	
	`logs:CreateLogDelivery`
	
	`logs:GetLogDelivery`
	
	`logs:UpdateLogDelivery`
	
	`logs:DeleteLogDelivery`
	
	`logs:ListLogDeliveries`
	
	`logs:PutResourcePolicy`
	
	`logs:DescribeResourcePolicies`
	
	`logs:DescribeLogGroups`	

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
CreateEnvironment	ec2:CreateNetworkInterface ec2:CreateNetworkInterfacePermission ec2:DescribeNetworkInterfaces ec2:DescribeSecurityGroups ec2:DescribeSubnets ec2:DescribeVpcAttribute ec2:DescribeVpcs ec2:ModifyNetworkInterfaceAttribute elasticfilesystem:DescribeMountTargets elasticloadbalancing:AddTags elasticloadbalancing:CreateLoadBalancer elasticloadbalancing:DeleteLoadBalancer kms:DescribeKey kms:CreateGrant fsx:DescribeFileSystems	Environnement

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
	`iam:CreateServiceLinkedRole`	

DeleteApplication	`elasticloadbalancing:DeleteListener`
	`elasticloadbalancing:DeleteTargetGroup`
	`logs:DeleteLogDelivery`
		Application

DeleteApplicationFromEnvironment	`elasticloadbalancing:DeleteListener` `elasticloadbalancing:DeleteTargetGroup`	Application Environnement
DeleteEnvironment	`elasticloadbalancing:DeleteLoadBalancer`	Environnement
GetApplication		Application
GetApplicationVersion		Application
GetBatchJobExecution		Application
GetDataSetDetails		Application
GetDataSetImportTask		Application
GetDataSetExportTask		Application
GetDeployment		Application
GetEnvironment		Environnement

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
ListApplications		*
ListApplicationVersions		*
ListBatchJobDefinitions		*
ListBatchJobExecutions		*
ListDataSetImportHistory		*
ListDataSetExportHistory		*
ListDataSets		*
ListDeployments		*
ListEngineVersions		*
ListEnvironments		*
ListTagsForResource		*
StartApplication		Application
StartBatchJob		Application
StopApplication		Application
TagResource		*
UntagResource		*

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources

UpdateApplication	`s3:GetObject`
	
	`s3:ListBucket`
		Application

UpdateEnvironment	`kms:DescribeKey`	Environnement

Clés d'état des politiques pour la modernisation des AWS ordinateurs centraux

Prend en charge les clés de condition de politique spécifiques au service : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR` opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Les clés de condition suivantes sont spécifiques à la modernisation du AWS mainframe

```
m2:EngineType
            m2:InstanceType
```

Pour consulter la liste des clés de condition de modernisation du AWS mainframe, voir Clés de condition pour la modernisation du AWS mainframe dans la référence d'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, voir Actions définies par la modernisation du AWS mainframe.

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Listes de contrôle d'accès (ACLs) dans le cadre de la AWS modernisation du mainframe

Supports ACLs : Non

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Contrôle d'accès basé sur les attributs (ABAC) avec modernisation du mainframe AWS

Prise en charge d'ABAC (balises dans les politiques) : Oui

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux entités IAM (utilisateurs ou rôles) et à de nombreuses AWS ressources. L'étiquetage des entités et des ressources est la première étape d'ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l'identification du principal correspond à celle de la ressource à laquelle il tente d'accéder.

L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur ABAC, consultez Définition d'autorisations avec l'autorisation ABAC dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

Utilisation d'informations d'identification temporaires dans le cadre de la AWS modernisation du mainframe

Prend en charge les informations d'identification temporaires : oui

Certains Services AWS ne fonctionnent pas lorsque vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, y compris celles qui Services AWS fonctionnent avec des informations d'identification temporaires, consultez Services AWS la section relative à l'utilisation d'IAM dans le guide de l'utilisateur d'IAM.

Vous utilisez des informations d'identification temporaires si vous vous connectez à l' AWS Management Console aide d'une méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS l'aide du lien d'authentification unique (SSO) de votre entreprise, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Passage d'un rôle utilisateur à un rôle IAM (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide de l' AWS API AWS CLI or. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour y accéder AWS. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Sessions d'accès direct pour la modernisation du AWS mainframe

Prend en charge les sessions d'accès direct (FAS) : oui

Lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une

action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

> **Important**
>
> Ces jetons permettent à AWS Mainframe Modernization d'accéder aux données des clients sans votre accord explicite ; par exemple, AWS Mainframe Modernization déploie des artefacts d'application avec les données commerciales associées à partir d'un compartiment Amazon S3 sans obtenir l'autorisation explicite du client. Vous devrez peut-être mettre à jour toute documentation de conformité en conséquence.

Rôles de service pour la modernisation AWS du mainframe

Prend en charge les rôles de service : oui

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

AWS La modernisation du mainframe prend en charge les rôles de service liés aux accrocs d'activité (transaction/tâche terminée ou fin, etc.).

> **Warning**
>
> La modification des autorisations associées à un rôle de service peut perturber la fonctionnalité de modernisation AWS du mainframe. Modifiez les rôles de service uniquement lorsque AWS Mainframe Modernization fournit des instructions à cet effet.

Choisir un rôle IAM dans la modernisation du AWS mainframe

Si vous avez déjà créé un rôle IAM que vos applications exécutées sur Amazon EC2 peuvent assumer, vous pouvez choisir ce rôle lorsque vous créez un modèle de lancement ou une configuration de lancement. AWS La modernisation du mainframe vous fournit une liste de rôles parmi lesquels choisir. Lors de la création de ces rôles, il est important d'associer des politiques IAM de moindre privilège qui restreignent l'accès aux appels d'API spécifiques requis par l'application. Pour plus d'informations, consultez le rôle IAM pour les applications qui s'exécutent sur des EC2 instances Amazon dans le manuel Amazon EC2 Auto Scaling User Guide.

Rôles liés aux services pour AWS la modernisation du mainframe

Prend en charge les rôles liés aux services : Oui

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus de détails sur la création ou la gestion des rôles liés au service AWS Mainframe Modernization, consultez. Utilisation des rôles liés aux services pour AWS Mainframe Modernization

Pour plus d'informations sur la création ou la gestion des rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôle lié à un service. Choisissez le lien Oui pour consulter la documentation du rôle lié à ce service.

Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier les ressources de modernisation AWS du mainframe. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l'API AWS Management Console, AWS Command Line Interface (AWS CLI) ou de AWS l'API. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques IAM (console) dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par AWS Mainframe Modernization, y compris le format du ARNs pour chacun des types de ressources, voir Actions, ressources et clés de condition pour la modernisation du AWS mainframe dans la référence d'autorisation de service.

Rubriques

- Bonnes pratiques en matière de politiques
- Utilisation de la console de modernisation AWS du mainframe
- Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer des ressources de modernisation du AWS mainframe dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service

AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue. Compte AWS Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la console de modernisation AWS du mainframe

Pour accéder à la console AWS Mainframe Modernization, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et d'afficher des informations détaillées sur les ressources de modernisation du AWS mainframe dans votre Compte AWS. Si vous créez une politique basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs ou rôles) tributaires de cette politique.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement l'API AWS CLI ou l' AWS API. Autorisez plutôt l'accès à uniquement aux actions qui correspondent à l'opération d'API qu'ils tentent d'effectuer.

Pour garantir que les utilisateurs et les rôles peuvent toujours utiliser la console de modernisation du AWS mainframe, associez également la politique de modernisation du AWS mainframe `ConsoleAccess` ou la politique `ReadOnly` AWS gérée aux entités. Pour plus d'informations, consultez Ajout d'autorisations à un utilisateur dans le Guide de l'utilisateur IAM.

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Résolution des problèmes liés à la modernisation AWS du mainframe (identité et accès)

Utilisez les informations suivantes pour vous aider à diagnostiquer et à résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation de AWS Mainframe Modernization et de l'IAM.

Rubriques

- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite permettre à des personnes extérieures à moi d'accéder Compte AWS à mes ressources de modernisation de l' AWS ordinateur central

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez un message d'erreur indiquant que vous n'êtes pas autorisé à effectuer l'`iam:PassRole`action, vos politiques doivent être mises à jour pour vous permettre de transférer un rôle à AWS Mainframe Modernization.

Certains vous Services AWS permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans AWS Mainframe Modernization. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je souhaite permettre à des personnes extérieures à moi d'accéder Compte AWS à mes ressources de modernisation de l' AWS ordinateur central

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACLs), vous pouvez utiliser ces politiques pour autoriser les utilisateurs à accéder à vos ressources.

Pour plus d'informations, consultez les éléments suivants :

- Pour savoir si la modernisation AWS du mainframe prend en charge ces fonctionnalités, consultezComment fonctionne la modernisation AWS du mainframe avec IAM.
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour en savoir plus sur la différence entre l'utilisation des rôles et des politiques basées sur les ressources pour l'accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Utilisation des rôles liés aux services pour AWS Mainframe Modernization

AWS Mainframe Modernization utilise des AWS Identity and Access Management rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM directement lié à. AWS Mainframe Modernization Les rôles liés au service sont prédéfinis par AWS Mainframe Modernization et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service facilite la configuration AWS Mainframe Modernization car vous n'avez pas à ajouter manuellement les autorisations nécessaires. AWS Mainframe Modernization définit les autorisations associées à ses rôles liés aux services et, sauf indication contraire, seul AWS

Mainframe Modernization peut assumer ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Cela protège vos AWS Mainframe Modernization ressources car vous ne pouvez pas supprimer par inadvertance l'autorisation d'accès aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés aux services, consultez la section AWS Services qui fonctionnent avec IAM et recherchez les services dont la valeur est Oui dans la colonne Rôles liés aux services. Sélectionnez un Oui ayant un lien pour consulter la documentation du rôle lié à un service, pour ce service.

Autorisations des rôles liés à un service pour AWS Mainframe Modernization

AWS Mainframe Modernization utilise le rôle lié au service nommé AWSServiceRoleForAWSM2: configurez le réseau pour qu'il se connecte à votre VPC et accède à des ressources telles que les systèmes de fichiers.

Le rôle AWSService RoleFor AWSM2 lié à un service fait confiance aux services suivants pour assumer le rôle :

- `m2.amazonaws.com`

La politique d'autorisations de rôle nommée AWSM2 ServicePolicy AWS Mainframe Modernization permet d'effectuer les actions suivantes sur les ressources spécifiées :

- Créez, supprimez, décrivez et attachez des autorisations aux interfaces EC2 réseau Amazon pour que l' AWS Mainframe Modernization environnement établisse la connectivité au VPC du client.
- Enregistrez ou désenregistrez des entrées dans Elastic Load Balancing, qui permet aux clients de se connecter à l' AWS Mainframe Modernization environnement.
- Décrivez le système de FSx fichiers Amazon EFS ou Amazon, le cas échéant.
- Envoyez des métriques au client CloudWatch depuis l'environnement d'exécution.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
```

```
    "Effect": "Allow",
    "Action": [
     "ec2:DescribeSubnets",
     "ec2:CreateNetworkInterface",
     "ec2:DeleteNetworkInterface",
     "ec2:DescribeNetworkInterfaces",
     "ec2:CreateNetworkInterfacePermission",
     "ec2:ModifyNetworkInterfaceAttribute"
    ],
    "Resource": "*"
   },
   {
    "Effect": "Allow",
    "Action": [
     "elasticfilesystem:DescribeMountTargets"
    ],
    "Resource": "*"
   },
   {
    "Effect": "Allow",
    "Action": [
     "elasticloadbalancing:RegisterTargets",
     "elasticloadbalancing:DeregisterTargets"
    ],
    "Resource": "*"
   },
   {
    "Effect": "Allow",
    "Action": [
     "fsx:DescribeFileSystems"
    ],
    "Resource": "*"
   },
   {
    "Effect": "Allow",
    "Action": [
     "cloudwatch:PutMetricData"
    ],
    "Resource": "*",
    "Condition": {
     "StringEquals": {
      "cloudwatch:namespace": [
       "AWS/M2"
      ]
```

```
      }
    }
   }
  ]
}
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour AWS Mainframe Modernization

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement d'exécution dans le AWS Management Console AWS CLI, le ou l' AWS API, vous AWS Mainframe Modernization créez le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement d'exécution, le rôle lié au service est à nouveau AWS Mainframe Modernization créé pour vous.

Modification d'un rôle lié à un service pour AWS Mainframe Modernization

AWS Mainframe Modernization ne vous permet pas de modifier le rôle AWSService RoleFor AWSM2 lié au service. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour plus d'informations, consultez Modification d'un rôle lié à un service dans le IAM Guide de l'utilisateur.

Suppression d'un rôle lié à un service pour AWS Mainframe Modernization

Si vous n'avez plus besoin d'utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

> **Note**
>
> Si le AWS Mainframe Modernization service utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression risque d'échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer AWS Mainframe Modernization les ressources utilisées par AWSService RoleFor AWSM2

- Supprimez les environnements d'exécution dans AWS Mainframe Modernization. Assurez-vous de supprimer les applications d'un environnement avant de supprimer l'environnement lui-même.

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au AWSService RoleFor AWSM2 service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service AWS Mainframe Modernization

AWS Mainframe Modernization prend en charge l'utilisation de rôles liés au service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Régions et points de terminaison AWS.

Validation de conformité pour la AWS modernisation du mainframe

Des auditeurs tiers évaluent la sécurité et la conformité de la modernisation des AWS mainframes dans le cadre de multiples programmes de AWS conformité. Il s'agit notamment des certifications SOC, PCI, FedRAMP, HIPAA et d'autres.

Pour obtenir la liste des AWS services concernés par des programmes de conformité spécifiques, voir Services AWS concernés par programme de conformité . Pour obtenir des renseignements généraux, consultez Programmes de conformitéAWS .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Lorsque vous utilisez la modernisation des AWS mainframes, votre responsabilité en matière de conformité dépend de la sensibilité de vos données, des objectifs de conformité de votre entreprise et des lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides démarrage rapide de la sécurité et de la conformité. Ces guides de déploiement traitent des considérations architecturales et fournissent des étapes pour déployer des environnements de base axés sur la sécurité et la conformité sur AWS.
- Livre blanc sur l'architecture pour la sécurité et la conformité HIPAA — Ce livre blanc décrit comment les entreprises peuvent créer des applications conformes à la loi HIPAA. AWS
- AWS Ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- Évaluation des ressources à l'aide des règles énoncées dans le guide du AWS Config développeur : AWS Configévalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub— Ce AWS service fournit une vue complète de l'état de votre sécurité interne, AWS ce qui vous permet de vérifier votre conformité aux normes et aux meilleures pratiques du secteur de la sécurité.

Résilience dans la AWS modernisation des mainframes

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par un réseau à latence faible, à débit élevé et à forte redondance. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont davantage disponibles, tolérantes aux pannes et ont une plus grande capacité de mise à l'échelle que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

Sécurité de l'infrastructure dans AWS Mainframe Modernization

En tant que service géré, AWS Mainframe Modernization il est protégé par la sécurité du réseau AWS mondial. Pour plus d'informations sur les services AWS de sécurité et sur la manière dont

AWS l'infrastructure est protégée, consultez la section Sécurité du AWS cloud. Pour concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder AWS Mainframe Modernization via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Accès AWS Mainframe Modernization via un point de terminaison AWS PrivateLink d'interface

Vous pouvez l'utiliser AWS PrivateLink pour créer une connexion privée entre votre VPC et. AWS Mainframe Modernization Vous pouvez y accéder AWS Mainframe Modernization comme s'il se trouvait dans votre VPC, sans utiliser de passerelle Internet, de périphérique NAT, de connexion VPN ou AWS Direct Connect de connexion. Les instances de votre VPC n'ont pas besoin d'adresses IP publiques pour y accéder. AWS Mainframe Modernization

Vous établissez cette connexion privée en créant un point de terminaison d'interface optimisé par AWS PrivateLink. Nous créons une interface réseau de point de terminaison dans chaque sous-réseau que vous activez pour le point de terminaison d'interface. Il s'agit d'interfaces réseau gérées par le demandeur qui servent de point d'entrée pour le trafic destiné à AWS Mainframe Modernization.

Pour plus d'informations, consultez la section Accès Services AWS par AWS PrivateLink le biais du AWS PrivateLink guide.

Considérations relatives à AWS Mainframe Modernization

Avant de configurer un point de terminaison d'interface pour AWS Mainframe Modernization, consultez les considérations du AWS PrivateLink guide.

AWS Mainframe Modernization prend en charge les appels à toutes ses actions d'API via le point de terminaison de l'interface.

Créez un point de terminaison d'interface pour AWS Mainframe Modernization

Vous pouvez créer un point de terminaison d'interface pour AWS Mainframe Modernization utiliser la console Amazon VPC ou le AWS Command Line Interface ()AWS CLI. Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide AWS PrivateLink .

Créez un point de terminaison d'interface pour AWS Mainframe Modernization utiliser le nom de service suivant :

```
com.amazonaws.region.m2
```

Si vous activez le DNS privé pour le point de terminaison de l'interface, vous pouvez envoyer des demandes d'API à AWS Mainframe Modernization l'aide de son nom DNS régional par défaut. Par exemple, `m2.us-east-1.amazonaws.com`.

Création d'une politique de point de terminaison pour votre point de terminaison d'interface

Une politique de point de terminaison est une ressource IAM que vous pouvez attacher à votre point de terminaison d'interface. La politique de point de terminaison par défaut autorise un accès complet AWS Mainframe Modernization via le point de terminaison de l'interface. Pour contrôler l'accès autorisé AWS Mainframe Modernization depuis votre VPC, associez une politique de point de terminaison personnalisée au point de terminaison de l'interface.

Une politique de point de terminaison spécifie les informations suivantes :

- Les principaux qui peuvent effectuer des actions (Comptes AWS utilisateurs et rôles IAM).
- Les actions qui peuvent être effectuées.
- La ressource sur laquelle les actions peuvent être effectuées.

Pour plus d'informations, consultez Contrôle de l'accès aux services à l'aide de politiques de point de terminaison dans le Guide AWS PrivateLink .

Exemple : politique de point de terminaison VPC pour les actions AWS Mainframe Modernization

Voici un exemple de politique de point de terminaison. Lorsque vous attachez cette politique à votre point de terminaison d'interface, elle accorde l'accès aux actions AWS Mainframe Modernization répertoriées pour tous les principaux sur toutes les ressources.

```
//Example of an endpoint policy where access is granted to the
//listed AWS Mainframe Modernization actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Allow",
         "Action": [
            "m2:ListApplications",
            "m2:ListEnvironments",
            "m2:ListDeployments"
         ],
         "Resource":"*"
      }
   ]
}

//Example of an endpoint policy where access is denied to all the
//AWS Mainframe Modernization CREATE actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Deny",
         "Action": [
            "m2:Create*"
         ],
         "Resource":"*"
      }
   ]
}
```

Surveillance de la AWS modernisation du mainframe

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances de AWS Mainframe Modernization et de vos autres solutions AWS. AWS fournit les outils de surveillance suivants pour suivre la modernisation du AWS mainframe, signaler tout problème et prendre des mesures automatiques le cas échéant :

- Amazon CloudWatch surveille vos AWS ressources et les applications que vous utilisez AWS en temps réel. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés, et définir des alarmes qui vous informent ou prennent des mesures lorsqu'une métrique spécifique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation du processeur ou d'autres indicateurs de vos EC2 instances Amazon et lancer automatiquement de nouvelles instances en cas de besoin. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d' EC2 instances Amazon et d'autres sources. CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.
- AWS CloudTrailcapture les appels d'API et les événements associés effectués par ou pour le compte de votre AWS compte et envoie les fichiers journaux dans un compartiment Amazon S3 que vous spécifiez. Vous pouvez identifier les utilisateurs et les comptes appelés AWS, l'adresse IP source à partir de laquelle les appels ont été effectués et la date des appels. Pour plus d'informations, consultez le AWS CloudTrail Guide de l'utilisateur .

Surveillance de la modernisation des AWS mainframes avec Amazon CloudWatch

Vous pouvez surveiller la modernisation AWS du mainframe à l'aide d' CloudWatchun outil qui collecte les données brutes et les traite en indicateurs lisibles en temps quasi réel. Ces statistiques sont enregistrées pour une durée de 15 mois ; par conséquent, vous pouvez accéder aux informations historiques et acquérir un meilleur point de vue de la façon dont votre service ou application web s'exécute. Vous pouvez également définir des alarmes qui surveillent certains seuils

et envoient des notifications ou prennent des mesures lorsque ces seuils sont atteints. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.

Les tableaux suivants répertorient les mesures et les dimensions de la modernisation des AWS mainframes. L'espace de noms pour ces métriques est`AWS/M2`.

Métriques de l'environnement d'exécution

Métrique	Description
CPUUtilization	L'utilisation du processeur par les instances de l'environnement. Dimension : EnvironmentID Unités : pourcentage Statistiques valides : moyenne, minimum, maximum
InboundNetworkThroughput	Débit réseau entrant des instances de l'environnement. Dimension : EnvironmentID Unités : octets par seconde Statistiques valides : moyenne, minimum, maximum
MemoryUtilization	Utilisation de la mémoire par les instances de l'environnement. Dimension : EnvironmentID Unités : pourcentage Statistiques valides : moyenne, minimum, maximum

Métrique	Description
OutboundNetworkThroughput	Débit réseau sortant des instances de l'environnement. Dimension : EnvironmentID Unités : octets par seconde Statistiques valides : moyenne, minimum, maximum

Métriques d'application

Métrique	Description
BatchJobCompletedCount	Le nombre de tâches terminées pendant l'intervalle de temps. Cette métrique est disponible pour Rocket Software (anciennement Micro Focus) et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : somme
BatchJobFailedCount	Le nombre de tâches ayant échoué pendant l'intervalle de temps. Cette métrique est disponible pour Rocket Software et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre

Métrique	Description
	Statistiques valides : somme
JvmMemoryFree	La quantité de mémoire disponible qui n'est pas actuellement utilisée par la machine virtuelle Java. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : octets Statistiques valides : moyenne, minimum, maximum
JvmMemoryMax	La quantité maximale de mémoire autorisée pour la machine virtuelle Java. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : octets Statistiques valides : moyenne, minimum, maximum

Métrique	Description
JvmMemoryUsed	La quantité de mémoire activement utilisée par la machine virtuelle Java. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : octets Statistiques valides : moyenne, minimum, maximum
ProcessesActiveCount	Nombre actif de processus d'exécution de services simultanés qui traitent des demandes. Cette métrique n'est disponible que pour le moteur d'exécution Rocket Software. Dimension : ApplicationID Unités : nombre Statistiques valides : somme

Métrique	Description
SessionCount	Nombre de sessions HTTP pour l'application. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : moyenne, minimum, maximum
SharedMemoryFree	Mémoire disponible pour que le serveur d'entreprise stocke toutes les informations dont il a besoin pour exécuter des transactions et des tâches. Cette métrique n'est disponible que pour le moteur d'exécution Rocket Software. Dimension : ApplicationID Unités : kilo-octets Statistiques valides : moyenne, minimum, maximum

Métrique	Description
SharedMemoryTotal	Quantité totale de mémoire partagée allouée au serveur d'entreprise pour stocker toutes les informations dont il a besoin pour exécuter des transactions et des tâches. Cette métrique n'est disponible que pour le moteur d'exécution Rocket Software. Dimension : ApplicationID Unités : kilo-octets Statistiques valides : moyenne, minimum, maximum
ThreadActiveCount	Le nombre de threads du moteur qui traitent les demandes. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : moyenne, minimum, maximum

Métrique	Description
TransactionCompletedCount	Le nombre de transactions validées pendant l'intervalle de temps. Cette métrique est disponible pour Rocket Software et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : somme
TransactionFailedCount	Le nombre de transactions ayant échoué pendant l'intervalle de temps. Cette métrique est disponible pour Rocket Software et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : somme

Métrique	Description
TransactionResponseTime	Temps écoulé entre le moment où un utilisateur envoie une demande et le moment où l'applica tion indique que la demande est terminée. Cette métrique est disponible pour Rocket Software et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : millisecondes Statistiques valides : moyenne, minimum, maximum

Dimensions

Dimension	Description
applicationId	Cette dimension filtre la métrique en fonction de l'identifiant de l'application identifiée.
Identifiant de l'environnement	Cette dimension filtre la métrique en fonction de l'environnement identifié par ID.

Journalisation des appels d'API de modernisation AWS du mainframe à l'aide de AWS CloudTrail

AWS La modernisation du mainframe est intégrée à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans le cadre de la modernisation du AWS mainframe. CloudTrail capture tous les appels d'API pour la modernisation AWS du mainframe sous forme d'événements. Les appels capturés incluent des appels provenant de la console de modernisation du AWS mainframe et des appels de code vers les opérations de l'API de modernisation du AWS mainframe. Si vous créez un suivi, vous pouvez activer la diffusion

continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements liés à la modernisation du AWS mainframe. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à AWS Mainframe Modernization, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite et des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

AWS Informations sur la modernisation du mainframe dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans le cadre de la modernisation du AWS mainframe, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris les événements liés à la modernisation AWS du mainframe, créez une trace. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal d'activité dans la console, il s'applique à toutes les régions AWS. Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour plus d'informations, consultez les ressources suivantes :

- Présentation de la création d'un journal de suivi
- CloudTrail services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux provenant de plusieurs régions
- Réception de fichiers CloudTrail journaux provenant de plusieurs comptes

Toutes les actions de modernisation AWS du mainframe sont enregistrées CloudTrail et documentées dans le manuel AWS Mainframe Modernization API Reference. Par exemple, les appels au`CreateApplication`, `CreateEnvironment` et les `CreateDeployment` actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer :

- Si la demande a été effectuée avec des informations d'identification d'utilisateur root ou d'utilisateur root.
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour de plus amples informations, veuillez consulter l'élément userIdentity CloudTrail .

Comprendre les AWS entrées des fichiers journaux de modernisation des mainframes

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal illustrant l'`CreateApplication`action.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAII6WZTHGYAEXAMPLE",
        "arn": "arn:aws:sts::444455556666:assumed-role/Admin/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAII6WZTHGYAEXAMPLE",
                "arn": "arn:aws:iam::444455556666:role/Admin",
                "accountId": "444455556666",
                "userName": "Admin"
```

```
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-06-01T20:38:22Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-06-01T20:40:39Z",
    "eventSource": "m2.amazonaws.com",
    "eventName": "CreateApplication",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "72.21.196.65",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:91.0) Gecko/20100101
 Firefox/91.0",
    "requestParameters": {
        "clientToken": "1abc23de-f45g-6789-h01i-jkl2m3456789",
        "name": "MyApp",
        "description": "",
        "engineType": "microfocus",
        "definition": {
            "content": "{}"
        },
        "tags": {}
    },
    "responseElements": {
        "applicationVersion": 1,
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "applicationArn": "arn:aws:m2:us-east-1:444455556666:app/
lsfhmwhw7fffrosff2lncwqcua",
        "applicationId": "lsfhmwhw7fffrosff2lncwqcua"
    },
    "requestID": "36982d38-fcde-4bfe-a89a-7bd78d43c926",
    "eventID": "d7f0fc36-46ae-4157-9a79-c79f385fda98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management"
}
```

Résolution des problèmes liés à la AWS modernisation des mainframes

Utilisez les informations de cette section pour vous aider à résoudre les erreurs courantes dans les applications de modernisation des AWS mainframes et les environnements d'exécution utilisant à la fois les moteurs AWS Blu Age et Rocket Software.

Rubriques

- Erreur de résolution des problèmes : expiration du délai d'attente pendant que le nom de l'ensemble de données soit déverrouillé
- Erreur de résolution des problèmes : Impossible d'accéder à l'URL d'une application
- Résolution des problèmes : AWS Blu Insights ne s'ouvre pas depuis la console
- Erreur de résolution des problèmes : environnement malsain
- Résolution des problèmes de licence pour Rocket Software (anciennement Micro Focus)

Erreur de résolution des problèmes : expiration du délai d'attente pendant que le nom de l'ensemble de données soit déverrouillé

Cette page explique comment résoudre votre erreur lorsque vous constatez qu'une autre application d'un environnement verrouille un ensemble de données partagé.

- Moteur : AWS Blu Age
- Composant : Blusam

Si vous voyez cette erreur dans les CloudWatch journaux Amazon d'une application de modernisation de AWS mainframe utilisant le moteur AWS Blu Age et exécutée dans un environnement utilisant le modèle de haute disponibilité, cela indique qu'une autre application bloque un ensemble de données partagé. Généralement, cette situation se produit si l'autre application se bloque ou échoue et ne libère pas le verrou.

Recherchez une application défaillante et vérifiez si elle utilise le même ensemble de données que celui indiqué dans le message d'erreur. Vérifiez si l'application s'exécute dans un environnement d'exécution avec le modèle de haute disponibilité. L'application qui a déclenché l'exception de délai d'expiration ne peut pas continuer et affiche le `Failed` statut.

Cause courante

L'application `example-app-1` tente de verrouiller un enregistrement `example-record-1` pour une opération d'écriture. Cette opération crée à la fois un verrou sur l'ensemble de données`example-dataset-1`, qui en est propriétaire`example-record-1`, et un verrou sur `example-record-1` lui-même. Maintenant`example-app-2`, une autre application essaie de verrouiller le même enregistrement`example-record-1`. L'ensemble de données et l'enregistrement étant déjà verrouillés, il `example-app-2` attend que le verrou soit déverrouillé. En cas de `example-app-1` panne, le verrouillage bloqué sur l'ensemble de données existe `example-dataset-1` toujours, ce qui entraîne l'annulation de sa tentative `example-app-2` d'écriture et le déclenchement d'une exception de délai d'expiration. Cette situation de blocage empêche toutes les applications d'y accéder. `example-dataset-1`

Résolution

Pour résoudre le problème immédiatement, vous pouvez forcer le déverrouillage. Pour éviter qu'une situation similaire ne se reproduise à l'avenir, vous pouvez configurer deux paramètres qui contrôlent le mécanisme de réparation auto Blusam.

Forcer le verrou à le relâcher

Le gestionnaire de verrous Blusam utilise Amazon ElastiCache (Redis OSS) pour fournir des verrous partagés entre les applications. Pour débloquer les verrous ElastiCache, utilisez l'utilitaire Redis CLI. Vous ne pouvez pas supprimer un verrouillage d'enregistrement individuel. Vous devez supprimer tous les verrous du jeu de données propriétaire. Procédez comme suit :

1. Connectez-vous à votre à ElastiCache l'aide de la commande suivante :

   ```
   redis-cli -h hostname -p port
   ```

   Vous trouverez les informations vous concernant ElastiCache dans la ElastiCache console à l'adresse https://console.aws.amazon.com/elasticache/.
2. Entrez votre mot de passe.
3. Entrez la commande que vous souhaitez exécuter, comme suit :

Command	Objectif
KEYS *	Récupérez toutes les clés existantes.
CLÉS * *YOUR_DATASET_NAME*	Obtenez une clé de verrouillage du jeu de données.
DEL *THE_RETURNED_KEY*	Supprimez un verrou de jeu de données.
FLUSHDB	Nettoyez tout le Redis. **Warning** Toutes les données du cache Redis seront perdues. Si le Redis est utilisé à d'autres fins, telles que la gestion de sessions HTTP, vous ne voudrez peut-être pas l'utiliserFLUSHDB.

Configurer le mécanisme de réparation auto Blusam

Le gestionnaire de verrous Blusam inclut un mécanisme de réparation automatique pour éviter les blocages sur les ensembles de données ou les enregistrements. Vous pouvez ajuster les paramètres suivants dans la définition de l'application (`application-main.yml`) pour configurer le mécanisme de réparation automatique :

- `locksDeadTime`: fait référence à la durée maximale pendant laquelle une application peut maintenir un verrou. Lorsque ce délai est écoulé, le verrou est déclaré expiré et immédiatement déverrouillé. La `locksDeadTime` valeur est exprimée en millisecondes et la valeur par défaut est 1000.
- `locksCheck`: définit la stratégie du gestionnaire de verrous Blusam pour vérifier les verrous. Tous les verrous Blusam ElastiCache sont horodatés et ont une date d'expiration. La valeur du `locksCheck` paramètre détermine si les verrous expirés sont supprimés.
  - `off`: aucune vérification n'est exécutée à aucun moment. Des blocages peuvent survenir. (Non recommandé)

- `reboot`: les vérifications sont exécutées lorsqu'une instance d'application AWS Mainframe Modernization exécutée dans un environnement d'exécution AWS Mainframe Modernization est démarrée ou redémarrée. Tous les verrous expirés sont immédiatement libérés. (Par défaut)
- `timeout`: les vérifications sont exécutées lorsqu'une instance de l'application AWS Mainframe Modernization exécutée dans un environnement d'exécution AWS Mainframe Modernization est démarrée ou redémarrée, ou lorsqu'un délai expire lors d'une tentative de verrouillage d'un ensemble de données. Les verrous expirés sont immédiatement libérés.

Pour plus d'informations sur la définition d'une application AWS Blu Age, consultezAWS Exemple de définition d'application Blu Age.

Gestionnaire de serrures Blusam

Dans le contexte d'un environnement d'exécution de modernisation des AWS mainframes utilisant le modèle de haute disponibilité, une application AWS Blu Age peut être déployée plusieurs fois. Pour les applications qui gèrent des ensembles de données Blusam, des problèmes d'accès simultanés peuvent survenir. Le gestionnaire de verrous Blusam garantit l'intégrité des données et gère l'accès en lecture et en écriture aux enregistrements et aux ensembles de données en fournissant des verrous partagés entre les applications qui les utilisent. ElastiCache Ce mécanisme permet à plusieurs applications de lire l'enregistrement simultanément et garantit qu'une seule application à la fois écrit l'enregistrement.

Serrures d'écriture

Pour mettre à jour ou supprimer un enregistrement spécifique, l'application doit d'abord verrouiller l'ensemble de données propriétaire de l'enregistrement, puis verrouiller l'enregistrement lui-même. Lorsque l'enregistrement est verrouillé, le verrouillage du jeu de données est libéré et les autres enregistrements du même ensemble de données peuvent être utilisés. Lorsque l'opération de mise à jour ou de suppression est terminée, le verrouillage des enregistrements conservés est débloqué. Une seule application à la fois peut mettre à jour l'enregistrement, ce qui empêche les autres applications de lire ou d'écrire jusqu'à ce que le verrou soit relâché, si la politique d'application définie autorise l'attente de publication.

Lire les verrous

Tant qu'aucun verrou d'écriture n'est maintenu sur l'enregistrement ou le jeu de données, plusieurs applications peuvent lire les mêmes enregistrements en même temps. Pour verrouiller un enregistrement pour une opération d'écriture, tous les verrous de lecture doivent être libérés.

> **Note**
>
> Le gestionnaire de verrous Blusam gère l'accès depuis plusieurs threads dans une application donnée en utilisant le même mécanisme de verrouillage.

Erreur de résolution des problèmes : Impossible d'accéder à l'URL d'une application

Cette page explique comment résoudre votre erreur lorsque vous ne parvenez pas à accéder à l'URL d'une application de modernisation du AWS mainframe en cours d'exécution.

- Moteur : AWS Blu Age et Rocket Software (anciennement Micro Focus)
- Composant : applications

Si vous ne pouvez pas accéder à l'URL d'une application de modernisation du AWS mainframe en cours d'exécution que vous avez créée et déployée dans un environnement d'exécution AWS Mainframe Modernization, vous devrez peut-être configurer les règles entrantes sur le groupe de sécurité que vous avez associé à l'environnement d'exécution.

Cause courante

Lorsque vous créez un environnement d'exécution, le groupe de sécurité que vous fournissez, y compris le groupe de sécurité par défaut, doit avoir des règles entrantes configurées pour autoriser le trafic vers les applications déployées depuis l'extérieur du VPC, si vous souhaitez autoriser ce type d'accès.

Résolution

Vérifiez si le groupe de sécurité Amazon VPC associé à l'environnement d'exécution autorise le trafic vers l'environnement sur les ports d'application appropriés. Pour vérifier les règles du groupe de sécurité, procédez comme suit :

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Dans le menu de navigation de gauche, sélectionnez Environments.

3. Choisissez l'environnement d'exécution qui héberge l'application à laquelle vous souhaitez vous connecter.
4. Choisissez Configurations.
5. Dans Sécurité et réseau, choisissez le groupe de sécurité. Le lien ouvre les détails du groupe de sécurité dans la console Amazon VPC.
6. Si nécessaire, choisissez Modifier les règles entrantes et ajoutez la règle suivante si elle n'est pas déjà présente :

   Type

   TCP personnalisé

   Port

   8196 ou le port qui correspond aux propriétés de l'écouteur spécifiées dans la définition de l'application. Pour de plus amples informations, veuillez consulter Étape 2 : Création de la définition de l'application.

   Source

   Adresse IP à partir de laquelle vous appelez l'application. Vous pouvez choisir MyIP dans le menu déroulant. Si le délai d'expiration persiste, essayez de choisir N'importe où IPV4 ou n'importe où IPV6. Assurez-vous d'arrêter l'application et de la redémarrer après avoir ajouté la règle entrante au groupe de sécurité.

   Pour plus d'informations, consultez la section Utiliser les règles des groupes de sécurité dans le guide de l'utilisateur Amazon VPC.

Résolution des problèmes : AWS Blu Insights ne s'ouvre pas depuis la console

Cette page décrit comment vous pouvez résoudre le problème que la page Blu Insights ne s'ouvre pas depuis la console AWS Mainframe Modernization.

- Moteur : AWS Blu Age
- Composant : Blu Insights

Lorsque vous essayez d'accéder à Blu Insights depuis la console AWS Mainframe Modernization, celle-ci ne s'ouvre pas et le nouvel onglet se ferme immédiatement.

Cause courante

Le rôle que vous utilisez pour accéder à Blu Insights ne dispose pas d'autorisations suffisantes.

Résolution

Associez une politique IAM au rôle pour lui permettre d'accéder à Blu Insights. Assurez-vous que la politique inclut au moins les autorisations suivantes.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "m2:GetSignedBluinsightsUrl"
            ],
            "Resource": "*"
            }
    ]
}
```

Assurez-vous de remplacer `region` et `account` d'utiliser le bon Région AWS et Compte AWS.

Erreur de résolution des problèmes : environnement malsain

Cette page explique comment résoudre votre erreur lorsque vous recevez une notification indiquant que l'un de vos environnements de modernisation du AWS mainframe ne fonctionne pas correctement.

- Moteur : AWS Blu Age et Rocket Software (anciennement Micro Focus)
- Composant : environnements

Si vous recevez une notification indiquant que l'un de vos environnements de modernisation du AWS mainframe n'est plus fonctionnel, cela s'applique à vous. Vous êtes averti via l'une des sources suivantes :

- L'état de l'environnement défectueux est indiqué dans la console de modernisation de votre AWS mainframe.
- Notification par e-mail concernant l'état de l'environnement malsain provenant de AWS Health.
- Vous pouvez voir un événement connexe lié à la modernisation du AWS mainframe dans votre AWS Health tableau de bord, sous État de votre compte.

Cause courante

L'erreur se produit lorsque les ressources de votre AWS compte associées à l'environnement de modernisation du AWS mainframe sont inaccessibles. Ce problème est souvent dû au fait que les ressources liées à l'environnement sont modifiées ou supprimées.

Résolution

Pour obtenir des conseils spécifiques, utilisez le code d'erreur fourni dans l'e-mail envoyé par AWS Health ou via votre console de modernisation AWS du mainframe.

Code d'erreur :

- Stockage inaccessible

Cette erreur indique que le stockage attaché (Amazon Elastic File System ou systèmes de FSx fichiers Amazon) de l'environnement n'a pas pu être monté correctement. Pour vérifier les informations relatives à un environnement malsain, procédez comme suit :

1. Ouvrez la console de modernisation du AWS mainframe à https://console.aws.amazon.com/m2/l'adresse.
2. Sélectionnez l'environnement malsain, puis sélectionnez Configuration.
3. Choisissez Attached Storage pour afficher les ressources de stockage associées à cet environnement.
4. Vérifiez les configurations liées au réseau, telles que le groupe de sécurité, le sous-réseau et le Amazon VPC associés au stockage. Si ces configurations sont incorrectes, essayez de les restaurer pour résoudre ce problème.

> **Note**
>
> Si le stockage a été supprimé, l'environnement ne peut pas être restauré. Dans ce cas, vous devriez envisager de supprimer l'environnement malsain.

Résolution des problèmes de licence pour Rocket Software (anciennement Micro Focus)

Cette page décrit comment résoudre les problèmes de licence avec le moteur Rocket Software Runtime

- Moteur : Rocket Software
- Composant : Amazon EC2

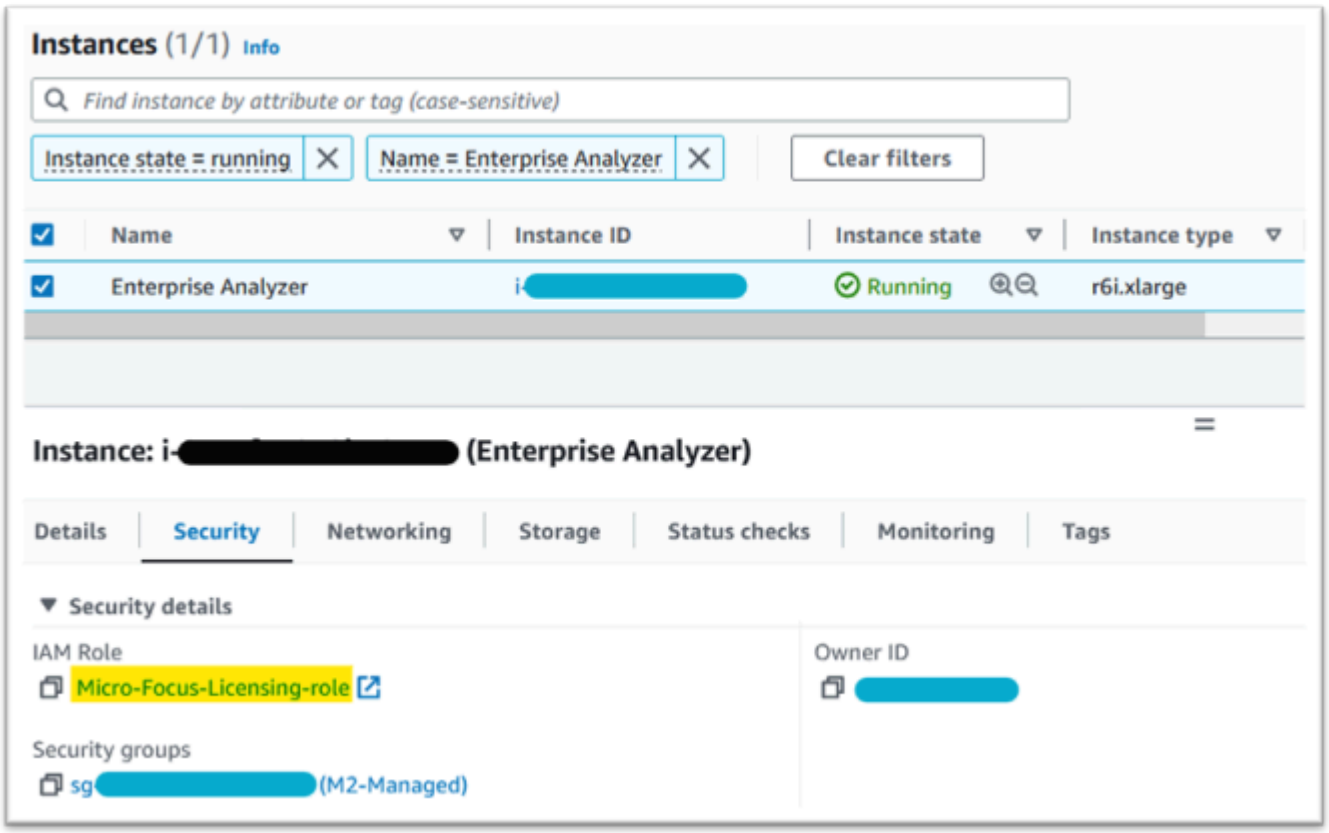
Si vous ne parvenez pas à accéder au ou à l'utiliser AMIs, les informations suivantes peuvent vous aider.

Rubriques

- Vérifiez que l' EC2 instance Amazon possède le rôle de licence IAM
- Utiliser l'analyseur d'accessibilité
- Exécutez le daemon de licence
- Problèmes de licence avec Enterprise Server ou Enterprise Build Tools sous Linux après l'application de correctifs au système d'exploitation

Vérifiez que l' EC2 instance Amazon possède le rôle de licence IAM

Cela peut être vérifié dans l'onglet Sécurité des détails de l' EC2 instance Amazon. Cela peut être modifié à l'aide de l'option de sécurité du menu déroulant Actions.



Utiliser l'analyseur d'accessibilité

Trouvez l'Analyseur de Reachability sur la page de la console. AWS Network Manager

Créez et analysez un chemin entre l' EC2 instance Amazon créée à partir de l'AMI et le point de terminaison Amazon S3 VPC.

Si l' EC2 instance Amazon n'a pas accès à Internet, répétez l'analyse du chemin vers les 4 points de terminaison.

Pour plus d'informations sur Reachability Analyzer, consultez Getting started with Reachability Analyzer dans le guide Reachability Analyzer.

Exécutez le daemon de licence

Sur Windows Enterprise Developer, utilisez la commande suivante à partir d'une invite de commande :

```
“C:\Program Files (x86)\Micro Focus\Enterprise Developer\AdoptOpenJDK\bin\java” -jar
 "C:\Program Files (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

et examinez le résultat. Ignorez les messages SLF4 J et recherchez la première exception.

Sur Enterprise Analyzer, utilisez la commande suivante à partir d'une invite de commande :

```
"C:\Program Files (x86)\Micro Focus\AdoptOpenJDK\bin\java" -jar "C:\Program Files
 (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

et examinez le résultat. Ignorez les messages SLF4 J et recherchez la première exception.

Sur Linux, exécutez :

```
java -jar /var/microfocuslicensing/bin/aws-license-daemon.jar
```

Ignorez les messages SLF4 J et recherchez la première exception.

Par exemple, si la ressource Amazon S3 n'est pas disponible, l'exception est la suivante :

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Exception in thread "main" software.amazon.awssdk.services.s3.model.S3Exception: Access
 Denied (Service: S3, Status Code: 403, Request ID: P6
```

Le message d'exception indique quelle ressource n'est pas disponible. Comparez les valeurs de configuration à celles présentées dans cette rubrique.

Problèmes de licence avec Enterprise Server ou Enterprise Build Tools sous Linux après l'application de correctifs au système d'exploitation

Si vous rencontrez des problèmes de licence avec Enterprise Server ou Enterprise Build Tools sous Linux après l'application de correctifs au système d'exploitation, mettez à jour le démon de licence en téléchargeant et en exécutant un script de correctif. Pour ce faire, utilisez les commandes suivantes sur l'invite de commandes :

```
sudo curl https://d148y999krizvm.cloudfront.net/patch/v8/linux/patch.sh -o /var/
microfocuslicensing/bin/patch.sh
sudo chmod +x /var/microfocuslicensing/bin/patch.sh
sudo /var/microfocuslicensing/bin/patch.sh
```

```
sudo ./startmfcesd.sh
```

> **Note**
>
> Ce script de correctif fonctionnera également avec la version 9, même si le chemin de téléchargement correspond à la version 8.

Historique des documents pour le guide de l'utilisateur sur la modernisation du AWS mainframe

Le tableau suivant décrit les versions de documentation relatives à la modernisation des AWS mainframes.

Modification	Description	Date
AWS Notes de mise à jour 4.7.0 et 4.8.0 de Blu Age	La version 4.7.0 de AWS Blu Age Runtime et AWS Blu Age Transformation Engines inclut des mises à niveau des dépendances de base vers Spring Boot 3.4.2 et Angular 19, améliorant ainsi la sécurité et les performances tout en modernisant le framework applicatif frontal. La version 4.8.0 inclut des fonctionnalités clés telles que la prise en charge de la transformation des bases de données IBM IMS et une nouvelle fonctionn alité de cache en mémoire qui améliore les performances des programmes gourmands en données pour la modernisation de 00. AS4	18 mai 2025
AWS Notes de mise à jour de Blu Age 4.6.0	Cette version de AWS Blu Age Runtime et du moteur de transformation AWS Blu Age met l'accent sur les nouvelles fonctionnalités et améliorations pour zOS et AS4 00.	24 janvier 2025

AWS Notes de mise à jour de Blu Age 4.5.0	Cette version de AWS Blu Age Runtime et du moteur de transformation AWS Blu Age se concentre sur les fonctionn alités clés de la prise en charge de la JCL, de la prise en charge des répertoires et des groupes d'activation pour les applications modernisées AS/400 et des dépendances mises à jour.	20 décembre 2024
AWS FAQ sur Blu Age	Découvrez la capacité de refactorisation de AWS Blu Age grâce à cette liste complète de. FAQs	20 décembre 2024
Transformation Amazon Q Developer pour le mainframe	Vous pouvez en savoir plus sur la fonctionnalité Transform for mainframe d'Amazon Q Developer qui vous permet de moderniser vos anciennes applications mainframe COBOL en applications Java.	2 décembre 2024
AWS Notes de mise à jour de Blu Age 4.4.0	Cette version du moteur d'exécution et de transform ation AWS Blu Age est axée sur la mise à niveau des dépendances critiques et des technologies prises en charge tout en améliorant les performances de multiples fonctionnalités.	13 novembre 2024

Déployez AWS Blu Age Runtime sur des conteneurs	Découvrez comment configure r AWS Blu Age Runtime sur des conteneurs afin de le déployer sur Amazon ECS (géré par Amazon EC2 ou AWS Fargate) et Amazon EKS géré par Amazon EC2.	28 octobre 2024
Informations d'identification utilisateur LDAP mises à jour	Vous pouvez désormais créer et gérer les informati ons d'identification utilisateur LDAP pour l'authentification et l'autorisation à l'aide de AWS la console ou AWS CLI (ou du SDK).	21 octobre 2024
Configuration de l'application gérée par Rocket Software	Vous pouvez désormais configurer vos applications avec le moteur d'exécuti on Rocket Software pour personnaliser des propriétés supplémentaires, notamment les intégrations.	21 octobre 2024
Replateforme avec le runtime NTT DATA Unikix	Découvrez comment utiliser Amazon Machine Images (AMIs) pour créer un environnement personnalisé pour le réhébergement et la replateforme d'applications mainframe à l'aide de NTT DATA AWS .	25 septembre 2024

AWS Modernisation du mainframe | Tests d'applications (IAM)	Vous pouvez en apprendre davantage sur la gestion de l'accès pour les tests d'applications de modernisation des AWS mainframes grâce aux fonctionnalités et politiques IAM disponibles.	20 septembre 2024
AWS Sorties de Blu Age	Cette section du guide de l'utilisateur sur la modernisation du AWS mainframe présente tous les détails relatifs aux versions de AWS Blu Age, les notes de version de AWS Blu Age, les instructions de mise à niveau de AWS Blu Age et le cycle de vie global de AWS Blu Age.	16 septembre 2024
AWS Cycle de vie des composants de modernisation du mainframe	Cette page décrit le cycle de vie de chaque composant de modernisation du AWS mainframe, y compris les mises à niveau de version, le plan de publication global, ainsi que les plans de fin de support et de retrait.	5 septembre 2024
Conversion d'un assembleur avec mLogica	AWS La conversion du code de modernisation du mainframe avec mLogica est une fonctionnalité de modernisation du AWS mainframe qui convertit automatiquement le code assembleur du mainframe z/OS en COBOL.	22 juillet 2024

Version GA des tests d'applications	Documents de disponibilité générale pour les tests d'applications. AWS Les tests d'applications de modernisation du mainframe fournissent des tests d'équivalence fonctionnelle automatisés pour vos projets de migration. Cette version inclut une page de protection des données, des flux de travail de console et des mises à jour d'autres pages de documentation depuis la prévisualisation.	12 juin 2024
Tutoriel Managed Runtime pour Rocket Software mis à jour	Ce didacticiel explique comment déployer et exécuter l' CardDemo exemple d'application dans un environnement d'exécution géré AWS Mainframe Modernization avec le moteur d'exécution Rocket Software.	5 février 2024
Notes de publication pour la version 3.9.0 de AWS Blu Age Runtime and Modernization Tools.	Cette version de AWS Blu Age Runtime and Modernization Tools met l'accent sur de multiples améliorations transversales apportées au produit dans le but d'améliorer les performances des architectures à haute disponibilité, ainsi que sur de nouvelles fonctionnalités permettant d'améliorer l'exécution des tâches à un niveau supérieur.	18 décembre 2023

Transférez des fichiers entre le mainframe et AWS	Nouvelle fonctionnalité publiée pour transférer des fichiers du mainframe source vers AWS.	27 novembre 2023
Gérez les transactions pour les applications	Nouvelle fonctionnalité publiée pour afficher et modifier les transactions pour les applications de modernisation AWS du mainframe.	16 octobre 2023
Notes de publication pour la version 3.6.0 de AWS Blu Age Runtime and Modernization Tools.	Cette version de AWS Blu Age Runtime and Modernization Tools fournit de nouvelles fonctionnalités pour les migrations existantes vers zOS et AS4 00, principalement destinées à étendre les mécanismes de support CICS, à compléter les fonctionnalités JCL, à optimiser les performances des fonctionnalités simultanées et à volume élevé, et à ajouter des fonctionnalités. multi-data-source	4 août 2023
Vous pouvez désormais déployer une nouvelle version d'une application lorsque celle-ci est arrêtée.	Auparavant, pour déployer une nouvelle version d'une application, vous deviez supprimer la version déployée. Vous pouvez maintenant simplement arrêter la version déployée et déployer une nouvelle version.	26 juillet 2023

AWS Package d'exécution Blu Age pour faciliter le EC2 déploiement d'Amazon	AWS La modernisation du mainframe avec le runtime AWS Blu Age est désormais disponible avec plus de flexibilité pour configurer la pile complète et le déploiement sur les EC2 instances Amazon de votre Compte AWS ordinateur.	6 juillet 2023
Connexion unique à AWS Blu Age Blu Insights.	AWS Blu Age Blu Insights est disponible AWS Management Console via l'authentification unique.	31 mars 2023
Version GA	Publication générale du guide de l'utilisateur sur la modernisation du AWS mainframe.	8 juin 2022
Première version	Publication initiale (version préliminaire publique) du guide de l'utilisateur sur la modernisation du AWS mainframe.	30 novembre 2021

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.