Guide de l’utilisateur

AWS Amplify Hébergement

AWS Amplify Hébergement: Guide de l’utilisateur

Table of Contents

Bienvenue chez AWS Amplify Hosting

Amplify Hosting fournit un flux de travail basé sur Git pour héberger des applications Web sans serveur complètes avec un déploiement continu. Amplify déploie votre application sur le réseau AWS mondial de diffusion de contenu (CDN). Ce guide de l'utilisateur fournit les informations dont vous avez besoin pour démarrer avec Amplify Hosting.

Cadres pris en charge

Amplify Hosting prend en charge de nombreux frameworks SSR, frameworks d'applications monopages (SPA) et générateurs de sites statiques courants, notamment les suivants.

Cadres SSR

- Next.js
- Nuxt
- Astroavec un adaptateur communautaire
- SvelteKitavec un adaptateur communautaire
- Tout framework SSR avec un adaptateur personnalisé

Cadres SPA

- React
- Angular
- Vue.js
- Ionic
- Ember

Générateurs de sites statiques

- Eleventy
- Gatsby
- Hugo
- Jekyll

- VuePress

Fonctionnalités d'Amplify Hosting

Branches de fonctionnalités

Gérez les environnements de production et de préparation pour votre frontend et votre backend en connectant de nouvelles succursales.

Domaines personnalisés

Connectez votre application à un domaine personnalisé.

Aperçus par pull request

Prévisualisez les modifications lors de la révision du code.

End-to-end test

Améliorez la qualité de votre application grâce à end-to-end des tests.

Branches protégées par mot de passe

Protégez votre application web par un mot de passe pour pouvoir utiliser de nouvelles fonctions sans les rendre accessibles publiquement.

Redirections et réécritures

Configurez des réécritures et des redirections pour maintenir les classements SEO et acheminer le trafic en fonction des exigences de votre application cliente.

Déploiements atomiques

Les déploiements atomiques éliminent les fenêtres de maintenance en garantissant que votre application Web n'est mise à jour qu'une fois le déploiement complet terminé. Cela permet d'éliminer les scénarios dans lesquels les fichiers ne sont pas correctement chargés.

Commencer à utiliser Amplify Hosting

Pour démarrer avec Amplify Hosting, consultez le Commencer à déployer une application sur Amplify Hosting didacticiel. Après avoir terminé le didacticiel, vous saurez comment connecter une application Web dans un référentiel Git (GitHub, BitBucket GitLab, ou AWS CodeCommit) et la déployer sur Amplify Hosting avec un déploiement continu.

Création d'un backend

AWS Amplify Gen 2 introduit une expérience de développement TypeScript basée sur le code et axée sur le code pour définir les backends. Pour savoir comment utiliser Amplify Gen 2 pour créer et connecter un backend à votre application, consultez la section Créer et connecter un backend dans la documentation Amplify.

Pour mieux comprendre l'approche axée sur le code Gen 2 d'Amplify, consultez l'atelier Amplify Gen 2 sur le site Web de Workshop Studio.AWS Dans ce didacticiel complet, vous allez créer une application sans serveur avec React et Next.js et apprendre à utiliser les bibliothèques de données et d'authentification Amplify Gen 2 et la bibliothèque Amplify UI pour ajouter des fonctionnalités à l'application.

Si vous recherchez la documentation pour créer des backends pour une application Gen 1, à l'aide de la CLI et d'Amplify Studio, voir Build & connect backend dans la documentation Amplify Gen 1.

Tarifs d'Amplify Hosting

AWS Amplify ne vous facture que ce que vous utilisez. Pour plus d'informations, consultez Tarification d'AWS Amplify.

Commencer à déployer une application sur Amplify Hosting

Pour vous aider à comprendre le fonctionnement d'Amplify Hosting, les didacticiels suivants vous expliquent comment créer et déployer des applications créées à l'aide de frameworks SSR courants pris en charge par Amplify.

Tutoriels

- Déployer une application Next.js sur Amplify Hosting
- Déployer une application Nuxt.js sur Amplify Hosting
- Déployer une application Astro.js sur Amplify Hosting
- Déployer une SvelteKit application sur Amplify Hosting

Déployer une application Next.js sur Amplify Hosting

Ce didacticiel explique comment créer et déployer une application Next.js à partir d'un dépôt Git.

Avant de commencer ce didacticiel, remplissez les conditions préalables suivantes.

Inscrivez-vous pour un Compte AWS

Si vous n'êtes pas encore AWS client, vous devez en créer un Compte AWS en suivant les instructions en ligne. L'inscription vous permet d'accéder à Amplify et à d'autres AWS services que vous pouvez utiliser avec votre application.

Création d'une application

Créez une application Next.js de base à utiliser pour ce didacticiel, en create-next-appsuivant les instructions de la documentation Next.js.

Création d'un dépôt Git

Amplify prend en charge GitHub Bitbucket, et GitLab. AWS CodeCommit Transférez votre `create-next-app` application vers votre dépôt Git.

Étape 1 : Connecter un dépôt Git

Au cours de cette étape, vous connectez votre application Next.js dans un dépôt Git à Amplify Hosting.

Pour connecter une application dans un dépôt Git

1. Ouvrez la console Amplify.
2. Si vous déployez votre première application dans la région actuelle, vous partirez par défaut de la page de AWS Amplifyservice.

   Choisissez Déployer une application en haut de la page.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.

   Pour les GitHub référentiels, Amplify utilise GitHub la fonctionnalité Apps pour autoriser l'accès à Amplify. Pour plus d'informations sur l'installation et l'autorisation de l' GitHubapplication, consultezConfiguration de l'accès d'Amplify aux référentiels GitHub.

   > **Note**
   >
   > Après avoir autorisé la console Amplify avec Bitbucket, or GitLab, AWS CodeCommit Amplify récupère un jeton d'accès auprès du fournisseur de référentiel, mais ne le stocke pas sur les serveurs. AWS Amplify accède à votre référentiel à l'aide de clés de déploiement installées dans un référentiel spécifique.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :
   a. Sélectionnez le nom du référentiel à connecter.
   b. Sélectionnez le nom de la branche du référentiel à connecter.
   c. Choisissez Suivant.

Étape 2 : Confirmer les paramètres de construction

Amplify détecte automatiquement la séquence de commandes de génération à exécuter pour la branche que vous déployez. Au cours de cette étape, vous passez en revue et confirmez vos paramètres de compilation.

Pour confirmer les paramètres de génération d'une application

1. Sur la page des paramètres de l'application, recherchez la section Paramètres de création.

Vérifiez que la commande Frontend build et le répertoire de sortie Build sont corrects. Pour cet exemple d'application Next.js, le répertoire de sortie Build est défini sur `.next`.

2. La procédure d'ajout d'un rôle de service varie selon que vous souhaitez créer un nouveau rôle ou utiliser un rôle existant.
   - Pour créer un nouveau rôle :
     - Choisissez Créer et utiliser un nouveau rôle de service.
   - Pour utiliser un rôle existant :
     a. Choisissez Utiliser un rôle existant.
     b. Dans la liste des rôles de service, sélectionnez le rôle à utiliser.
3. Choisissez Suivant.

Étape 3 : déployer l'application

Au cours de cette étape, vous déployez votre application sur le réseau AWS mondial de diffusion de contenu (CDN).

Pour enregistrer et déployer une application

1. Sur la page de révision, vérifiez que les détails de votre référentiel et les paramètres de l'application sont corrects.
2. Choisissez Save and deploy (Enregistrer et déployer). La création de votre interface prend généralement 1 à 2 minutes, mais cela peut varier en fonction de la taille de l'application.
3. Lorsque le déploiement est terminé, vous pouvez consulter votre application à l'aide du lien vers le domaine `amplifyapp.com` par défaut.

> **Note**
>
> Pour renforcer la sécurité de vos applications Amplify, le domaine amplifyapp.com est enregistré dans la liste des suffixes publics (PSL). Pour plus de sécurité, nous vous recommandons d'utiliser des cookies avec un `__Host-` préfixe si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Amplify. Cette pratique vous aidera à protéger votre domaine contre les tentatives de falsification de

requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Étape 4 : (Facultatif) nettoyer les ressources

Si vous n'avez plus besoin de l'application que vous avez déployée pour le didacticiel, vous pouvez la supprimer. Cette étape permet de vous assurer de ne pas être facturé pour des ressources que vous n'utilisez pas.

Pour supprimer une application

1. Dans le menu des paramètres de l'application du volet de navigation, sélectionnez Paramètres généraux.
2. Sur la page des paramètres généraux, choisissez Supprimer l'application.
3. Dans la fenêtre de confirmation, entrez**`delete`**. Choisissez ensuite Supprimer l'application.

Ajoutez des fonctionnalités à votre application

Maintenant qu'une application est déployée sur Amplify, vous pouvez explorer certaines des fonctionnalités suivantes disponibles pour votre application hébergée.

Variables d'environnement

Les applications ont souvent besoin d'informations de configuration au moment de l'exécution. Ces configurations peuvent être des détails de connexion à la base de données, des clés d'API ou des paramètres. Les variables d'environnement permettent d'exposer ces configurations au moment de la création. Pour plus d'informations, consultez la section Variables d'environnement.

Domaines personnalisés

Dans ce didacticiel, Amplify héberge votre application pour vous sur le `amplifyapp.com` domaine par défaut avec une URL telle que. `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com` Lorsque vous connectez votre application à un domaine personnalisé, les utilisateurs voient que votre application est hébergée sur une URL personnalisée, telle que`https://www.example.com`. Pour plus d'informations, consultez la section Configuration de domaines personnalisés.

Aperçus par pull request

Les aperçus des pull requests Web permettent aux équipes de prévisualiser les modifications apportées par les pull requests (PRs) avant de fusionner le code avec une branche de production ou d'intégration. Pour plus d'informations, consultez les aperçus Web pour les pull requests.

Gérer plusieurs environnements

Pour savoir comment Amplify utilise les branches de fonctionnalités et les GitFlow flux de travail pour prendre en charge plusieurs déploiements, voir Déploiements de branches de fonctionnalités et flux de travail d'équipe.

Déployer une application Nuxt.js sur Amplify Hosting

Suivez les instructions suivantes pour déployer une application Nuxt.js sur Amplify Hosting. Nuxt a implémenté un adaptateur prédéfini à l'aide du serveur Nitro. Cela vous permet de déployer un projet Nuxt sans aucune configuration supplémentaire.

Pour déployer une application Nuxt sur Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :
   a. Sélectionnez le nom du référentiel à connecter.
   b. Sélectionnez le nom de la branche du référentiel à connecter.
   c. Choisissez Suivant.
5. Si vous souhaitez qu'Amplify soit en mesure de fournir des journaux d'applications à Amazon CloudWatch Logs, vous devez l'activer explicitement dans la console. Ouvrez la section Paramètres avancés, puis choisissez Activer les journaux d'applications SSR dans la section Déploiement du rendu côté serveur (SSR).
6. Choisissez Suivant.
7. Sur la page Révision, choisissez Enregistrer et déployer.

Déployer une application Astro.js sur Amplify Hosting

Suivez les instructions suivantes pour déployer une application Astro.js sur Amplify Hosting. Vous pouvez utiliser une application existante ou créer une application de démarrage en utilisant l'un des exemples officiels fournis par Astro. Pour créer une application de démarrage, voir Utiliser un thème ou un modèle de démarrage dans la documentation Astro.

Pour déployer un site Astro avec SSR sur Amplify Hosting, vous devez ajouter un adaptateur à votre application. Nous ne maintenons pas d'adaptateur appartenant à Amplify pour le framework Astro. Ce didacticiel utilise l'`astro-aws-amplify`adaptateur créé par un membre de la communauté. Cet adaptateur est disponible sur github. com/alexnguyennz/astro-aws-amplify sur le site Web. GitHub AWS ne gère pas cet adaptateur.

Pour déployer une application Astro sur Amplify Hosting

1. Sur votre ordinateur local, accédez à l'application Astro à déployer.
2. Pour installer l'adaptateur, ouvrez une fenêtre de terminal et exécutez la commande suivante. Cet exemple utilise l'adaptateur communautaire disponible sur github. com/alexnguyennz/astro-aws-amplifier. Vous pouvez le *`astro-aws-amplify`* remplacer par le nom de l'adaptateur que vous utilisez.

   ```
   npm install astro-aws-amplify
   ```

3. Dans le dossier de projet de votre application Astro, ouvrez le `astro.config.mjs` fichier. Mettez à jour le fichier pour ajouter l'adaptateur. Le fichier doit ressembler à ce qui suit.

   ```
   import { defineConfig } from 'astro/config';
   import mdx from '@astrojs/mdx';
   import awsAmplify from 'astro-aws-amplify';

   import sitemap from '@astrojs/sitemap';

   // https://astro.build/config
   export default defineConfig({
     site: 'https://example.com',
     integrations: [mdx(), sitemap()],
     adapter: awsAmplify(),
     output: 'server',
   });
   ```

4. Validez la modification et transférez le projet vers votre dépôt Git.

   Vous êtes maintenant prêt à déployer votre application Astro sur Amplify.

5. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
6. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
7. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
8. Sur la page Ajouter une branche de référentiel, procédez comme suit :

   a. Sélectionnez le nom du référentiel à connecter.

   b. Sélectionnez le nom de la branche du référentiel à connecter.

   c. Choisissez Suivant.

9. Sur la page des paramètres de l'application, recherchez la section Paramètres de création. Pour Construire le répertoire de sortie, entrez **`.amplify-hosting`**.
10. Vous devez également mettre à jour les commandes de génération du frontend de l'application dans la spécification de construction. Pour ouvrir la spécification de construction, choisissez Modifier le fichier YML.
11. Dans le `amplify.yml` fichier, recherchez la section des commandes de construction du frontend. Saisissez **`mv node_modules ./.amplify-hosting/compute/default`**.

    Votre fichier de paramètres de compilation doit ressembler à ce qui suit.

```
version: 1
frontend:
    phases:
        preBuild:
            commands:
                - 'npm ci --cache .npm --prefer-offline'
        build:
            commands:
                - 'npm run build'
                - 'mv node_modules ./.amplify-hosting/compute/default'
    artifacts:
        baseDirectory: .amplify-hosting
        files:
            - '**/*'
    cache:
        paths:
```

```
            - '.npm/**/*'
```

12. Choisissez Enregistrer.
13. Si vous souhaitez qu'Amplify soit en mesure de fournir des journaux d'applications à Amazon CloudWatch Logs, vous devez l'activer explicitement dans la console. Ouvrez la section Paramètres avancés, puis choisissez Activer les journaux d'applications SSR dans la section Déploiement du rendu côté serveur (SSR).
14. Choisissez Suivant.
15. Sur la page Révision, choisissez Enregistrer et déployer.

Déployer une SvelteKit application sur Amplify Hosting

Suivez les instructions suivantes pour déployer une SvelteKit application sur Amplify Hosting. Vous pouvez utiliser votre propre application ou créer une application de démarrage. Pour plus d'informations, consultez la section Création d'un projet dans la SvelteKit documentation.

Pour déployer une SvelteKit application avec SSR sur Amplify Hosting, vous devez ajouter un adaptateur à votre projet. Nous ne maintenons pas d'adaptateur appartenant à Amplify pour le SvelteKit framework. Dans cet exemple, nous utilisons le `amplify-adapter` créé par un membre de la communauté. L'adaptateur est disponible sur github. com/gzimbron/amplify-adaptateur sur le GitHub site Web. AWS ne gère pas cet adaptateur.

Pour déployer une SvelteKit application sur Amplify Hosting

1. Sur votre ordinateur local, accédez à l' SvelteKit application à déployer.
2. Pour installer l'adaptateur, ouvrez une fenêtre de terminal et exécutez la commande suivante. Cet exemple utilise l'adaptateur communautaire disponible sur github. com/gzimbron/amplify-adaptateur. Si vous utilisez un autre adaptateur communautaire, remplacez-le *`amplify-adapter`* par le nom de votre adaptateur.

   ```
   npm install amplify-adapter
   ```

3. Dans le dossier du projet de votre SvelteKit application, ouvrez le `svelte.config.js` fichier. Modifiez le fichier pour utiliser `amplify-adapter` ou remplacez-le *`'amplify-adapter'`* par le nom de votre adaptateur. Le fichier doit ressembler à ce qui suit.

   ```
   import adapter from 'amplify-adapter';
   ```

```
import { vitePreprocess } from '@sveltejs/vite-plugin-svelte';


/** @type {import('@sveltejs/kit').Config} */
const config = {
        // Consult https://kit.svelte.dev/docs/integrations#preprocessors
        // for more information about preprocessors
        preprocess: vitePreprocess(),

        kit: {
                // adapter-auto only supports some environments, see https://
kit.svelte.dev/docs/adapter-auto for a list.
                // If your environment is not supported, or you settled on a
 specific environment, switch out the adapter.
                // See https://kit.svelte.dev/docs/adapters for more information
 about adapters.
                adapter: adapter()
        }
};

export default config;
```

4. Validez la modification et transférez l'application vers votre dépôt Git.
5. Vous êtes maintenant prêt à déployer votre SvelteKit application sur Amplify.

   Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
6. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
7. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
8. Sur la page Ajouter une branche de référentiel, procédez comme suit :

   a. Sélectionnez le nom du référentiel à connecter.

   b. Sélectionnez le nom de la branche du référentiel à connecter.

   c. Choisissez Suivant.
9. Sur la page des paramètres de l'application, recherchez la section Paramètres de création. Pour Construire le répertoire de sortie, entrez**`build`**.
10. Vous devez également mettre à jour les commandes de génération du frontend de l'application dans la spécification de construction. Pour ouvrir la spécification de construction, choisissez Modifier le fichier YML.

11. Dans le `amplify.yml` fichier, recherchez la section des commandes de construction du frontend. Entrez **`- cd build/compute/default/`** et**`- npm i --production`**.

    Votre fichier de paramètres de compilation doit ressembler à ce qui suit.

```
version: 1
frontend:
    phases:
        preBuild:
            commands:
                - 'npm ci --cache .npm --prefer-offline'
        build:
            commands:
                - 'npm run build'
                - 'cd build/compute/default/'
                - 'npm i --production'

    artifacts:
        baseDirectory: build
        files:
            - '**/*'
    cache:
        paths:
            - '.npm/**/*'
```

12. Choisissez Enregistrer.
13. Si vous souhaitez qu'Amplify soit en mesure de fournir des journaux d'applications à Amazon CloudWatch Logs, vous devez l'activer explicitement dans la console. Ouvrez la section Paramètres avancés, puis choisissez Activer les journaux d'applications SSR dans la section Déploiement du rendu côté serveur (SSR).
14. Choisissez Suivant.
15. Sur la page Révision, choisissez Enregistrer et déployer.

Déploiement d'applications rendues côté serveur avec Amplify Hosting

Vous pouvez l'utiliser AWS Amplify pour déployer et héberger des applications Web qui utilisent le rendu côté serveur (SSR). Amplify Hosting détecte automatiquement les applications créées à l'aide du framework Next.js et vous n'avez pas à effectuer de configuration manuelle dans le. AWS Management Console

Amplify prend également en charge tout framework SSR basé sur Javascript avec un adaptateur de build open source qui transforme la sortie de compilation d'une application en la structure de répertoire attendue par Amplify Hosting. Par exemple, vous pouvez déployer des applications créées avec Nuxt, Astro et les SvelteKit frameworks en installant les adaptateurs disponibles.

Les utilisateurs avancés peuvent utiliser la spécification de déploiement pour créer un adaptateur de build ou configurer un script post-build.

Vous pouvez déployer les frameworks suivants sur Amplify Hosting avec une configuration minimale.

Next.js

- Amplify prend en charge les applications Next.js 15 sans avoir besoin d'un adaptateur. Consultez Amplify le support pour Next.js pour démarrer.

Nuxt.js

- Amplify prend en charge les déploiements d'applications Nuxt.js avec un adaptateur prédéfini. Consultez Amplify le support pour Nuxt.js pour démarrer.

Astro.js

- Amplify prend en charge les déploiements d'applications Astro.js avec un adaptateur communautaire. Consultez Amplify le support pour Astro.js pour démarrer.

SvelteKit

- Amplify prend en charge les déploiements SvelteKit d'applications avec un adaptateur communautaire. Consultez Amplify le support pour SvelteKit pour démarrer.

Adaptateurs open source

- Utiliser un adaptateur open source - Pour obtenir des instructions sur l'utilisation d'un adaptateur ne figurant pas dans la liste précédente, voirUtilisation d'adaptateurs open source pour n'importe quel framework SSR.

- Créez un adaptateur de framework - Les auteurs de framework qui souhaitent intégrer les fonctionnalités fournies par un framework peuvent utiliser la spécification de déploiement d'Amplify Hosting pour configurer votre sortie de build afin qu'elle soit conforme à la structure attendue par Amplify. Pour plus d'informations, consultez la section Utilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de build.
- Configurer un script post-build - Vous pouvez utiliser la spécification de déploiement d'Amplify Hosting pour manipuler la sortie de votre build selon les besoins de scénarios spécifiques. Pour plus d'informations, consultez la section Utilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de build. Pour obtenir un exemple, consultez Déploiement d'un serveur Express à l'aide du manifeste de déploiement.

Rubriques

Amplify le support pour Next.js

Amplify prend en charge le déploiement et l'hébergement d'applications Web rendues côté serveur (SSR) créées à l'aide de Next.js. Next.js est un framework React pour développer SPAs avec JavaScript. Vous pouvez déployer des applications créées avec des versions de Next.js jusqu'à Next.js 15, avec des fonctionnalités telles que l'optimisation des images et le middleware.

Les développeurs peuvent utiliser Next.js pour combiner la génération de sites statiques (SSG) et le SSR dans un seul projet. Les pages SSG sont prérendues au moment de la création, et les pages SSR sont prérendues au moment de la demande.

Le prérendu peut améliorer les performances et l'optimisation des moteurs de recherche. Comme Next.js préaffiche toutes les pages du serveur, le contenu HTML de chaque page est prêt lorsqu'il

atteint le navigateur du client. Ce contenu peut également être chargé plus rapidement. Des temps de chargement plus rapides améliorent l'expérience de l'utilisateur final avec un site Web et ont un impact positif sur le classement SEO du site. Le pré-rendu améliore également le référencement en permettant aux robots des moteurs de recherche de trouver et d'explorer facilement le contenu HTML d'un site Web.

Next.js fournit un support analytique intégré pour mesurer divers indicateurs de performance, tels que le délai jusqu'au premier octet (TTFB) et le premier contenu de peinture (FCP). Pour plus d'informations sur Next.js, consultez Getting started on the Next.js website.

Support des fonctionnalités de Next.js

Amplify Hosting Compute gère entièrement le rendu côté serveur (SSR) pour les applications créées avec les versions 12 à 15 de Next.js.

Si vous avez déployé une application Next.js sur Amplify avant la sortie d'Amplify Hosting Compute en novembre 2022, votre application utilise l'ancien fournisseur SSR d'Amplify, Classic (Next.js 11 uniquement). Amplify Hosting Compute ne prend pas en charge les applications créées à l'aide de Next.js version 11 ou antérieure. Nous vous recommandons vivement de migrer vos applications Next.js 11 vers le fournisseur SSR géré par le calcul Amplify Hosting.

La liste suivante décrit les fonctionnalités spécifiques prises en charge par le fournisseur de SSR de calcul Amplify Hosting.

Fonctionnalités prises en charge

- Pages rendues côté serveur (SSR)
- Pages statiques
- Routes d'API
- Routes dynamiques
- Suivez tous les itinéraires
- SSG (génération statique)
- Régénération statique incrémentielle (ISR)
- Routage de sous-chemins internationalisé (i18n)
- Routage de domaine internationalisé (i18n)
- Détection automatique des paramètres régionaux internationalisée (i18n)

- Intergiciel
- Variables d’environnement
- Optimisation de l'image
- Répertoire de l'application Next.js 13

Fonctions non prises en charge

- Routes d'API Edge (le middleware Edge n'est pas pris en charge)
- Régénération statique incrémentielle (ISR) à la demande
- Diffusion de Next.js
- Exécution d'un intergiciel sur des actifs statiques et des images optimisées
- Exécution de code après une réponse avec `unstable_after` (fonctionnalité expérimentale publiée avec Next.js 15)

Images du fichier Next.js

La taille de sortie maximale d'une image ne doit pas dépasser 4,3 Mo. Vous pouvez stocker un fichier image plus volumineux quelque part et utiliser le composant Image Next.js pour le redimensionner et l'optimiser au format Webp ou AVIF, puis l'utiliser dans une taille plus petite.

Notez que la documentation Next.js vous conseille d'installer le module de traitement d'image Sharp pour permettre à l'optimisation des images de fonctionner correctement en production. Toutefois, cela n'est pas nécessaire pour les déploiements d'Amplify. Amplify déploie automatiquement Sharp pour vous.

Déploiement d'une application SSR Next.js sur Amplify

Par défaut, Amplify déploie de nouvelles applications SSR à l'aide du service de calcul d'Amplify Hosting avec prise en charge des versions 12 à 15 de Next.js. Amplify Hosting Compute gère entièrement les ressources nécessaires au déploiement d'une application SSR. Les applications SSR de votre compte Amplify que vous avez déployées avant le 17 novembre 2022 utilisent le fournisseur SSR Classic (Next.js 11 uniquement).

Nous vous recommandons vivement de migrer les applications utilisant le SSR classique (Next.js 11 uniquement) vers le fournisseur de SSR de calcul Amplify Hosting. Amplify n'effectue pas de migrations automatiques pour vous. Vous devez migrer manuellement votre application, puis lancer

une nouvelle version pour terminer la mise à jour. Pour obtenir des instructions, veuillez consulter Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute.

Suivez les instructions ci-dessous pour déployer une nouvelle application Next.js SSR.

Pour déployer une application SSR sur Amplify à l'aide du fournisseur de calcul SSR d'Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :
   a. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.
   b. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.
   c. Choisissez Suivant.
5. L'application nécessite un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.
   - Pour permettre à Amplify de créer automatiquement un rôle et de l'associer à votre application :
     - Choisissez Créer et utiliser un nouveau rôle de service.
   - Pour associer un rôle de service que vous avez créé précédemment :
     a. Choisissez Utiliser un rôle de service existant.
     b. Sélectionnez le rôle à utiliser dans la liste.
6. Choisissez Suivant.
7. Sur la page Révision, choisissez Enregistrer et déployer.

Paramètres du fichier Package.json

Lorsque vous déployez une application Next.js, Amplify inspecte le script de génération de l'application dans le `package.json` fichier pour déterminer le type d'application.

Voici un exemple de script de génération pour une application Next.js. Le script de compilation "`next build`" indique que l'application prend en charge les pages SSG et SSR. Ce script de génération est également utilisé pour les applications SSG de Next.js 14 ou version ultérieure uniquement.

```
"scripts": {
  "dev": "next dev",
  "build": "next build",
  "start": "next start"
},
```

Voici un exemple de script de génération pour une application SSG Next.js 13 ou antérieure. Le script de génération "`next build && next export`" indique que l'application ne prend en charge que les pages SSG.

```
"scripts": {
  "dev": "next dev",
  "build": "next build && next export",
  "start": "next start"
},
```

Amplifier les paramètres de compilation pour une application Next.js SSR

Après avoir inspecté le `package.json` fichier de votre application, Amplify vérifie les paramètres de compilation de l'application. Vous pouvez enregistrer les paramètres de compilation dans la console Amplify ou dans un `amplify.yml` fichier à la racine de votre référentiel. Pour de plus amples informations, veuillez consulter Configuration des paramètres de compilation pour une application Amplify.

Si Amplify détecte que vous déployez une application SSR Next.js et qu'aucun `amplify.yml` fichier n'est présent, il génère une spécification de construction pour l'application et la définit sur. `baseDirectory .next` Si vous déployez une application dans laquelle un `amplify.yml` fichier est présent, les paramètres de génération du fichier remplacent ceux de la console. Par conséquent, vous devez définir manuellement le `baseDirectory` to `.next` dans le fichier.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur`.next`. Cela indique que les artefacts de construction sont destinés à une application Next.js qui prend en charge les pages SSG et SSR.

```
version: 1
frontend:
```

```
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Amplifier les paramètres de compilation pour une application SSG Next.js 13 ou antérieure

Si Amplify détecte que vous déployez une application SSG Next.js 13 ou antérieure, il génère une spécification de build pour l'application et la définit sur. `baseDirectory` `out` Si vous déployez une application dans laquelle un `amplify.yml` fichier est présent, vous devez définir manuellement le `baseDirectory` to `out` dans le fichier. Le `out` répertoire est le dossier par défaut créé par Next.js pour stocker les actifs statiques exportés. Lorsque vous configurez les paramètres de spécification de build de votre application, modifiez le nom du `baseDirectory` dossier pour qu'il corresponde à la configuration de votre application.

Voici un exemple des paramètres de génération d'une application où il `baseDirectory` est défini `out` pour indiquer que les artefacts de génération concernent une application Next.js 13 ou antérieure qui ne prend en charge que les pages SSG.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: out
```

```
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Amplifier les paramètres de compilation pour une application SSG Next.js 14 ou version ultérieure

Dans la version 14 de Next.js, la `next export` commande était obsolète et remplacée par `output: 'export'` dans le `next.config.js` fichier pour activer les exportations statiques. Si vous déployez une application Next.js 14 SSG uniquement dans la console, Amplify génère une spécification de construction pour l'application et la définit sur. `baseDirectory .next` Si vous déployez une application dans laquelle un `amplify.yml` fichier est présent, vous devez définir manuellement le `baseDirectory` to `.next` dans le fichier. Il s'agit du même `baseDirectory` paramètre qu'Amplify utilise pour les `WEB_COMPUTE` applications Next.js qui prennent en charge les pages SSG et SSR.

Voici un exemple des paramètres de génération pour une application Next.js 14 SSG uniquement avec le paramètre `baseDirectory` défini sur. `.next`

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute

Lorsque vous déployez une nouvelle application Next.js, Amplify utilise par défaut la dernière version prise en charge de Next.js. Actuellement, le fournisseur de SSR de calcul Amplify Hosting prend en charge la version 15 de Next.js.

La console Amplify détecte les applications de votre compte qui ont été déployées avant la sortie de novembre 2022 du service de calcul Amplify Hosting avec prise en charge complète des versions 12 à 15 de Next.js. La console affiche une bannière d'information identifiant les applications dotées de branches déployées à l'aide de l'ancien fournisseur SSR d'Amplify, Classic (Next.js 11 uniquement). Nous vous recommandons vivement de migrer vos applications vers le fournisseur de calcul SSR d'Amplify Hosting.

Si vous mettez à jour votre application hébergée Next.js 11 vers Next.js 12 ou version ultérieure, un `"target" property is no longer supported` message d'erreur peut s'afficher lorsqu'un déploiement est déclenché. Dans ce cas, vous devez migrer vers Amplify Hosting Compute.

Vous devez migrer manuellement l'application et toutes ses branches de production en même temps. Une application ne peut pas contenir à la fois des branches Classic (Next.js 11 uniquement) et Next.js 12 ou version ultérieure.

Suivez les instructions suivantes pour migrer une application vers le fournisseur de calcul SSR d'Amplify Hosting.

Pour migrer une application vers le fournisseur de calcul SSR d'Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application Next.js que vous souhaitez migrer.

   > **Note**
   >
   > Avant de migrer une application dans la console Amplify, vous devez d'abord mettre à jour le fichier package.json de l'application pour utiliser Next.js version 12 ou ultérieure.

3. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
4. Sur la page d'accueil de l'application, la console affiche une bannière si l'application possède des branches déployées à l'aide du fournisseur SSR Classic (Next.js 11 uniquement). Sur la bannière, choisissez Migrer.

5. Dans la fenêtre de confirmation de la migration, sélectionnez les trois instructions et choisissez Migrer.
6. Amplify créera et redéploiera votre application pour terminer la migration.

Annulation d'une migration SSR

Lorsque vous déployez une application Next.js, Amplify Hosting détecte les paramètres de votre application et définit la valeur de plate-forme interne de l'application. Il existe trois valeurs de plateforme valides. Une application SSG est définie sur la valeur `WEB` de la plateforme. Une application SSR utilisant Next.js version 11 est définie sur la valeur `WEB_DYNAMIC` de la plateforme. Une application SSR de Next.js 12 ou version ultérieure est définie sur la valeur `WEB_COMPUTE` de la plateforme.

Lorsque vous migrez une application en suivant les instructions de la section précédente, Amplify change la valeur de plateforme de votre application de à`WEB_DYNAMIC`. `WEB_COMPUTE` Une fois la migration vers Amplify Hosting terminée, vous ne pouvez pas annuler la migration dans la console. Pour annuler la migration, vous devez utiliser le AWS Command Line Interface pour redéfinir la plateforme de l'application. `WEB_DYNAMIC` Ouvrez une fenêtre de terminal et entrez la commande suivante pour mettre à jour l'ID de l'application et la région avec vos informations uniques.

```
aws amplify update-app --app-id abcd1234 --platform WEB_DYNAMIC --region us-west-2
```

Ajout de la fonctionnalité SSR à une application Next.js statique

Vous pouvez ajouter la fonctionnalité SSR à une application statique (SSG) Next.js existante déployée avec Amplify. Avant de commencer le processus de conversion de votre application SSG en SSR, mettez-la à jour pour utiliser Next.js version 12 ou ultérieure et ajoutez la fonctionnalité SSR. Ensuite, vous devrez effectuer les étapes suivantes.

1. Utilisez le AWS Command Line Interface pour modifier le type de plateforme de l'application.
2. Ajoutez un rôle de service à l'application.
3. Mettez à jour le répertoire de sortie dans les paramètres de compilation de l'application.
4. Mettez à jour le `package.json` fichier de l'application pour indiquer que celle-ci utilise le SSR.

Mettre à jour la plateforme

Il existe trois valeurs valides pour le type de plateforme. Une application SSG est définie sur le type `WEB` de plateforme. Une application SSR utilisant Next.js version 11 est définie sur le type `WEB_DYNAMIC` de plate-forme. Pour les applications déployées sur Next.js 12 ou version ultérieure à l'aide du SSR géré par Amplify Hosting Compute, le type de plate-forme est défini sur. `WEB_COMPUTE`

Lorsque vous avez déployé votre application en tant qu'application SSG, Amplify a défini le type de plateforme sur. `WEB` Utilisez le AWS CLI pour modifier la plateforme de votre application`WEB_COMPUTE`. Ouvrez une fenêtre de terminal et entrez la commande suivante, en mettant à jour le texte en rouge avec votre identifiant d'application unique et votre région.

```
aws amplify update-app --app-id abcd1234 --platform WEB_COMPUTE --region us-west-2
```

Ajouter un rôle de service

Un rôle de service est le rôle Gestion des identités et des accès AWS (IAM) qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Suivez ces étapes pour ajouter un rôle de service à une application SSG déjà déployée avec Amplify.

Pour ajouter un rôle de service

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Si vous n'avez pas encore créé de rôle de service dans votre compte Amplify, consultez Ajouter un rôle de service pour terminer cette étape préalable.
3. Choisissez l'application statique Next.js à laquelle vous souhaitez ajouter un rôle de service.
4. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
5. Sur la page des détails de l'application, choisissez Modifier
6. Pour Rôle de service, choisissez le nom d'un rôle de service existant ou le nom du rôle de service que vous avez créé à l'étape 2.
7. Choisissez Enregistrer.

Mettre à jour les paramètres de compilation

Avant de redéployer votre application avec la fonctionnalité SSR, vous devez mettre à jour les paramètres de compilation de l'application afin de définir le répertoire de sortie sur. `.next` Vous

pouvez modifier les paramètres de compilation dans la console Amplify ou dans un `amplify.yml` fichier stocké dans votre dépôt. Pour plus d'informations, voir,Configuration des paramètres de compilation pour une application Amplify.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur`.next`.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Mettre à jour le fichier package.json

Après avoir ajouté un rôle de service et mis à jour les paramètres de compilation, mettez à jour le `package.json` fichier de l'application. Comme dans l'exemple suivant, définissez le script de génération sur `"next build"` pour indiquer que l'application Next.js prend en charge les pages SSG et SSR.

```
"scripts": {
  "dev": "next dev",
  "build": "next build",
  "start": "next start"
},
```

Amplify détecte la modification du `package.json` fichier dans votre dépôt et redéploie l'application avec la fonctionnalité SSR.

Rendre les variables d'environnement accessibles aux environnements d'exécution côté serveur

Amplify Hosting prend en charge l'ajout de variables d'environnement aux versions de votre application en les définissant dans la configuration du projet dans la console Amplify.

Cependant, un composant serveur Next.js n'a pas accès à ces variables d'environnement par défaut. Ce comportement est intentionnel pour protéger les secrets stockés dans les variables d'environnement que votre application utilise pendant la phase de génération.

Pour rendre des variables d'environnement spécifiques accessibles à Next.js, vous pouvez modifier le fichier de spécification de build Amplify afin de les définir dans les fichiers d'environnement reconnus par Next.js. Cela permet à Amplify de charger ces variables d'environnement avant de créer l'application.

> **Important**
>
> Nous vous recommandons vivement de ne pas stocker d'informations d'identification, de secrets ou d'informations sensibles dans vos variables d'environnement, car tout utilisateur ayant accès aux artefacts de déploiement peut les lire.
> Pour permettre à votre fonction de calcul SSR d'accéder aux AWS ressources, nous vous recommandons d'utiliser des rôles IAM.

L'exemple de spécification de construction suivant montre comment ajouter des variables d'environnement dans la section des commandes de construction.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - env | grep -e API_BASE_URL >> .env.production
        - env | grep -e NEXT_PUBLIC_ >> .env.production
        - npm run build
  artifacts:
```

```
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
      - .next/cache/**/*
```

Dans cet exemple, la section des commandes de génération inclut deux commandes qui écrivent des variables d'environnement dans le `.env.production` fichier avant l'exécution de la compilation de l'application. Amplify Hosting permet à votre application d'accéder à ces variables lorsque l'application reçoit du trafic.

La ligne suivante, tirée de la section des commandes de génération de l'exemple précédent, montre comment prendre une variable spécifique de l'environnement de génération et l'ajouter au `.env.production` fichier.

```
- env | grep -e API_BASE_URL -e APP_ENV >> .env.production
```

Si les variables existent dans votre environnement de génération, le `.env.production` fichier contiendra les variables d'environnement suivantes.

```
API_BASE_URL=localhost
APP_ENV=dev
```

La ligne suivante, tirée de la section des commandes de construction de l'exemple précédent, montre comment ajouter une variable d'environnement avec un préfixe spécifique au `.env.production` fichier. Dans cet exemple, toutes les variables avec le préfixe `NEXT_PUBLIC_` sont ajoutées.

```
- env | grep -e NEXT_PUBLIC_ >> .env.production
```

Si plusieurs variables avec le `NEXT_PUBLIC_` préfixe existent dans l'environnement de construction, le `.env.production` fichier ressemblera à ce qui suit.

```
NEXT_PUBLIC_ANALYTICS_ID=abcdefghijk
NEXT_PUBLIC_GRAPHQL_ENDPOINT=uowelalsmlsadf
NEXT_PUBLIC_FEATURE_FLAG=true
```

Variables d'environnement SSR pour monorepos

Si vous déployez une application SSR dans un monorepo et que vous souhaitez rendre des variables d'environnement spécifiques accessibles à Next.js, vous devez préfixer le `.env.production` fichier avec la racine de votre application. L'exemple de spécification de construction suivant pour une application Next.js dans un monorepo Nx montre comment ajouter des variables d'environnement dans la section des commandes de génération.

```
version: 1
applications:
  - frontend:
      phases:
        preBuild:
          commands:
            - npm ci
        build:
          commands:
            - env | grep -e API_BASE_URL -e APP_ENV >> apps/app/.env.production
            - env | grep -e NEXT_PUBLIC_ >> apps/app/.env.production
            - npx nx build app
      artifacts:
        baseDirectory: dist/apps/app/.next
        files:
          - '**/*'
      cache:
        paths:
          - node_modules/**/*
      buildPath: /
    appRoot: apps/app
```

Les lignes suivantes de la section des commandes de génération de l'exemple précédent montrent comment prendre des variables spécifiques de l'environnement de génération et les ajouter au `.env.production` fichier d'une application dans un monorepo avec la racine de l'application. `apps/app`

```
- env | grep -e API_BASE_URL -e APP_ENV >> apps/app/.env.production
- env | grep -e NEXT_PUBLIC_ >> apps/app/.env.production
```

Déploiement d'une application Next.js dans un monorepo

Amplify prend en charge les applications en monorepos génériques ainsi que les applications en monorepos créées à l'aide de npm workspace, pnpm workspace, Yarn workspace, Nx et Turborepo. Lorsque vous déployez votre application, Amplify détecte automatiquement le framework de construction monorepo que vous utilisez. Amplify applique automatiquement les paramètres de génération pour les applications dans un espace de travail npm, un espace de travail Yarn ou Nx. Les applications Turborepo et pnpm nécessitent une configuration supplémentaire. Pour de plus amples informations, veuillez consulter Configuration des paramètres de compilation de monorepo.

Pour un exemple détaillé de Nx, consultez le billet de blog Share code between Next.js apps with Nx on AWS Amplify Hosting.

Amplify le support pour Nuxt.js

Nuxt est un framework permettant de créer des applications Web complètes avec Vue.js.

adaptateur

Vous pouvez déployer une application Nuxt.js sur Amplify à l'aide d'un adaptateur prédéfini sans configuration. Pour plus d'informations sur l'adaptateur, consultez la documentation de Nuxt.

didacticiel

Pour savoir comment déployer une application Nuxt.js sur Amplify, consultez. Déployer une application Nuxt.js sur Amplify Hosting

Démonstration

Pour une démonstration vidéo, voir Nuxt Hosting With ZERO Configuration in Minutes (With AWS) on YouTube.

Amplify le support pour Astro.js

Astro est un framework Web permettant de créer des applications Web axées sur le contenu.

adaptateur

Vous pouvez déployer une application Astro.js sur Amplify à l'aide d'un adaptateur communautaire. Nous ne maintenons pas d'adaptateur appartenant à Amplify pour le framework Astro. Cependant, un adaptateur est disponible sur github. com/alexnguyennz/astro-aws-amplify sur le site Web. GitHub Cet adaptateur a été créé par un membre de la communauté et n'est pas maintenu par AWS.

didacticiel

Pour savoir comment déployer une application Astro sur Amplify, consultez. Déployer une application Astro.js sur Amplify Hosting

Démonstration

Pour une démonstration vidéo, consultez Comment déployer un site Web Astro AWS sur le YouTube canal Amazon Web Services.

Amplify le support pour SvelteKit

SvelteKit est un framework permettant de créer des applications Web complètes avec Svelte.

adaptateur

Vous pouvez déployer une SvelteKit application sur Amplify à l'aide d'un adaptateur communautaire. Nous ne maintenons pas d'adaptateur appartenant à Amplify pour le SvelteKit framework. Cependant, un adaptateur est disponible sur github. com/gzimbron/amplify-adaptateur sur le GitHub site Web. Cet adaptateur a été créé par un membre de la communauté et n'est pas maintenu par AWS.

didacticiel

Pour savoir comment déployer une SvelteKit application sur Amplify, consultez. Déployer une SvelteKit application sur Amplify Hosting

Démonstration

Pour une démonstration vidéo, consultez Comment déployer un SvelteKit site Web (avec API) AWS sur le YouTube canal Amazon Web Services.

Déploiement d'une application SSR sur Amplify

Vous pouvez utiliser ces instructions pour déployer une application créée avec n'importe quel framework avec un bundle de déploiement conforme à la sortie de compilation attendue par Amplify. Si vous déployez une application Next.js, aucun adaptateur n'est nécessaire.

Si vous déployez une application SSR qui utilise un adaptateur framework, vous devez d'abord installer et configurer l'adaptateur. Pour obtenir des instructions, veuillez consulter Utilisation d'adaptateurs open source pour n'importe quel framework SSR.

Pour déployer une application SSR sur Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :

    a. Sélectionnez le nom du référentiel à connecter.

b. Sélectionnez le nom de la branche du référentiel à connecter.

c. Choisissez Suivant.

5. Sur la page des paramètres de l'application, Amplify détecte automatiquement les applications Next.js SSR.

   Si vous déployez une application SSR qui utilise un adaptateur pour un autre framework, vous devez activer Amazon CloudWatch Logs de manière explicite. Ouvrez la section Paramètres avancés, puis choisissez Activer les journaux d'applications SSR dans la section Déploiement du rendu côté serveur (SSR).

6. L'application nécessite un rôle de service IAM qu'Amplify assume pour vous fournir des journaux. Compte AWS

   La procédure d'ajout d'un rôle de service varie selon que vous souhaitez créer un nouveau rôle ou utiliser un rôle existant.

   - Pour créer un nouveau rôle :

     - Choisissez Créer et utiliser un nouveau rôle de service.

   - Pour utiliser un rôle existant, procédez comme suit :

     a. Choisissez Utiliser un rôle existant.

     b. Dans la liste des rôles de service, sélectionnez le rôle à utiliser.

7. Choisissez Suivant.

8. Sur la page Révision, choisissez Enregistrer et déployer.

Fonctionnalités prises en charge par le SSR

Cette section fournit des informations sur le support d'Amplify pour les fonctionnalités SSR.

Amplify fournit une prise en charge de la version de Node.js correspondant à la version de Node.js utilisée pour créer votre application.

Amplify fournit une fonction d'optimisation d'image intégrée qui prend en charge toutes les applications SSR. Si vous ne souhaitez pas utiliser la fonctionnalité d'optimisation d'image par défaut, vous pouvez implémenter un chargeur d'optimisation d'image personnalisé.

Rubriques

- Optimisation de l'image pour les applications SSR
- Amazon CloudWatch Logs pour les applications SSR
- Prise en charge du SSR Amplify Next.js 11

Prise en charge de la version Node.js pour les applications Next.js

Lorsqu'Amplify crée et déploie une application de calcul Next.js, elle utilise la version Node.js d'exécution qui correspond à la version principale de Node.js celle utilisée pour créer l'application.

> **Note**
>
> À compter du 15 septembre 2025, l'hébergement Amplify ne prendra plus en charge les environnements d'exécution Node.js 14, Node.js 16 et Node.js 18. Les environnements d'exécution pris en charge incluent Node.js 20 et Node.js 22.

Vous pouvez spécifier la Node.js version à utiliser dans la fonction de remplacement du package Live de la console Amplify. Pour plus d'informations sur la configuration des mises à jour des packages en direct, consultezUtilisation de versions de package et de dépendance spécifiques dans l'image de construction. Vous pouvez également spécifier la Node.js version à l'aide d'autres mécanismes, tels que nvm des commandes. Si vous ne spécifiez pas de version, Amplify utilise par défaut la version actuelle utilisée par le conteneur de compilation Amplify.

Optimisation de l'image pour les applications SSR

Amplify Hosting fournit une fonctionnalité d'optimisation d'image intégrée qui prend en charge toutes les applications SSR. Grâce à l'optimisation d'image d'Amplify, vous pouvez fournir des images de haute qualité au format, aux dimensions et à la résolution adaptés à l'appareil qui y accède, tout en conservant la plus petite taille de fichier possible.

Actuellement, vous pouvez soit utiliser le composant Image Next.js pour optimiser les images à la demande, soit implémenter un chargeur d'images personnalisé. Si vous utilisez Next.js 13 ou une version ultérieure, vous n'avez aucune autre action à effectuer pour utiliser la fonction d'optimisation d'image d'Amplify. Si vous implémentez un chargeur personnalisé, consultez la rubrique suivante sur l'utilisation d'un chargeur d'images personnalisé.

Utilisation d'un chargeur d'images personnalisé

Si vous utilisez un chargeur d'image personnalisé, Amplify détecte le chargeur dans le `next.config.js` fichier de votre application et n'utilise pas la fonction d'optimisation d'image intégrée. Pour plus d'informations sur les chargeurs personnalisés pris en charge par Next.js, consultez la documentation relative aux images Next.js.

Amazon CloudWatch Logs pour les applications SSR

Amplify envoie des informations sur votre environnement d'exécution SSR à Amazon CloudWatch Logs dans votre. Compte AWS Lorsque vous déployez une application SSR, celle-ci nécessite un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.

Si vous choisissez d'autoriser Amplify à créer un rôle IAM pour vous, le rôle aura déjà les autorisations nécessaires pour créer des journaux. CloudWatch Si vous créez votre propre rôle IAM, vous devrez ajouter les autorisations suivantes à votre politique pour permettre à Amplify d'accéder à Amazon CloudWatch Logs.

```
logs:CreateLogStream
logs:CreateLogGroup
logs:DescribeLogGroups
logs:PutLogEvents
```

Pour de plus amples informations sur les rôles de service, veuillez consulter Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend.

Prise en charge du SSR Amplify Next.js 11

Si vous avez déployé une application Next.js sur Amplify avant la sortie d'Amplify Hosting Compute le 17 novembre 2022, votre application utilise l'ancien fournisseur SSR d'Amplify, Classic (Next.js 11 uniquement). La documentation de cette section s'applique uniquement aux applications déployées à l'aide du fournisseur SSR Classic (Next.js 11 uniquement).

> **Note**
>
> Nous vous recommandons vivement de migrer vos applications Next.js 11 vers le fournisseur SSR géré par le calcul Amplify Hosting. Pour de plus amples informations, veuillez consulter Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute.

La liste suivante décrit les fonctionnalités spécifiques prises en charge par le fournisseur SSR Amplify Classic (Next.js 11 uniquement).

Fonctionnalités prises en charge

- Pages rendues côté serveur (SSR)
- Pages statiques
- Routes d'API
- Routes dynamiques
- Suivez tous les itinéraires
- SSG (génération statique)
- Régénération statique incrémentielle (ISR)
- Routage de sous-chemins internationalisé (i18n)
- Variables d'environnement

Fonctions non prises en charge

- Optimisation de l'image
- Régénération statique incrémentielle (ISR) à la demande
- Routage de domaine internationalisé (i18n)
- Détection automatique des paramètres régionaux internationalisée (i18n)
- Intergiciel
- Intergiciel Edge
- Routes de l'API Edge

Tarification des applications Next.js 11 SSR

Lors du déploiement de votre application Next.js 11 SSR, Amplify crée des ressources backend supplémentaires dans AWS votre compte, notamment :

- Un bucket Amazon Simple Storage Service (Amazon S3) qui stocke les ressources des actifs statiques de votre application. Pour plus d'informations sur les frais d'Amazon S3, consultez la section Tarification d'Amazon S3.
- Une CloudFront distribution Amazon pour diffuser l'application. Pour plus d'informations sur CloudFront les frais, consultez Amazon CloudFront Pricing.
- Quatre fonctions Lambda @Edge pour personnaliser le contenu diffusé. CloudFront

Gestion des identités et des accès AWS autorisations pour les applications Next.js 11 SSR

Amplify nécessite des autorisations Gestion des identités et des accès AWS (IAM) pour déployer une application SSR. Pour les applications SSR, Amplify déploie des ressources telles qu'un compartiment Amazon S3, CloudFront une distribution, des fonctionsLambda@Edge, une file d'attente Amazon SQS (si vous utilisez l'ISR) et des rôles IAM. Si vous ne disposez pas des autorisations minimales requises, vous recevrez un `Access Denied` message d'erreur lorsque vous tenterez de déployer votre application SSR. Pour fournir à Amplify les autorisations requises, vous devez spécifier un rôle de service.

Pour créer un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom, voir. Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend Ces instructions montrent comment créer un rôle qui associe la politique `AdministratorAccess-Amplify` gérée.

La politique `AdministratorAccess-Amplify` gérée donne accès à de multiples AWS services, y compris aux actions IAM, et doit être considérée comme aussi puissante que la `AdministratorAccess` politique. Cette politique fournit plus d'autorisations que ce qui est nécessaire pour déployer votre application SSR.

Il est recommandé de suivre la meilleure pratique consistant à accorder le moindre privilège et à réduire les autorisations accordées au rôle de service. Au lieu d'accorder des autorisations d'accès d'administrateur à votre rôle de service, vous pouvez créer votre propre politique IAM gérée par le client qui accorde uniquement les autorisations requises pour déployer votre application SSR.

Reportez-vous à la section Création de politiques IAM dans le Guide de l'utilisateur IAM pour obtenir des instructions sur la création d'une politique gérée par le client.

Si vous créez votre propre politique, reportez-vous à la liste suivante des autorisations minimales requises pour déployer une application SSR.

```
acm:DescribeCertificate
acm:DescribeCertificate
acm:ListCertificates
acm:RequestCertificate
cloudfront:CreateCloudFrontOriginAccessIdentity
cloudfront:CreateDistribution
cloudfront:CreateInvalidation
cloudfront:GetDistribution
cloudfront:GetDistributionConfig
cloudfront:ListCloudFrontOriginAccessIdentities
cloudfront:ListDistributions
cloudfront:ListDistributionsByLambdaFunction
cloudfront:ListDistributionsByWebACLId
cloudfront:ListFieldLevelEncryptionConfigs
cloudfront:ListFieldLevelEncryptionProfiles
cloudfront:ListInvalidations
cloudfront:ListPublicKeys
cloudfront:ListStreamingDistributions
cloudfront:UpdateDistribution
cloudfront:TagResource
cloudfront:UntagResource
cloudfront:ListTagsForResource
iam:AttachRolePolicy
iam:CreateRole
iam:CreateServiceLinkedRole
iam:GetRole
iam:PutRolePolicy
iam:PassRole
lambda:CreateFunction
lambda:EnableReplication
lambda:DeleteFunction
lambda:GetFunction
lambda:GetFunctionConfiguration
lambda:PublishVersion
lambda:UpdateFunctionCode
lambda:UpdateFunctionConfiguration
lambda:ListTags
```

```
lambda:TagResource
lambda:UntagResource
route53:ChangeResourceRecordSets
route53:ListHostedZonesByName
route53:ListResourceRecordSets
s3:CreateBucket
s3:GetAccelerateConfiguration
s3:GetObject
s3:ListBucket
s3:PutAccelerateConfiguration
s3:PutBucketPolicy
s3:PutObject
s3:PutBucketTagging
s3:GetBucketTagging
lambda:ListEventSourceMappings
lambda:CreateEventSourceMapping
iam:UpdateAssumeRolePolicy
iam:DeleteRolePolicy
sqs:CreateQueue           // SQS only needed if using ISR feature
sqs:DeleteQueue
sqs:GetQueueAttributes
sqs:SetQueueAttributes
amplify:GetApp
amplify:GetBranch
amplify:UpdateApp
amplify:UpdateBranch
```

Résolution des problèmes liés aux déploiements de Next.js 11 SSR

Si vous rencontrez des problèmes inattendus lors du déploiement d'une application SSR classique (Next.js 11 uniquement) avec Amplify, consultez les rubriques de résolution des problèmes suivantes.

Rubriques

- Ma candidature a atteint une CloudFront limite
- Les fonctions Lambda @Edge sont créées dans la région USA Est (Virginie du Nord)
- Mon application Next.js utilise des fonctionnalités non prises en charge
- Les images de mon application Next.js ne se chargent pas
- Régions non prises en charge

Le répertoire de sortie de mon application est remplacé

Le répertoire de sortie d'une application Next.js déployée avec Amplify doit être défini sur. `.next` Si le répertoire de sortie de votre application est remplacé, vérifiez le `next.config.js` fichier. Pour que le répertoire de sortie de build soit défini par défaut sur`.next`, supprimez la ligne suivante du fichier :

```
distDir: 'build'
```

Vérifiez que le répertoire de sortie est défini sur `.next` dans vos paramètres de compilation. Pour plus d'informations sur l'affichage des paramètres de compilation de votre application, consultezConfiguration des paramètres de compilation pour une application Amplify.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur`.next`.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

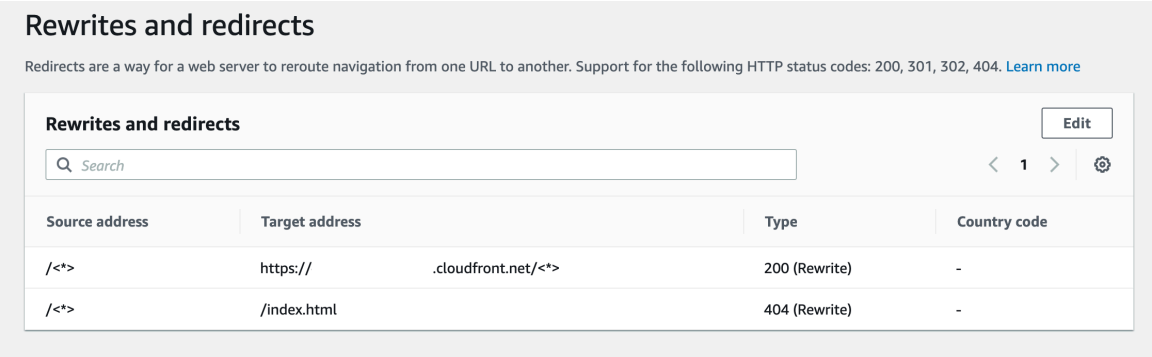
Je reçois une erreur 404 après le déploiement de mon site SSR

Si vous recevez une erreur 404 après le déploiement de votre site, le problème peut être dû au remplacement de votre répertoire de sortie. Pour vérifier votre `next.config.js` fichier et vérifier le répertoire de sortie de compilation correct dans les spécifications de compilation de votre application, suivez les étapes décrites dans la rubrique précédente,Le répertoire de sortie de mon application est remplacé.

Il manque la règle de réécriture pour les distributions CloudFront SSR dans mon application

Lorsque vous déployez une application SSR, Amplify crée une règle de réécriture pour CloudFront vos distributions SSR. Si vous ne pouvez pas accéder à votre application dans un navigateur Web, vérifiez que la règle de CloudFront réécriture existe pour votre application dans la console Amplify. S'il est absent, vous pouvez l'ajouter manuellement ou redéployer votre application.

Pour afficher ou modifier les règles de réécriture et de redirection d'une application dans la console Amplify, dans le volet de navigation, choisissez Paramètres de l'application, puis Réécritures et redirections. La capture d'écran suivante montre un exemple des règles de réécriture qu'Amplify crée pour vous lorsque vous déployez une application SSR. Notez que dans cet exemple, une règle CloudFront de réécriture existe.



Mon application est trop volumineuse pour être déployée

Amplify limite la taille d'un déploiement SSR à 50 Mo. Si vous essayez de déployer une application SSR Next.js sur Amplify et que vous obtenez `RequestEntityTooLargeException` une erreur, cela signifie que votre application est trop volumineuse pour être déployée. Vous pouvez essayer de contourner ce problème en ajoutant du code de nettoyage du cache à votre `next.config.js` fichier.

Voici un exemple de code contenu dans le `next.config.js` fichier qui effectue le nettoyage du cache.

```
module.exports = {
    webpack: (config, { buildId, dev, isServer, defaultLoaders, webpack }) => {
        config.optimization.splitChunks.cacheGroups = { }
        config.optimization.minimize = true;
        return config
      },
}
```

Ma compilation échoue en raison d'une erreur de mémoire insuffisante

Next.js vous permet de mettre en cache des artefacts de build afin d'améliorer les performances lors des builds suivants. En outre, le AWS CodeBuild conteneur d'Amplify compresse et télécharge ce cache sur Amazon S3, en votre nom, afin d'améliorer les performances de compilation ultérieures. Cela pourrait entraîner l'échec de votre compilation en raison d'une erreur de mémoire insuffisante.

Effectuez les actions suivantes pour empêcher votre application de dépasser la limite de mémoire pendant la phase de création. Tout d'abord, `.next/cache/**/*` supprimez-le de la section cache.paths de vos paramètres de compilation. Supprimez ensuite la variable d'`NODE_OPTIONS`environnement de votre fichier de paramètres de compilation. Définissez plutôt la variable d'`NODE_OPTIONS`environnement dans la console Amplify pour définir la limite de mémoire maximale du nœud. Pour plus d'informations sur la définition des variables d'environnement à l'aide de la console Amplify, consultez. Définition de variables d'environnement

Après avoir apporté ces modifications, réessayez de créer. En cas de succès, ajoutez-le à `.next/cache/**/*` nouveau à la section cache.paths de votre fichier de paramètres de compilation.

Pour plus d'informations sur la configuration du cache Next.js afin d'améliorer les performances de compilation, consultez AWS CodeBuild sur le site Web Next.js.

Mon application possède à la fois des branches SSR et SSG

Vous ne pouvez pas déployer une application qui possède à la fois des branches SSR et SSG. Si vous devez déployer à la fois des branches SSR et SSG, vous devez déployer une application qui utilise uniquement des branches SSR et une autre qui utilise uniquement des branches SSG.

Mon application stocke les fichiers statiques dans un dossier avec un chemin réservé

Next.js peut servir des fichiers statiques à partir d'un dossier nommé `public` qui est stocké dans le répertoire racine du projet. Lorsque vous déployez et hébergez une application Next.js avec Amplify, votre projet ne peut pas inclure de dossiers avec le chemin. `public/static` Amplify réserve le

`public/static` chemin à utiliser lors de la distribution de l'application. Si votre application inclut ce chemin, vous devez renommer le `static` dossier avant de le déployer avec Amplify.

Ma candidature a atteint une CloudFront limite

CloudFront les quotas de service limitent votre AWS compte à 25 distributions associées à des fonctions Lambda @Edge. Si vous dépassez ce quota, vous pouvez soit supprimer les CloudFront distributions non utilisées de votre compte, soit demander une augmentation du quota. Pour plus d'informations, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Les fonctions Lambda @Edge sont créées dans la région USA Est (Virginie du Nord)

Lorsque vous déployez une application Next.js, Amplify crée des fonctions Lambda @Edge pour personnaliser le contenu diffusé. CloudFront Les fonctions Lambda @Edge sont créées dans la région USA Est (Virginie du Nord), et non dans la région où votre application est déployée. Il s'agit d'une restriction Lambda @Edge. Pour plus d'informations sur les fonctions Lambda @Edge, consultez la section Restrictions relatives aux fonctions Edge dans le manuel Amazon CloudFront Developer Guide.

Mon application Next.js utilise des fonctionnalités non prises en charge

Les applications déployées avec Amplify prennent en charge les versions principales de Next.js jusqu'à la version 11. Pour une liste détaillée des fonctionnalités de Next.js prises en charge et non prises en charge par Amplify, voir. supported features

Lorsque vous déployez une nouvelle application Next.js, Amplify utilise par défaut la dernière version prise en charge de Next.js. Si vous avez une application Next.js existante que vous avez déployée sur Amplify avec une ancienne version de Next.js, vous pouvez migrer l'application vers le fournisseur de calcul SSR d'Amplify Hosting. Pour obtenir des instructions, veuillez consulter Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute.

Les images de mon application Next.js ne se chargent pas

Lorsque vous ajoutez des images à votre application Next.js à l'aide du `next/image` composant, la taille de l'image ne peut pas dépasser 1 Mo. Lorsque vous déployez l'application sur Amplify, les images de plus de 1 Mo renvoient une erreur 503. Cela est dû à une limite Lambda @Edge qui limite la taille d'une réponse générée par une fonction Lambda, y compris les en-têtes et le corps, à 1 Mo.

La limite de 1 Mo s'applique aux autres artefacts de votre application, tels que les fichiers PDF et les documents.

Régions non prises en charge

Amplify ne prend pas en charge le déploiement de l'application SSR classique (Next.js 11 uniquement) dans toutes les régions AWS où Amplify est disponible. Le SSR classique (Next.js 11 uniquement) n'est pas pris en charge dans les régions suivantes : Europe (Milan) eu-south-1, Moyen-Orient (Bahreïn) me-south-1 et Asie-Pacifique (Hong Kong) ap-east-1.

Résolution des problèmes liés aux déploiements SSR

Si vous rencontrez des problèmes inattendus lors du déploiement d'une application SSR avec le calcul d'Amplify Hosting, Résolution des problèmes liés aux applications rendues côté serveur consultez le chapitre sur le dépannage d'Amplify.

Avancé : adaptateurs open source

Les auteurs de frameworks peuvent utiliser la spécification de déploiement basée sur le système de fichiers pour développer des adaptateurs de build open source personnalisés pour leurs frameworks spécifiques. Ces adaptateurs transformeront la sortie de compilation d'une application en un bundle de déploiement conforme à la structure de répertoire attendue d'Amplify Hosting. Ce bundle de déploiement inclura tous les fichiers et actifs nécessaires pour héberger une application, y compris la configuration d'exécution, telle que les règles de routage.

Si vous n'utilisez pas de framework, vous pouvez développer votre propre solution pour générer une sortie de build attendue par Amplify.

Rubriques

- Utilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de compilation
- Déploiement d'un serveur Express à l'aide du manifeste de déploiement
- Intégration de l'optimisation des images pour les auteurs de frameworks
- Utilisation d'adaptateurs open source pour n'importe quel framework SSR

Utilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de compilation

La spécification de déploiement d'Amplify Hosting est une spécification basée sur un système de fichiers qui définit la structure de répertoire qui facilite les déploiements vers Amplify Hosting.

Un framework peut générer cette structure de répertoire attendue en sortie de sa commande de construction, ce qui lui permet de tirer parti des primitives de service d'Amplify Hosting. Amplify Hosting comprend la structure du bundle de déploiement et le déploie en conséquence.

Pour une démonstration vidéo expliquant comment utiliser les spécifications de déploiement, consultez Comment héberger un site Web via AWS Amplify le YouTube canal Amazon Web Services.

Voici un exemple de la structure de dossiers qu'Amplify attend pour le bundle de déploiement. À un niveau élevé, il possède un dossier nommé`static`, un dossier nommé `compute` et un fichier manifeste de déploiement nommé`deploy-manifest.json`.

```
.amplify-hosting/
### compute/
#   ### default/
#       ### chunks/
#       #   ### app/
#       #       ### _nuxt/
#       #       #   ### index-xxx.mjs
#       #       #   ### index-styles.xxx.js
#       #       ### server.mjs
#       ### node_modules/
#       ### server.js
```

```
### static/
#   ### css/
#   #   ### nuxt-google-fonts.css
#   ### fonts/
#   #   ### font.woff2
#   ### _nuxt/
#   #   ### builds/
#   #   #   ### latest.json
#   #   ### entry.xxx.js
#   ### favicon.ico
#   ### robots.txt
### deploy-manifest.json
```

Amplify SSR : support primitif

La spécification de déploiement d'Amplify Hosting définit un contrat qui correspond étroitement aux primitives suivantes.

Actifs statiques

Fournit des frameworks capables d'héberger des fichiers statiques.

Calcul

Fournit des frameworks capables d'exécuter un serveur HTTP Node.js sur le port 3000.

Optimisation de l'image

Fournit aux frameworks un service permettant d'optimiser les images lors de l'exécution.

règles de routage

Fournit des frameworks dotés d'un mécanisme permettant de mapper les chemins des demandes entrantes vers des cibles spécifiques.

L'.amplify-hosting/staticannuaire

Vous devez placer dans le `.amplify-hosting/static` répertoire tous les fichiers statiques accessibles au public destinés à être servis à partir de l'URL de l'application. Les fichiers contenus dans ce répertoire sont servis via la primitive static assets.

Les fichiers statiques sont accessibles à la racine (/) de l'URL de l'application sans aucune modification de leur contenu, de leur nom de fichier ou de leur extension. En outre, les sous-

répertoires sont conservés dans la structure de l'URL et apparaissent avant le nom du fichier. À titre d'exemple, `.amplify-hosting/static/favicon.ico` sera servi depuis `https://myAppId.amplify-hostingapp.com/favicon.ico` et `.amplify-hosting/static/_nuxt/main.js` sera servi depuis `https://myAppId.amplify-hostingapp.com/_nuxt/main.js`

Si un framework permet de modifier le chemin de base de l'application, il doit ajouter le chemin de base aux actifs statiques du `.amplify-hosting/static` répertoire. Par exemple, si le chemin de base est`/folder1/folder2`, la sortie de génération pour un actif statique appelé `main.css` sera`.amplify-hosting/static/folder1/folder2/main.css`.

L'.amplify-hosting/computeannuaire

Une ressource de calcul unique est représentée par un sous-répertoire unique nommé `default` contenu dans le `.amplify-hosting/compute` répertoire. Le chemin est`.amplify-hosting/compute/default`. Cette ressource de calcul correspond à la primitive de calcul d'Amplify Hosting.

Le contenu du `default` sous-répertoire doit être conforme aux règles suivantes.

- Un fichier doit exister à la racine du `default` sous-répertoire pour servir de point d'entrée à la ressource de calcul.
- Le fichier du point d'entrée doit être un module Node.js et il doit démarrer un serveur HTTP qui écoute sur le port 3000.
- Vous pouvez placer d'autres fichiers dans le `default` sous-répertoire et les référencer à partir du code du fichier du point d'entrée.
- Le contenu du sous-répertoire doit être autonome. Le code du module du point d'entrée ne peut faire référence à aucun module en dehors du sous-répertoire. Notez que les frameworks peuvent regrouper leur serveur HTTP comme ils le souhaitent. Si le processus de calcul peut être lancé avec la `node server.js` commande, où `server.js is` est le nom du fichier d'entrée, depuis le sous-répertoire, Amplify considère que la structure du répertoire est conforme à la spécification de déploiement.

Amplify Hosting regroupe et déploie tous les fichiers du `default` sous-répertoire vers une ressource de calcul provisionnée. Chaque ressource de calcul se voit attribuer 512 Mo de stockage éphémère. Ce stockage n'est pas partagé entre les instances d'exécution, mais est partagé entre les invocations suivantes au sein de la même instance d'exécution. Les instances d'exécution sont limitées à une durée d'exécution maximale de 15 minutes, et le seul chemin accessible en écriture au sein de l'instance d'exécution est le `/tmp` répertoire. La taille non compressée de chaque ensemble de

ressources de calcul ne peut pas dépasser 220 Mo. Par exemple, le `.amplify/compute/default` sous-répertoire ne peut pas dépasser 220 Mo lorsqu'il est décompressé.

le fichier .amplify-hosting/deploy-manifest.json ;

Utilisez le `deploy-manifest.json` fichier pour stocker les détails de configuration et les métadonnées d'un déploiement. Au minimum, un `deploy-manifest.json` fichier doit inclure un `version` attribut, l'`routes`attribut avec une route fourre-tout spécifiée et l'`framework`attribut avec des métadonnées de structure spécifiées.

La définition d'objet suivante illustre la configuration d'un manifeste de déploiement.

```
type DeployManifest = {
  version: 1;
  routes: Route[];
  computeResources?: ComputeResource[];
  imageSettings?: ImageSettings;
  framework: FrameworkMetadata;
};
```

Les rubriques suivantes décrivent les détails et l'utilisation de chaque attribut du manifeste de déploiement.

Utilisation de l'attribut version

L'`version`attribut définit la version de la spécification de déploiement que vous implémentez. Actuellement, la seule version pour la spécification de déploiement d'Amplify Hosting est la version 1. L'exemple JSON suivant illustre l'utilisation de l'`version`attribut.

```
"version": 1
```

Utilisation de l'attribut routes

L'`routes`attribut permet aux frameworks de tirer parti de la primitive des règles de routage d'Amplify Hosting. Les règles de routage fournissent un mécanisme permettant d'acheminer les chemins des demandes entrantes vers une cible spécifique du bundle de déploiement. Les règles de routage dictent uniquement la destination d'une demande entrante et sont appliquées une fois que la demande a été transformée par des règles de réécriture et de redirection. Pour plus d'informations sur la façon dont Amplify Hosting gère les réécritures et les redirections, consultez. Configuration des redirections et des réécritures pour une application Amplify

Les règles de routage ne réécrivent ni ne transforment la demande. Si une demande entrante correspond au modèle de chemin d'un itinéraire, la demande est acheminée telle quelle vers la cible de l'itinéraire.

Les règles de routage spécifiées dans le `routes` tableau doivent être conformes aux règles suivantes.

- Un itinéraire fourre-tout doit être spécifié. Un itinéraire fourre-tout possède le `/*` modèle qui correspond à toutes les demandes entrantes.
- Le `routes` tableau peut contenir un maximum de 25 éléments.
- Vous devez spécifier un `Static` itinéraire ou un `Compute` itinéraire.
- Si vous spécifiez un `Static` itinéraire, le `.amplify-hosting/static` répertoire doit exister.
- Si vous spécifiez un `Compute` itinéraire, le `.amplify-hosting/compute` répertoire doit exister.
- Si vous spécifiez un `ImageOptimization` itinéraire, vous devez également spécifier un `Compute` itinéraire. Cela est nécessaire car l'optimisation des images n'est pas encore prise en charge pour les applications purement statiques.

La définition d'objet suivante illustre la configuration de l'`Route`objet.

```
type Route = {
  path: string;
  target: Target;
  fallback?: Target;
}
```

Le tableau suivant décrit les propriétés de l'`Route`objet.

Clé	Type	Obligatoire	Description
path	String	Oui	Définit un modèle qui correspond aux chemins des demandes entrantes (à l'exception de la chaîne de requête).

Clé	Type	Obligatoire	Description
			La longueur maximale du chemin est de 255 caractères. Un tracé doit commencer par la barre oblique/. Un chemin peut contenir n'importe lequel des caractères suivants : [A-Z], [a-z], [0-9], [_-.*$/~"'@ : +]. Pour la correspon dance de modèles, seuls les caractères génériques suivants sont pris en charge : • *(correspond à 0 caractères ou plus) • Le /* modèle est appelé modèle fourre-tout et correspond à toutes les demandes entrantes.

Clé	Type	Obligatoire	Description
cible	Cible	Oui	Un objet qui définit la cible vers laquelle acheminer la demande correspondante. Si un `Compute` itinéraire est spécifié, un itinéraire correspondant `ComputeResource` doit exister. Si un `ImageOptimization` itinéraire est spécifié, `imageSettings` il doit également être spécifié.

Clé	Type	Obligatoire	Description
repli	Cible	Non	Objet qui définit la cible vers laquelle se rabattre si la cible d'origine renvoie une erreur 404. Le `target` type et le `fallback` type ne peuvent pas être identiques pour un itinéraire spécifiqu e. Par exemple, le repli de `Static` à n'`Static`est pas autorisé. Les solutions de secours ne sont prises en charge que pour les requêtes GET qui n'ont pas de corps. Si un corps est présent dans la demande, il sera supprimé lors du repli.

La définition d'objet suivante illustre la configuration de l'`Target`objet.

```
type Target = {
  kind: TargetKind;
  src?: string;
  cacheControl?: string;
}
```

Le tableau suivant décrit les propriétés de l'`Target`objet.

Clé	Type	Obligatoire	Description
sorte	Type cible	Oui	Et `enum` qui définit le type de cible. Les valeurs valides sont `Static`, `Compute` et `ImageOpti mization` .
src	String	Oui pour `Compute` Non pour les autres primitives	Chaîne qui indique le nom du sous-répertoire du bundle de déploiement qui contient le code exécutable de la primitive. Valide et obligatoire uniquemen t pour la primitive Compute. La valeur doit pointer vers l'une des ressources de calcul présentes dans le bundle de déploieme nt. Actuellement, la seule valeur prise en charge pour ce champ est`default`.
Contrôle du cache	String	Non	Chaîne qui indique la valeur de l'en-tête Cache-Con trol à appliquer à la réponse. Valable uniquement pour le Static et les

Clé	Type	Obligatoire	Description
			ImageOptimization primitives. La valeur spécifiée est remplacée par des en-têtes personnalisés. Pour plus d'informations sur les en-têtes clients d'Amplify Hosting, consultez. Configuration d'en-têtes personnalisés pour une application Amplify **Note** Cet en-tête Cache-Control est uniquement appliqué aux réponses réussies dont le code d'état est défini sur 200 (OK).

La définition d'objet suivante illustre l'utilisation de l'`TargetKind`énumération.

```
enum TargetKind {
  Static = "Static",
  Compute = "Compute",
  ImageOptimization = "ImageOptimization"
```

```
}
```

La liste suivante indique les valeurs valides pour l'`TargetKind`énumération.

Statique

Achemine les demandes vers la primitive des actifs statiques.

Calcul

Achemine les demandes vers la primitive de calcul.

ImageOptimization

Achemine les demandes vers la primitive d'optimisation d'image.

L'exemple JSON suivant illustre l'utilisation de l'`routes`attribut avec plusieurs règles de routage spécifiées.

```
"routes": [
    {
      "path": "/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/_nuxt/builds/meta/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/builds/*",
      "target": {
        "cacheControl": "public, max-age=1, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/*",
      "target": {
```

```
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/*.*",
      "target": {
        "kind": "Static"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ]
```

Pour plus d'informations sur la spécification des règles de routage dans votre manifeste de déploiement, voir Bonnes pratiques pour configurer les règles de routage

Utilisation de l'attribut ComputerResources

L'`computeResources`attribut permet aux frameworks de fournir des métadonnées sur les ressources de calcul allouées. Chaque ressource de calcul doit être associée à un itinéraire correspondant.

La définition d'objet suivante illustre l'utilisation de l'`ComputeResource`objet.

```
type ComputeResource = {
  name: string;
  runtime: ComputeRuntime;
  entrypoint: string;
};

type ComputeRuntime = 'nodejs20.x' | 'nodejs22.x';
```

Le tableau suivant décrit les propriétés de l'`ComputeResource`objet.

Clé	Type	Obligatoire	Description
name	String	Oui	Spécifie le nom de la ressource de calcul. Le nom doit correspondre au nom du sous-répertoire situé dans le `.amplify-hosting/compute directory` . Pour la version 1 de la spécification de déploiement, la seule valeur valide est `default`.
environnement d'exécution	ComputeRuntime	Oui	Définit le temps d'exécution de la ressource de calcul provisionnée. Les valeurs valides sont `nodejs20.x` et `nodejs22.x` .
point d'entrée	String	Oui	Spécifie le nom du fichier de départ à partir duquel le code sera exécuté pour la ressource de calcul spécifiée. Le fichier doit exister dans le sous-répertoire qui représente une ressource de calcul.

Si vous avez une structure de répertoire qui ressemble à la suivante.

```
.amplify-hosting
|---compute
|   |---default
|       |---index.js
```

Le JSON de l'`computeResource`attribut ressemblera à ce qui suit.

```
"computeResources": [
    {
      "name": "default",
      "runtime": "nodejs20.x",
      "entrypoint": "index.js",
    }
  ]
```

Utilisation de l'attribut ImageSettings

L'`imageSettings`attribut permet aux frameworks de personnaliser le comportement de la primitive d'optimisation d'image, qui fournit une optimisation à la demande des images lors de l'exécution.

La définition d'objet suivante illustre l'utilisation de l'`ImageSettings`objet.

```
type ImageSettings = {
  sizes: number[];
  domains: string[];
  remotePatterns: RemotePattern[];
  formats: ImageFormat[];
  minumumCacheTTL: number;
  dangerouslyAllowSVG: boolean;
};

type ImageFormat = 'image/avif' | 'image/webp' | 'image/png' | 'image/jpeg';
```

Le tableau suivant décrit les propriétés de l'`ImageSettings`objet.

Clé	Type	Obligatoire	Description
tailles	Numéro []	Oui	Un tableau de largeurs d'image prises en charge.
domains	Chaîne []	Oui	Un ensemble de domaines externes autorisés qui peuvent utiliser l'optimisation d'image. Laissez le tableau vide pour autoriser uniquemen t le domaine de déploiement à utiliser l'optimisation des images.
Motifs distants	RemotePattern[]	Oui	Un tableau de modèles externes autorisés qui peuvent utiliser l'optimisation de l'image. Similaire aux domaines, mais offre un meilleur contrôle avec les expressions régulière s (regex).
formats	ImageFormat[]	Oui	Un tableau de formats d'image de sortie autorisés.
Cache minimal TL	Number	Oui	Durée du cache en secondes pour les images optimisées.

Clé	Type	Obligatoire	Description
Autorise dangereus ement le SVG	Booléen	Oui	Autorise l'entrée d'image URLs au format SVG. Cette option est désactivé e par défaut pour des raisons de sécurité.

La définition d'objet suivante illustre l'utilisation de l'`RemotePattern`objet.

```
type RemotePattern = {
  protocol?: 'https';
  hostname: string;
  port?: string;
  pathname?: string;
}
```

Le tableau suivant décrit les propriétés de l'`RemotePattern`objet.

Clé	Type	Obligatoire	Description
protocole ;	String	Non	Protocole du modèle de télécommande autorisé. La seule valeur valide est `https`.
hostname	String	Oui	Le nom d'hôte du modèle distant autorisé. Vous pouvez spécifier un littéral ou un caractère générique. Un seul `*` correspon d à un seul sous-domaine. Un double

Clé	Type	Obligatoire	Description
			`**` correspond à un nombre quelconque de sous-domaines. Amplify n'autorise pas les caractères génériques où seul `**` est spécifié.
port	String	Non	Port du modèle de télécommande autorisé.
chemin d'accès	String	Non	Le nom du chemin du modèle de télécommande autorisé.

L'exemple suivant illustre l'`imageSettings`attribut.

```
"imageSettings": {
    "sizes": [
      100,
      200
    ],
    "domains": [
      "example.com"
    ],
    "remotePatterns": [
      {
        "protocol": "https",
        "hostname": "example.com",
        "port": "",
        "pathname": "/**",
      }
    ],
    "formats": [
      "image/webp"
    ],
    "minumumCacheTTL": 60,
```

```
    "dangerouslyAllowSVG": false
  }
```

Utilisation de l'attribut framework

Utilisez l'`framework`attribut pour spécifier les métadonnées du framework.

La définition d'objet suivante illustre la configuration de l'`FrameworkMetadata`objet.

```
type FrameworkMetadata = {
  name: string;
  version: string;
}
```

Le tableau suivant décrit les propriétés de l'`FrameworkMetadata`objet.

Clé	Type	Obligatoire	Description
name	String	Oui	Le nom du framework.
version	String	Oui	Version du framework . Il doit s'agir d'une chaîne de version sémantique (semver) valide.

Bonnes pratiques pour configurer les règles de routage

Les règles de routage fournissent un mécanisme pour acheminer les chemins des demandes entrantes vers des cibles spécifiques du bundle de déploiement. Dans un bundle de déploiement, les auteurs du framework peuvent émettre des fichiers vers la sortie de build qui sont déployés sur l'une des cibles suivantes :

- Ressources statiques primitives — Les fichiers sont contenus dans le `.amplify-hosting/static` répertoire.
- Primitive de calcul — Les fichiers sont contenus dans le `.amplify-hosting/compute/default` répertoire.

Les auteurs du framework fournissent également un ensemble de règles de routage dans le fichier manifeste de déploiement. Chaque règle du tableau est comparée à la demande entrante dans un ordre séquentiel, jusqu'à ce qu'il y ait une correspondance. Lorsqu'il existe une règle de correspondance, la demande est acheminée vers la cible spécifiée dans la règle de correspondance. Facultativement, une cible de secours peut être spécifiée pour chaque règle. Si la cible d'origine renvoie une erreur 404, la demande est acheminée vers la cible de secours.

La spécification de déploiement exige que la dernière règle de l'ordre de traversée soit une règle fourre-tout. Une règle fourre-tout est spécifiée avec le `/*` chemin. Si la demande entrante ne correspond à aucune des routes précédentes du tableau de règles de routage, elle est acheminée vers la cible de règles fourre-tout.

Pour les frameworks SSR tels queNuxt.js, la cible de la règle fourre-tout doit être la primitive de calcul. Cela est dû au fait que les applications SSR ont des pages affichées côté serveur avec des itinéraires qui ne sont pas prévisibles au moment de la création. Par exemple, si une Nuxt.js application possède une page `/blog/[slug]` où se `[slug]` trouve un paramètre de route dynamique. La cible de la règle fourre-tout est le seul moyen d'acheminer les demandes vers ces pages.

En revanche, des modèles de trajectoire spécifiques peuvent être utilisés pour cibler des itinéraires connus au moment de la construction. Par exemple, Nuxt.js diffuse les actifs statiques depuis le `/_nuxt` chemin. Cela signifie que le `/_nuxt/*` chemin peut être ciblé par une règle de routage spécifique qui achemine les demandes vers la primitive des actifs statiques.

Routage des dossiers publics

La plupart des frameworks SSR offrent la possibilité de servir des actifs statiques mutables à partir d'un `public` dossier. Les fichiers tels que `favicon.ico` et `robots.txt` sont généralement conservés dans le `public` dossier et sont servis à partir de l'URL racine de l'application. Par exemple, le `favicon.ico` fichier est servi à partir de`https://example.com/favicon.ico`. Notez qu'il n'existe aucun modèle de chemin prévisible pour ces fichiers. Ils sont presque entièrement dictés par le nom du fichier. La seule façon de cibler les fichiers contenus dans le `public` dossier est d'utiliser la méthode fourre-tout. Cependant, la cible de l'itinéraire fourre-tout doit être la primitive de calcul.

Nous recommandons l'une des approches suivantes pour gérer votre `public` dossier.

1. Utilisez un modèle de chemin pour cibler les chemins de requête contenant des extensions de fichiers. Par exemple, vous pouvez l'utiliser `/*.*` pour cibler tous les chemins de demande contenant une extension de fichier.

Notez que cette approche peut ne pas être fiable. Par exemple, si le `public` dossier contient des fichiers sans extension, ils ne sont pas visés par cette règle. Un autre problème à prendre en compte avec cette approche est que l'application peut avoir des pages avec des points dans leur nom. Par exemple, une page `/blog/2021/01/01/hello.world` sera ciblée par la `/*.*` règle. Ce n'est pas idéal car la page n'est pas un actif statique. Toutefois, vous pouvez ajouter une cible de secours à cette règle pour garantir qu'en cas d'erreur 404 provenant de la primitive statique, la requête revienne à la primitive de calcul.

```
{
    "path": "/*.*",
    "target": {
        "kind": "Static"
    },
    "fallback": {
        "kind": "Compute",
        "src": "default"
    }
}
```

2. Identifiez les fichiers du `public` dossier au moment de la création et émettez une règle de routage pour chaque fichier. Cette approche n'est pas évolutive car la spécification de déploiement impose une limite de 25 règles.

```
{
    "path": "/favicon.ico",
    "target": {
        "kind": "Static"
    }
},
{
    "path": "/robots.txt",
    "target": {
        "kind": "Static"
    }
}
```

3. Recommandez aux utilisateurs de votre framework de stocker tous les actifs statiques mutables dans un sous-dossier du `public` dossier.

Dans l'exemple suivant, l'utilisateur peut stocker tous les actifs statiques modifiables dans le `public/assets` dossier. Ensuite, une règle de routage avec le modèle de chemin `/assets/*` peut être utilisée pour cibler tous les actifs statiques mutables présents dans le `public/assets` dossier.

```
{
    "path": "/assets/*",
    "target": {
        "kind": "Static"
    }
}
```

4. Spécifiez une solution de secours statique pour l'itinéraire fourre-tout. Cette approche présente des inconvénients qui sont décrits plus en détail dans la Routage de secours fourre-tout section suivante.

Routage de secours fourre-tout

Pour les frameworks SSR tels que ceux Nuxt.js où une route fourre-tout est spécifiée pour la cible primitive de calcul, les auteurs du framework peuvent envisager de spécifier une solution de secours statique pour la route fourre-tout afin de résoudre le problème de routage des dossiers. `public` Cependant, ce type de règle de routage interrompt les pages 404 affichées côté serveur. Par exemple, si l'utilisateur final visite une page qui n'existe pas, l'application affiche une page 404 avec un code d'état 404. Toutefois, si la route fourre-tout comporte une solution de secours statique, la page 404 n'est pas affichée. Au lieu de cela, la requête revient à la primitive statique et aboutit toujours à un code d'état 404, mais la page 404 n'est pas rendue.

```
{
    "path": "/*",
    "target": {
        "kind": "Compute",
        "src": "default"
    },
    "fallback": {
        "kind": "Static"
    }
}
```

Routage du chemin de base

Les frameworks qui offrent la possibilité de modifier le chemin de base de l'application sont censés ajouter le chemin de base aux actifs statiques du `.amplify-hosting/static` répertoire. Par exemple, si le chemin de base est`/folder1/folder2`, la sortie de génération pour un actif statique appelé main.css sera`.amplify-hosting/static/folder1/folder2/main.css`.

Cela signifie que les règles de routage doivent également être mises à jour pour refléter le chemin de base. Par exemple, si le chemin de base est`/folder1/folder2`, la règle de routage pour les actifs statiques du `public` dossier sera la suivante.

```
{
    "path": "/folder1/folder2/*.*",
    "target": {
        "kind": "Static"
    }
}
```

De même, les routes côté serveur doivent également être précédées du chemin de base. Par exemple, si le chemin de base est`/folder1/folder2`, la règle de routage de l'`/api`itinéraire ressemblera à ce qui suit.

```
{
    "path": "/folder1/folder2/api/*",
    "target": {
        "kind": "Compute",
        "src": "default"
    }
}
```

Cependant, le chemin de base ne doit pas être ajouté à l'itinéraire fourre-tout. Par exemple, si le chemin de base est le suivant`/folder1/folder2`, l'itinéraire fourre-tout restera le suivant.

```
{
    "path": "/*",
    "target": {
        "kind": "Compute",
        "src": "default"
    }
}
```

Exemples de routes Nuxt.js

Voici un exemple de `deploy-manifest.json` fichier pour une application Nuxt qui montre comment spécifier des règles de routage.

```
{
  "version": 1,
  "routes": [
    {
      "path": "/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/_nuxt/builds/meta/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/builds/*",
      "target": {
        "cacheControl": "public, max-age=1, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/*.*",
      "target": {
        "kind": "Static"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
```

```
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ],
  "computeResources": [
    {
      "name": "default",
      "entrypoint": "server.js",
      "runtime": "nodejs22.x"
    }
  ],
  "framework": {
    "name": "nuxt",
    "version": "3.8.1"
  }
}
```

Voici un exemple de `deploy-manifest.json` fichier pour Nuxt qui montre comment spécifier des règles de routage, y compris des chemins de base.

```
{
  "version": 1,
  "routes": [
    {
      "path": "/base-path/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/base-path/_nuxt/builds/meta/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
```

```
    {
      "path": "/base-path/_nuxt/builds/*",
      "target": {
        "cacheControl": "public, max-age=1, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/base-path/_nuxt/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/base-path/*.*",
      "target": {
        "kind": "Static"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ],
  "computeResources": [
    {
      "name": "default",
      "entrypoint": "server.js",
      "runtime": "nodejs22.x"
    }
  ],
  "framework": {
    "name": "nuxt",
    "version": "3.8.1"
  }
```

```
}
```

Pour plus d'informations sur l'utilisation de l'`routes`attribut, consultez[Utilisation de l'attribut routes](#).

Déploiement d'un serveur Express à l'aide du manifeste de déploiement

Cet exemple explique comment déployer un serveur Express de base à l'aide de la spécification de déploiement d'Amplify Hosting. Vous pouvez utiliser le manifeste de déploiement fourni pour spécifier le routage, les ressources de calcul et d'autres configurations.

Configurer un serveur Express localement avant de le déployer sur Amplify Hosting

1. Créez un nouveau répertoire pour votre projet et installez Express et Typescript.

```
mkdir express-app
cd express-app

# The following command will prompt you for information about your project
npm init

# Install express, typescript and types
npm install express --save
npm install typescript ts-node @types/node @types/express --save-dev
```

2. Ajoutez un `tsconfig.json` fichier à la racine de votre projet avec le contenu suivant.

```
{
  "compilerOptions": {
    "target": "es6",
    "module": "commonjs",
    "outDir": "./dist",
    "strict": true,
    "esModuleInterop": true,
    "skipLibCheck": true,
    "forceConsistentCasingInFileNames": true
  },
  "include": ["src/**/*.ts"],
  "exclude": ["node_modules"]
}
```

3. Créez un répertoire nommé `src` à la racine de votre projet.

4. Créez un `index.ts` fichier dans le `src` répertoire. Ce sera le point d'entrée de l'application qui démarre un serveur Express. Le serveur doit être configuré pour écouter sur le port 3000.

```
// src/index.ts
import express from 'express';

const app: express.Application = express();
const port = 3000;

app.use(express.text());

app.listen(port, () => {
  console.log(`server is listening on ${port}`);
});

// Homepage
app.get('/', (req: express.Request, res: express.Response) => {
  res.status(200).send("Hello World!");
});

// GET
app.get('/get', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-get-header", "get-header-value").send("get-response-
from-compute");
});

//POST
app.post('/post', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-post-header", "post-header-
value").send(req.body.toString());
});

//PUT
app.put('/put', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-put-header", "put-header-
value").send(req.body.toString());
});

//PATCH
app.patch('/patch', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-patch-header", "patch-header-
value").send(req.body.toString());
});
```

```
// Delete
app.delete('/delete', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-delete-header", "delete-header-value").send();
});
```

5. Ajoutez les scripts suivants à votre `package.json` fichier.

```
"scripts": {
  "start": "ts-node src/index.ts",
  "build": "tsc",
  "serve": "node dist/index.js"
}
```

6. Créez un répertoire nommé `public` à la racine de votre projet. Créez ensuite un fichier nommé `hello-world.txt` avec le contenu suivant.

```
Hello world!
```

7. Ajoutez un `.gitignore` fichier à la racine de votre projet avec le contenu suivant.

```
.amplify-hosting
dist
node_modules
```

Configurer le manifeste de déploiement d'Amplify

1. Créez un fichier nommé `deploy-manifest.json` dans le répertoire racine de votre projet.
2. Copiez et collez le manifeste suivant dans votre `deploy-manifest.json` fichier.

```
{
  "version": 1,
  "framework": { "name": "express", "version": "4.18.2" },
  "imageSettings": {
    "sizes": [
      100,
      200,
      1920
    ],
    "domains": [],
    "remotePatterns": [],
```

```
    "formats": [],
    "minimumCacheTTL": 60,
    "dangerouslyAllowSVG": false
  },
  "routes": [
    {
      "path": "/_amplify/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/*.*",
      "target": {
        "kind": "Static",
        "cacheControl": "public, max-age=2"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ],
  "computeResources": [
    {
      "name": "default",
      "runtime": "nodejs22.x",
      "entrypoint": "index.js"
    }
  ]
}
```

Le manifeste décrit comment Amplify Hosting doit gérer le déploiement de votre application. Les principaux paramètres sont les suivants.

- version — Indique la version de la spécification de déploiement que vous utilisez.
- framework — Ajustez-le pour spécifier la configuration de votre Express serveur.
- ImageSettings — Cette section est facultative pour un Express serveur, sauf si vous gérez l'optimisation des images.
- itinéraires : ils sont essentiels pour diriger le trafic vers les bonnes parties de votre application. L'`"kind": "Compute"`itinéraire dirige le trafic vers la logique de votre serveur.
- ComputerResources — Utilisez cette section pour spécifier le moteur d'exécution et le point d'entrée de votre Express serveur.

Configurez ensuite un script de post-génération qui déplace les artefacts de l'application créée dans le bundle de `.amplify-hosting` déploiement. La structure du répertoire est conforme à la spécification de déploiement d'Amplify Hosting.

Configurer le script de post-construction

1. Créez un répertoire nommé `bin` à la racine de votre projet.
2. Créez un fichier nommé `postbuild.sh` dans le `bin` répertoire. Ajoutez le contenu suivant au fichier `postbuild.sh`.

```
#!/bin/bash

rm -rf ./.amplify-hosting

mkdir -p ./.amplify-hosting/compute

cp -r ./dist ./.amplify-hosting/compute/default
cp -r ./node_modules ./.amplify-hosting/compute/default/node_modules

cp -r public ./.amplify-hosting/static

cp deploy-manifest.json ./.amplify-hosting/deploy-manifest.json
```

3. Ajoutez un `postbuild` script à votre `package.json` fichier. Le fichier doit ressembler à ce qui suit.

```
"scripts": {
  "start": "ts-node src/index.ts",
  "build": "tsc",
```

```
  "serve": "node dist/index.js",
  "postbuild": "chmod +x bin/postbuild.sh && ./bin/postbuild.sh"
}
```

4. Exécutez la commande suivante pour créer votre application.

```
npm run build
```

5. (Facultatif) Ajustez vos itinéraires pour Express. Vous pouvez modifier les itinéraires de votre manifeste de déploiement pour les adapter à votre serveur Express. Par exemple, si le `public` répertoire ne contient aucun actif statique, il se peut que vous n'ayez besoin que de la route fourre-tout menant à `"path": "/*"` Compute. Cela dépend de la configuration de votre serveur.

La structure finale de votre répertoire devrait ressembler à ce qui suit.

```
express-app/
### .amplify-hosting/
#   ### compute/
#   #   ### default/
#   #       ### node_modules/
#   #       ### index.js
#   ### static/
#   #   ### hello.txt
#   ### deploy-manifest.json
### bin/
#   ### .amplify-hosting/
#   #   ### compute/
#   #   #   ### default/
#   #   ### static/
#   ### postbuild.sh*
### dist/
#   ### index.js
### node_modules/
### public/
#   ### hello.txt
### src/
#   ### index.ts
### deploy-manifest.json
### package.json
### package-lock.json
```

```
### tsconfig.json
```

Déployez votre serveur

1. Transférez votre code dans votre dépôt Git, puis déployez votre application sur Amplify Hosting.
2. Mettez à jour vos paramètres de compilation pour qu'ils `baseDirectory` pointent vers ce qui `.amplify-hosting` suit. Au cours de la compilation, Amplify détectera le fichier manifeste dans le `.amplify-hosting` répertoire et déploiera votre serveur Express tel que configuré.

   ```
   version: 1
   frontend:
     phases:
       preBuild:
         commands:
           - nvm use 18
           - npm install
       build:
         commands:
           - npm run build
     artifacts:
       baseDirectory: .amplify-hosting
       files:
         - '**/*'
   ```

3. Pour vérifier que votre déploiement a réussi et que votre serveur fonctionne correctement, accédez à votre application à l'URL par défaut fournie par Amplify Hosting.

Intégration de l'optimisation des images pour les auteurs de frameworks

Les auteurs du framework peuvent intégrer la fonctionnalité d'optimisation d'image d'Amplify en utilisant la spécification de déploiement d'Amplify Hosting. Pour activer l'optimisation des images, votre manifeste de déploiement doit contenir une règle de routage qui cible le service d'optimisation des images. L'exemple suivant montre comment configurer la règle de routage.

```
// .amplify-hosting/deploy-manifest.json

{
  "routes": [
    {
      "path": "/images/*",
```

```
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=31536000, immutable"
      }
    }
  ]
}
```

Pour plus d'informations sur la configuration des paramètres d'optimisation des images à l'aide de la spécification de déploiement, consultezUtilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de compilation.

Comprendre l'API d'optimisation des images

L'optimisation des images peut être invoquée lors de l'exécution via l'URL de domaine d'une application Amplify, sur le chemin défini par la règle de routage.

```
GET https://{appDomainName}/{path}?{queryParams}
```

L'optimisation des images impose les règles suivantes aux images.

- Amplify ne peut pas optimiser les formats GIF, APNG et SVG ni les convertir dans un autre format.
- Les images SVG ne sont diffusées que si le `dangerouslyAllowSVG` paramètre est activé.
- La largeur ou la hauteur d'une image source ne peut pas dépasser 11 Mo ou 9 000 pixels.
- La limite de taille d'une image optimisée est de 4 Mo.
- Le protocole HTTPS est le seul protocole pris en charge pour le sourcing d'images à distance URLs.

En-têtes HTTP

L'en-tête HTTP de la demande Accept est utilisé pour spécifier les formats d'image, exprimés sous forme de types MIME, autorisés par le client (généralement un navigateur Web). Le service d'optimisation d'image tentera de convertir l'image au format spécifié. La valeur spécifiée pour cet en-tête aura une priorité supérieure à celle du paramètre de requête de format. Par exemple, une valeur valide pour l'en-tête Accept est`image/png, image/webp, */*` . Le paramètre de formats spécifié dans le manifeste de déploiement d'Amplify limitera les formats à ceux de la liste. Même si l'en-tête Accept demande un format spécifique, il sera ignoré si le format ne figure pas dans la liste d'autorisation.

Paramètres de demande URI

Le tableau suivant décrit les paramètres de demande d'URI pour l'optimisation des images.

Paramètre de la demande	Type	Obligatoire	Description	Exemple
url	String	Oui	Un chemin relatif ou une URL absolue vers l'image source. Pour une URL distante, seul le protocole https est pris en charge. La valeur doit être codée en URL.	`?url=https%3A%2F%2Fwww.example.com%2Fbuffalo.png`
width	Number	Oui	Largeur en pixels de l'image optimisée.	`?width=800`
height	Number	Non	Hauteur en pixels de l'image optimisée. Si ce n'est pas spécifié, l'image sera redimensionnée automatiquement pour correspondre à la largeur.	`?height=600`
ajuster	Valeurs d'énumération :`cover`,,`conta`	Non	Comment l'image est redimensionnée pour s'adapter à	`?width=800&height=600&fit=cover`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
	`inside` `outside`		la largeur et à la hauteur spécifiées.	
position	Valeurs d'énumération :`center`,,`top`, `bottom` `left`	Non	Une position à utiliser lorsque l'ajustement est `cover` ou`contain`.	`?fit=contain&position=centre`
trim	Number	Non	Découpe les pixels de tous les bords qui contiennent des valeurs similaires à la couleur d'arrière-plan spécifiée pour le pixel en haut à gauche.	`?trim=50`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
étendre	Objet	Non	Ajoute des pixels sur les bords de l'image en utilisant la couleur dérivée des pixels du bord le plus proche. Dans le format`{top}_{right}_{bottom}_{left}` , chaque valeur correspond au nombre de pixels à ajouter.	`?extend=10_0_5_0`
extract	Objet	Non	Recadre l'image dans le rectangle spécifié délimité par le haut, la gauche, la largeur et la hauteur. Le format est {left} _ {top} _ {width} _ {right} où chaque valeur est le nombre de pixels à recadrer.	`?extract=10_0_5_0`
format	String	Non	Format de sortie souhaité pour l'image optimisée .	`?format=webp`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
quality	Number	Non	La qualité de l'image, de 1 à 100. Utilisé uniquement lors de la conversion du format de l'image.	`?quality=50`
rotate	Number	Non	Fait pivoter l'image selon l'angle spécifié en degrés.	`?rotate=45`
retourner	Booléen	Non	Reflète l'image verticalement (de haut en bas) sur l'axe X. Cela se produit toujours avant la rotation, le cas échéant.	`?flip`
flop	Booléen	Non	Reflète l'image horizontalement (gauche-droite) sur l'axe Y. Cela se produit toujours avant la rotation, le cas échéant.	`?flop`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
affiner	Number	Non	La netteté améliore la définition des bords de l'image. Les valeurs valides sont comprises entre 0,000001 et 10.	`?sharpen=1`
median	Number	Non	Applique un filtre médian. Cela permet de supprimer le bruit ou de lisser les bords d'une image.	`?sharpen=3`
brouiller	Number	Non	Applique un flou gaussien du sigma spécifié. Les valeurs valides sont comprises entre 0,3 et 1 000.	`?blur=20`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
gamma	Number	Non	Applique une correction gamma pour améliorer la luminosité perçue d'une image redimensionnée. La valeur doit être comprise entre 1,0 et 3,0.	`?gamma=1`
nier	Booléen	Non	Inverse les couleurs de l'image.	`?negate`
normaliser	Booléen	Non	Améliore le contraste de l'image en étendant sa luminance pour couvrir une plage dynamique complète.	`?normalize`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
seuil	Number	Non	Remplace n'importe quel pixel de l'image par un pixel noir, si son intensité est inférieure au seuil spécifié. Ou avec un pixel blanc s'il est supérieur au seuil. Les valeurs valides sont comprises entre 0 et 255.	`?threshold=155`
teinte	String	Non	Teinte l'image à l'aide du RGB fourni tout en préservant la luminance de l'image.	`?tint=#7743CE`
niveaux de gris	Booléen	Non	Transforme l'image en niveaux de gris (noir et blanc).	`?grayscale`

Codes d'état des réponses

La liste suivante décrit les codes d'état de réponse pour l'optimisation des images.

Succès : code d'état HTTP 200

La demande a été traitée avec succès.

BadRequest - Code d'état HTTP 400

- Un paramètre de requête d'entrée n'a pas été spécifié correctement.
- L'URL distante n'est pas répertoriée comme autorisée dans le `remotePatterns` paramètre.
- L'URL distante ne se transforme pas en image.
- La largeur ou la hauteur demandées ne sont pas répertoriées comme autorisées dans le `sizes` paramètre.
- L'image demandée est au format SVG mais le `dangerouslyAllowSvg` paramètre est désactivé.

Introuvable - Code d'état HTTP 404

L'image source n'a pas été trouvée.

Contenu trop volumineux - code d'état HTTP 413

L'image source ou l'image optimisée dépassent la taille maximale autorisée en octets.

Comprendre la mise en cache optimisée des images

Amplify Hosting met en cache les images optimisées sur notre CDN afin que les demandes suivantes adressées à la même image, avec les mêmes paramètres de requête, soient traitées à partir du cache. Le temps de vie du cache (TTL) est contrôlé par l'`Cache-Control`en-tête. La liste suivante décrit les options qui s'offrent à vous pour spécifier l'`Cache-Control`en-tête.

- En utilisant la `Cache-Control` clé de la règle de routage qui cible l'optimisation de l'image.
- À l'aide d'en-têtes personnalisés définis dans l'application Amplify.
- Pour les images distantes, l'`Cache-Control`en-tête renvoyé par l'image distante est respecté.

La `minimumCacheTTL` valeur spécifiée dans les paramètres d'optimisation de l'image définit la limite inférieure de la Cache-Control max-age directive. Par exemple, si l'URL d'une image distante répond par un`Cache-Control s-max-age=10`, mais que la valeur de `minimumCacheTTL` est 60, alors 60 est utilisé.

Utilisation d'adaptateurs open source pour n'importe quel framework SSR

Vous pouvez utiliser n'importe quel adaptateur de construction de framework SSR créé pour être intégré à Amplify Hosting. Chaque infrastructure qui propose un adaptateur détermine la manière

dont l'adaptateur est configuré et connecté à son processus de création. En règle générale, vous installerez l'adaptateur en tant que dépendance de développement npm.

Après avoir créé une application avec un framework, consultez la documentation du framework pour savoir comment installer l'adaptateur Amplify Hosting et le configurer dans le fichier de configuration de votre application.

Créez ensuite un `amplify.yml` fichier dans le répertoire racine de votre projet. Dans le `amplify.yml` fichier, définissez le répertoire `baseDirectory` de sortie de compilation de votre application. Le framework exécute l'adaptateur pendant le processus de génération pour transformer la sortie en bundle de déploiement Amplify Hosting.

Le nom du répertoire de sortie de compilation peut être n'importe quoi, mais le `.amplify-hosting` nom du fichier a une importance. Amplify recherche d'abord un répertoire défini comme. `baseDirectory` S'il existe, Amplify y recherche la sortie de compilation. Si le répertoire n'existe pas, Amplify recherche la sortie de compilation qu'il contient`.amplify-hosting`, même si elle n'a pas été définie par le client.

Voici un exemple des paramètres de génération d'une application. Le `baseDirectory` est défini sur `.amplify-hosting` pour indiquer que la sortie de compilation se trouve dans le `.amplify-hosting` dossier. Tant que le contenu du `.amplify-hosting` dossier correspond aux spécifications de déploiement d'Amplify Hosting, l'application sera déployée avec succès.

```
version: 1
frontend:
  preBuild:
    commands:
      - npm install
  build:
    commands:
      - npm run build
  artifacts:
    baseDirectory: .amplify-hosting
```

Une fois que votre application est configurée pour utiliser un adaptateur de framework, vous pouvez la déployer sur Amplify Hosting. Pour obtenir les instructions complètes, consultez Déploiement d'une application SSR sur Amplify.

Déploiement d'un site Web statique vers Amplify à partir d'un compartiment Amazon S3

Vous pouvez utiliser l'intégration entre Amplify Hosting et Amazon S3 pour héberger du contenu de site Web statique stocké S3 en quelques clics. Le déploiement sur Amplify Hosting vous offre les avantages et fonctionnalités suivants.

- Déploiement automatique sur le réseau de diffusion de AWS contenu (CDN) disponible dans le monde entier alimenté par CloudFront
- prise en charge du HTTPS
- Connectez facilement votre site Web à un domaine personnalisé à l'aide de la console Amplify
- Apportez vos propres certificats SSL personnalisés
- Surveillez votre site Web à l'aide de journaux d'accès et de CloudWatch statistiques intégrés
- Configurez la protection par mot de passe pour votre site Web
- Créez des règles de redirection et de réécriture dans la console Amplify

Vous pouvez démarrer le processus de déploiement depuis la console Amplify, le AWS CLI, ou le. AWS SDKs Vous ne pouvez effectuer un déploiement vers Amplify qu'à partir d'un compartiment Amazon S3 à usage général situé dans votre propre compte. Amplify ne prend pas en charge l'accès aux compartiments entre comptes. S3

Lorsque vous déployez votre application depuis un bucket Amazon S3 à usage général vers Amplify Hosting, les AWS frais sont basés sur le modèle tarifaire d'Amplify. Pour plus d'informations, consultez Tarification d'AWS Amplify.

> **Important**
>
> Amplify Hosting n'est pas disponible partout Régions AWS où Amazon S3 est disponible. Pour déployer un site Web statique sur Amplify Hosting, le compartiment Amazon S3 à usage général contenant votre site Web doit être situé dans une région où Amplify est disponible. Pour obtenir la liste des régions où Amplify est disponible, consultez Points de terminaison Amplify dans le Référence générale d'Amazon Web Services.

Consultez les rubriques suivantes pour savoir comment déployer et mettre à jour un site Web statique depuis Amazon S3 vers Amplify Hosting.

Rubriques

Déploiement d'un site Web statique à S3 l'aide de la console Amplify

Suivez les instructions suivantes pour déployer un nouveau site Web statique à partir d'un compartiment à usage général Amazon S3 à l'aide de la console Amplify.

Pour déployer un site Web statique à partir d'un compartiment Amazon S3 à usage général à l'aide de la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez Deploy without Git.
4. Choisissez Suivant.
5. Sur la page Démarrer un déploiement manuel, procédez comme suit.
    a. Dans Nom de l'application, entrez le nom de votre application.
    b. Dans Nom de la branche, entrez le nom de la branche à déployer.
6. Pour Méthode, choisissez Amazon S3.
7. Pour l'S3emplacement des objets à héberger, choisissez Parcourir. Sélectionnez le compartiment à usage général Amazon S3 à utiliser, puis sélectionnez Choisir un préfixe.
8. Choisissez Save and deploy (Enregistrer et déployer).

Création d'une politique de compartiment pour déployer un site Web statique à S3 l'aide du AWS SDKs

Vous pouvez utiliser le AWS SDKs pour déployer un site Web statique depuis Amazon S3 vers Amplify Hosting. Si vous déployez votre site Web à l'aide d'un SDK, vous devez créer votre propre politique de compartiment qui accorde à Amplify Hosting l'autorisation de récupérer les objets de votre compartiment. S3

Pour en savoir plus sur la création de politiques de compartiment, consultez les politiques de compartiment pour Amazon S3 dans le guide de l'utilisateur d'Amazon Simple Storage Service.

L'exemple de politique de compartiment suivant accorde à Amplify Hosting les autorisations nécessaires pour répertorier les compartiments et récupérer les objets du compartiment pour l'identifiant d'application Compte AWS Amplify et la branche spécifiés.

Pour utiliser cet exemple :

- *amzn-s3-demo-website-bucket/prefix*Remplacez-le par le nom du bucket et du préfixe de votre site Web.
- *111122223333*Remplacez-le par votre Compte AWS identifiant.
- *region-id*Remplacez-le par Région AWS celui dans lequel se trouve votre application Amplify, tel que. **us-east-1**
- Remplacez *app_id* par votre identifiant d'application Amplify. Ces informations sont disponibles dans la console Amplify.
- Remplacez *branch_name* par le nom de votre succursale.

> **Note**
>
> Dans votre politique de compartiment, `aws:SourceArn` il doit s'agir d'un ARN de branche codé en URL (encodage en pourcentage).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowAmplifyToListPrefix_appid_branch_prefix_",
            "Effect": "Allow",
            "Principal": {
                "Service": "amplify.amazonaws.com"
            },
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-website-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "aws:SourceArn": "arn%3Aaws%3Aamplify%3Aregion-
id%3A111122223333%3Aapps%2Fapp_id%2Fbranches%2Fbranch_name",
                    "s3:prefix": ""
                }
            }
        },
        {
            "Sid": "AllowAmplifyToReadPrefix__appid_branch_prefix_",
            "Effect": "Allow",
            "Principal": {
                "Service": "amplify.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-website-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "aws:SourceArn": "arn%3Aaws%3Aamplify%3Aregion-
id%3A111122223333%3Aapps%2Fapp_id%2Fbranches%2Fbranch_name"
                }
            }
        },
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-website-bucket/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
```

```
    ]
}
```

Mettre à jour un site Web statique déployé sur Amplify à partir d'un bucket S3

Si vous mettez à jour l'un des objets d'un site Web statique dans un S3 bucket à usage général hébergé sur Amplify, vous devez redéployer l'application sur Amplify Hosting pour que les modifications prennent effet. Amplify Hosting ne détecte pas automatiquement les modifications apportées au S3 bucket. Nous vous recommandons d'utiliser la AWS Command Line Interface (CLI) pour mettre à jour votre site Web.

Synchroniser les mises à jour avec S3

Après avoir apporté des modifications aux fichiers de projet de votre site Web, utilisez la commande de synchronisation s3 suivante pour synchroniser les modifications que vous avez apportées à votre répertoire source local avec votre compartiment Amazon S3 à usage général cible. Pour utiliser cet exemple, remplacez-le *<source>* par le nom de votre répertoire local et *<target>* par le nom de votre compartiment Amazon S3.

```
aws s3 sync <source> <target>
```

Redéployer le site Web vers Amplify Hosting

Utilisez la commande amplify start-deployment suivante pour redéployer votre application mise à jour dans un compartiment Amazon S3 vers Amplify Hosting. Pour utiliser cet exemple, remplacez-le *<app_id>* par l'identifiant de votre application Amplify, *<branch_name>* par le nom de votre branche, ainsi que *s3://amzn-s3-demo-website-bucket/prefix* par votre S3 bucket et votre préfixe.

```
aws amplify start-deployment --app-id <app_id> --branch-name <branch_name> --source-
url s3://amzn-s3-demo-website-bucket/prefix --source-url-type BUCKET_PREFIX
```

Mettre à jour un S3 déploiement pour utiliser un compartiment et un préfixe au lieu d'un fichier .zip

Si vous avez déjà déployé un site Web statique sur Amplify Hosting à partir d'un fichier .zip dans un compartiment à usage général Amazon S3, vous pouvez mettre à jour le déploiement de l'application pour utiliser le nom et le préfixe du compartiment contenant les objets à héberger. Ce type de déploiement élimine le besoin de télécharger un fichier distinct dans votre compartiment contenant le contenu compressé de la sortie de compilation.

Pour migrer un site Web statique d'un fichier .zip vers le contenu du bucket

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez le nom de l'application déployée manuellement que vous souhaitez faire migrer depuis un fichier .zip vers une utilisation directe des fichiers de l'application.
3. Sur la page de présentation de l'application, choisissez Déployer les mises à jour.
4. Sur la page Déployer les mises à jour, pour Méthode, choisissez Amazon S3.
5. Pour l'S3emplacement des objets à héberger, choisissez Parcourir. Sélectionnez le compartiment à utiliser, puis sélectionnez Choisir un préfixe.
6. Choisissez Save and deploy (Enregistrer et déployer).

Déploiement d'une application sur Amplify sans dépôt Git

Les déploiements manuels vous permettent de publier votre application Web avec Amplify Hosting sans connecter un fournisseur Git. Vous pouvez glisser-déposer un dossier compressé depuis votre bureau et héberger votre site en quelques secondes. Vous pouvez également référencer des actifs dans un compartiment Amazon S3 ou spécifier une URL publique vers l'emplacement où vos fichiers sont stockés.

> **Note**
>
> Les déploiements manuels ont une limite maximale de taille de fichier .zip de 5 Go en raison des contraintes liées aux opérations de copie d'Amazon S3. Si l'un de vos artefacts de build dépasse cette taille, pensez à le diviser en archives plus petites ou à utiliser une autre méthode de déploiement.

Pour Amazon S3, vous pouvez également configurer des AWS Lambda déclencheurs pour mettre à jour votre site chaque fois que de nouvelles ressources sont téléchargées. Consultez le billet de blog Déployer des fichiers stockés sur Amazon S3, Dropbox ou votre ordinateur de bureau sur la AWS Amplify console pour plus de détails sur la configuration de ce scénario.

Amplify Hosting ne prend pas en charge les déploiements manuels pour les applications de rendu côté serveur (SSR). Pour de plus amples informations, veuillez consulter Déploiement d'applications rendues côté serveur avec Amplify Hosting.

Déploiements manuels par glisser-déposer

Pour déployer manuellement une application par glisser-déposer

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Dans le coin supérieur droit, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez Deploy without Git. Ensuite, choisissez Suivant.
4. Sur la page Démarrer un déploiement manuel, dans le champ Nom de l'application, entrez le nom de votre application.
5. Dans Nom de la branche, entrez un nom significatif, tel que **`development`** ou**`production`**.

6. Dans Méthode, choisissez Drag and drop.
7. Faites glisser un dossier de votre bureau vers la zone de dépôt ou utilisez l'option Choisir un dossier .zip pour sélectionner le fichier sur votre ordinateur. Le fichier que vous glissez et déposez ou sélectionnez doit être un dossier compressé contenant le contenu de votre sortie de compilation.
8. Choisissez Save and deploy (Enregistrer et déployer).

Amazon S3 ou déploiement manuel d'URL

> **Note**
>
> Si vous déployez un site Web statique à partir deS3, la procédure suivante nécessite que vous téléchargiez un dossier compressé contenant le contenu de votre sortie de build dans votre S3 compartiment. Nous vous recommandons de déployer un site Web statique directement à S3 partir du nom et du préfixe du bucket. Pour plus d'informations sur ce processus simplifié, consultezDéploiement d'un site Web statique vers Amplify à partir d'un compartiment Amazon S3.

Pour déployer manuellement une application depuis Amazon S3 ou une URL publique

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Dans le coin supérieur droit, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez Deploy without Git. Ensuite, choisissez Suivant.
4. Sur la page Démarrer un déploiement manuel, dans le champ Nom de l'application, entrez le nom de votre application.
5. Dans Nom de la branche, entrez un nom significatif, tel que **`development`** ou**`production`**.
6. Pour Méthode, choisissez Amazon S3 ou n'importe quelle URL.
7. La procédure de téléchargement de vos fichiers dépend de la méthode de téléchargement.
   - Amazon S3
     a. Pour S3 location of objects to host, choisissez Parcourir S3. Sélectionnez ensuite le nom du compartiment Amazon S3 dans la liste. Les listes de contrôle d'accès (ACLs) doivent être activées pour le compartiment que vous sélectionnez. Pour de plus amples

informations, veuillez consulter Résolution des problèmes d'accès au compartiment Amazon S3 pour les déploiements manuels.

b. Sélectionnez le nom du fichier .zip à déployer.

c. Choisissez Choisir un préfixe.

- N'importe quelle URL

  - Pour URL de ressource, entrez l'URL du fichier .zip à déployer.

8. Choisissez Save and deploy (Enregistrer et déployer).

> **Note**
>
> Lorsque vous créez le dossier compressé, assurez-vous de compresser le contenu de votre sortie de compilation et non le dossier de niveau supérieur. Par exemple, si votre sortie de compilation génère un dossier nommé « build » ou « public », naviguez d'abord dans ce dossier, sélectionnez tout le contenu et compressez-le à partir de là. Si vous ne le faites pas, le message d'erreur « Accès refusé » s'affichera car le répertoire racine du site ne sera pas initialisé correctement.

Résolution des problèmes d'accès au compartiment Amazon S3 pour les déploiements manuels

Lorsque vous créez un compartiment Amazon S3, vous utilisez son paramètre Amazon S3 Object Ownership pour contrôler si les listes de contrôle d'accès (ACLs) sont activées ou désactivées pour le compartiment. Pour déployer manuellement une application sur Amplify à partir d'un compartiment Amazon S3, celle-ci ACLs doit être activée sur le compartiment.

Si un `AccessControlList` message d'erreur s'affiche lorsque vous déployez à partir d'un compartiment Amazon S3, cela signifie que le compartiment a été créé et que vous devez l'activer dans la console Amazon S3. ACLs Pour obtenir des instructions, consultez la section Définition de la propriété d'un objet sur un compartiment existant dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Gestion de la configuration de compilation pour une application Amplify

Vous pouvez personnaliser les paramètres de compilation et la configuration de vos déploiements Amplify. Lorsque vous déployez une application, Amplify détecte automatiquement le framework frontal et les paramètres de build associés. Vous pouvez personnaliser les paramètres de construction dans la spécification de construction de l'application (buildspec) pour ajouter des variables d'environnement, exécuter des commandes de génération et spécifier les dépendances de construction.

L'image de construction par défaut d'Amplify est fournie avec plusieurs packages et dépendances préinstallés, mais vous pouvez également utiliser la fonctionnalité de mise à jour des packages en direct pour spécifier une version spécifique ou vous assurer que la dernière version est toujours installée. Si vous avez des dépendances spécifiques dont l'installation prend du temps lors d'une compilation à l'aide du conteneur par défaut d'Amplify, vous pouvez créer votre propre image de build personnalisée. Vous pouvez également personnaliser la taille de l'instance de build afin de fournir à votre déploiement d'applications les ressources de processeur, de mémoire et d'espace disque dont il a besoin.

Les builds sont lancés automatiquement à chaque validation dans votre dépôt Git et à chaque nouveau déploiement. Vous pouvez configurer la fonctionnalité de webhooks entrants pour lancer une compilation sans validation dans votre dépôt Git.

La fonctionnalité de notifications de build vous permet de partager des informations avec les membres de l'équipe sur les réussites et les échecs des builds.

Rubriques

Configuration des paramètres de compilation pour une application Amplify

Lorsque vous déployez une application, Amplify détecte automatiquement le framework frontal et les paramètres de build associés en inspectant le fichier de l'application `package.json` dans votre référentiel Git. Vous disposez des options suivantes pour stocker les paramètres de compilation de votre application :

- Enregistrez les paramètres de build dans la console Amplify - La console Amplify détecte automatiquement les paramètres de build et les enregistre afin que la console Amplify puisse y accéder. Amplify applique ces paramètres à toutes vos branches, sauf si un `amplify.yml` fichier est stocké dans votre référentiel.
- Enregistrez les paramètres de compilation dans votre dépôt : téléchargez le `amplify.yml` fichier et ajoutez-le à la racine de votre dépôt.

> **Note**
>
> Les paramètres de compilation ne sont visibles dans le menu d'hébergement de la console Amplify que lorsqu'une application est configurée pour un déploiement continu et connectée à un référentiel git. Pour obtenir des instructions sur ce type de déploiement, consultez Getting started.

Référence de spécification de construction

La spécification de construction (buildspec) d'une application Amplify est un ensemble de paramètres YAML et de commandes de génération qu'Amplify utilise pour exécuter votre build. La liste suivante décrit ces paramètres et leur mode d'utilisation.

version

Le numéro de version d'Amplify YAML.

par Root

Le chemin dans le référentiel dans lequel réside cette application. Ignoré sauf si plusieurs applications sont définies.

env

Ajoutez des variables d'environnement à cette section. Vous pouvez également ajouter des variables d'environnement à l'aide de la console.

dorsal

Exécutez les commandes Amplify CLI pour provisionner un backend, mettre à jour des fonctions Lambda ou des schémas GraphQL dans le cadre d'un déploiement continu.

frontend

Exécutez les commandes de construction du frontend.

test

Exécutez des commandes pendant une phase de test. Découvrez comment ajouter des tests à votre application.

phases de construction

Le frontend, le backend et le test comportent trois phases qui représentent les commandes exécutées au cours de chaque séquence de construction.

- PreBuild - Le script PreBuild s'exécute avant le début de la compilation proprement dite, mais après qu'Amplify ait installé les dépendances.
- génération : vos commandes de génération.
- PostBuild - Le script post-build s'exécute une fois la compilation terminée et Amplify a copié tous les artefacts nécessaires dans le répertoire de sortie.

chemin de construction

Le chemin à utiliser pour exécuter le build. Amplify utilise ce chemin pour localiser les artefacts de votre build. Si vous ne spécifiez pas de chemin, Amplify utilise la racine de l'application monorepo, par exemple. `apps/app`

artéfacts > répertoire de base

Le répertoire dans lequel se trouvent vos artefacts de build.

artéfacts > fichiers

Spécifiez les fichiers à partir de vos artefacts que vous souhaitez déployer. Entrez `**/*` pour inclure tous les fichiers.

cache

Spécifie les dépendances au moment de la construction, telles que le dossier node_modules. Lors de la première génération, les chemins fournis ici sont mis en cache. Lors des versions suivantes, Amplify restaure le cache sur les mêmes chemins avant d'exécuter vos commandes.

Amplify considère que tous les chemins de cache fournis sont relatifs à la racine de votre projet. Cependant, Amplify n'autorise pas le passage en dehors de la racine du projet. Par exemple, si vous spécifiez un chemin absolu, la génération réussira sans erreur, mais le chemin ne sera pas mis en cache.

Référence de syntaxe YAML pour les spécifications de construction

L'exemple suivant de spécification de construction illustre la syntaxe YAML de base.

```
version: 1
env:
  variables:
    key: value
backend:
  phases:
    preBuild:
      commands:
        - *enter command*
    build:
      commands:
        - *enter command*
    postBuild:
        commands:
        - *enter command*
frontend:
  buildpath:
  phases:
    preBuild:
      commands:
        - cd react-app
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    files:
```

```
        - location
        - location
    discard-paths: yes
    baseDirectory: location
  cache:
    paths:
        - path # A cache path relative to the project root
        - path # Traversing outside of the project root is not allowed
test:
  phases:
    preTest:
      commands:
        - *enter command*
    test:
      commands:
        - *enter command*
    postTest:
      commands:
        - *enter command*
  artifacts:
    files:
        - location
        - location
    configFilePath: *location*
    baseDirectory: *location*
```

Modification de la spécification de construction

Vous pouvez personnaliser les paramètres de compilation d'une application en modifiant la spécification de construction (buildspec) dans la console Amplify. Les paramètres de construction sont appliqués à toutes les branches de votre application, à l'exception des branches dont un `amplify.yml` fichier est enregistré dans le référentiel Git.

Pour modifier les paramètres de compilation dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application dont vous souhaitez modifier les paramètres de compilation.
3. Dans le volet de navigation, choisissez Hosting, puis Build settings.
4. Sur la page des paramètres de construction, dans la section Spécification de construction de l'application, choisissez Modifier.
5. Dans la fenêtre Modifier les spécifications de construction, entrez vos mises à jour.

6. Choisissez Enregistrer.

Vous pouvez utiliser les exemples décrits dans les rubriques suivantes pour mettre à jour vos paramètres de compilation pour des scénarios spécifiques.

Rubriques

Définition des paramètres de construction spécifiques à une branche à l'aide de scripts

Vous pouvez utiliser des scripts shell bash pour définir des paramètres de build spécifiques à la branche. Par exemple, le script suivant utilise la variable d'environnement système $ AWS_BRANCH pour exécuter un ensemble de commandes si le nom de la branche est main et un autre ensemble de commandes si le nom de la branche est dev.

```
frontend:
  phases:
    build:
      commands:
        - if [ "${AWS_BRANCH}" = "main" ]; then echo "main branch"; fi
        - if [ "${AWS_BRANCH}" = "dev" ]; then echo "dev branch"; fi
```

Configuration d'une commande pour accéder à un sous-dossier

Pour monorepos, les utilisateurs veulent pouvoir accéder à un `cd` dossier pour exécuter le build. Une fois que vous avez exécuté la `cd` commande, elle s'applique à toutes les étapes de votre build, vous n'avez donc pas besoin de répéter la commande en plusieurs phases.

```
version: 1
env:
  variables:
    key: value
frontend:
  phases:
    preBuild:
      commands:
        - cd react-app
        - npm ci
    build:
      commands:
        - npm run build
```

Déploiement du backend avec le front-end d'une application de première génération

> **Note**
>
> Cette section s'applique uniquement aux applications Amplify Gen 1. Un backend Gen 1 est créé à l'aide d'Amplify Studio et de l'interface de ligne de commande (CLI) Amplify.

La `amplifyPush` commande est un script d'assistance qui vous aide dans les déploiements du backend. Les paramètres de build ci-dessous déterminent automatiquement l'environnement backend approprié à déployer pour la branche active.

```
version: 1
env:
  variables:
    key: value
backend:
  phases:
    build:
      commands:
```

```
        - amplifyPush --simple
```

Configuration du dossier de sortie

Les paramètres de génération suivants définissent le répertoire de sortie sur le dossier public.

```
frontend:
  phases:
    commands:
      build:
        - yarn run build
  artifacts:
    baseDirectory: public
```

Installation de packages dans le cadre d'une compilation

Vous pouvez utiliser les `yarn` commandes `npm` or pour installer des packages lors de la compilation.

```
frontend:
  phases:
    build:
      commands:
        - npm install -g <package>
        - <package> deploy
        - yarn run build
  artifacts:
    baseDirectory: public
```

Utilisation d'un registre npm privé

Vous pouvez ajouter des références à un registre privé dans vos paramètres de génération ou l'ajouter en tant que variable d'environnement.

```
build:
  phases:
    preBuild:
      commands:
        - npm config set <key> <value>
        - npm config set registry https://registry.npmjs.org
        - npm config set always-auth true
        - npm config set email hello@amplifyapp.com
```

```
        - yarn install
```

Installation de packages de système d'exploitation

AL2023 L'image d'Amplify exécute votre code avec un nom d'utilisateur non privilégié. `amplify` Amplify accorde à cet utilisateur les privilèges nécessaires pour exécuter des commandes du système d'exploitation à l'aide de la commande Linux. `sudo` Si vous souhaitez installer des packages de système d'exploitation pour les dépendances manquantes, vous pouvez utiliser des commandes telles que `yum` et `rpm` avec`sudo`.

L'exemple de section de construction suivant illustre la syntaxe d'installation d'un package de système d'exploitation à l'aide de la `sudo` commande.

```
build:
  phases:
    preBuild:
      commands:
        - sudo yum install -y <package>
```

Configuration du stockage des valeurs clés pour chaque version

`envCache`Fournit un stockage des valeurs clés au moment de la construction. Les valeurs stockées dans le ne `envCache` peuvent être modifiées que lors d'une construction et peuvent être réutilisées lors de la prochaine génération. À l'aide de`envCache`, nous pouvons stocker des informations sur l'environnement déployé et les mettre à la disposition du conteneur de construction lors de versions successives. Contrairement aux valeurs stockées dans le`envCache`, les modifications apportées aux variables d'environnement au cours d'une génération ne sont pas conservées dans les versions futures.

Exemple d'utilisation :

```
envCache --set <key> <value>
envCache --get <key>
```

Ignorer le build pour un commit

Pour ignorer une compilation automatique sur un commit particulier, incluez le texte [skip-cd] à la fin du message de validation.

Désactiver les builds automatiques à chaque commit

Vous pouvez configurer Amplify pour désactiver les compilations automatiques à chaque validation de code. Pour le configurer, choisissez Paramètres de l'application, Paramètres des succursales, puis recherchez la section Branches qui répertorie les succursales connectées. Sélectionnez une branche, puis choisissez Actions, Désactiver la construction automatique. Les nouvelles validations vers cette branche ne démarreront plus une nouvelle version.

Configuration de la création et du déploiement d'un frontend basé sur Diff

Vous pouvez configurer Amplify pour utiliser des versions frontales basées sur les différences. Si cette option est activée, au début de chaque build, Amplify tente d'exécuter une différence sur votre `appRoot` dossier ou sur le `/src/` dossier par défaut. Si Amplify ne trouve aucune différence, il ignore les étapes de création, de test (si configuré) et de déploiement du frontend, et ne met pas à jour votre application hébergée.

Pour configurer la création et le déploiement d'un frontend basé sur Diff

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle configurer la création et le déploiement du frontend basé sur les différences.
3. Dans le volet de navigation, choisissez Hosting, Environment variables.
4. Dans la section Variables d'environnement, sélectionnez Gérer les variables.
5. La procédure de configuration de la variable d'environnement varie selon que vous activez ou désactivez la création et le déploiement du frontend basé sur le diff.
   - Pour activer la création et le déploiement d'une interface basée sur les différences
     a. Dans la section Gérer les variables, sous Variable, entrez`AMPLIFY_DIFF_DEPLOY`.
     b. Pour le champ Valeur, saisissez `true`.
   - Pour désactiver la création et le déploiement du frontend basé sur les différences
     - Effectuez l'une des actions suivantes :
       - Dans la section Gérer les variables, recherchez`AMPLIFY_DIFF_DEPLOY`. Pour le champ Valeur, saisissez `false`.
       - Supprimez la variable d'`AMPLIFY_DIFF_DEPLOY`environnement.
6. Choisissez Enregistrer.

Vous pouvez éventuellement définir la variable d'`AMPLIFY_DIFF_DEPLOY_ROOT`environnement pour remplacer le chemin par défaut par un chemin relatif à la racine de votre dépôt, tel que. `dist`

Configuration de versions de backend basées sur les différences pour une application de première génération

> **Note**
>
> Cette section s'applique uniquement aux applications Amplify Gen 1. Un backend Gen 1 est créé à l'aide d'Amplify Studio et de l'interface de ligne de commande (CLI) Amplify.

Vous pouvez configurer Amplify Hosting pour utiliser des versions de backend basées sur les différences à l'aide de la variable d'environnement. `AMPLIFY_DIFF_BACKEND` Lorsque vous activez les versions de backend basées sur le diff, au début de chaque build, Amplify tente d'exécuter un diff sur le `amplify` dossier de votre référentiel. Si Amplify ne trouve aucune différence, il ignore l'étape de création du backend et ne met pas à jour vos ressources backend. Si votre projet ne contient aucun `amplify` dossier dans votre référentiel, Amplify ignore la valeur de la `AMPLIFY_DIFF_BACKEND` variable d'environnement.

Si des commandes personnalisées sont actuellement spécifiées dans les paramètres de génération de votre phase de backend, les builds de backend conditionnels ne fonctionneront pas. Si vous souhaitez que ces commandes personnalisées s'exécutent, vous devez les déplacer vers la phase frontale de vos paramètres de génération dans le `amplify.yml` fichier de votre application.

Pour configurer des versions de backend basées sur les différences

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle configurer les versions de backend basées sur les différences.
3. Dans le volet de navigation, choisissez Hosting, Environment variables.
4. Dans la section Variables d'environnement, sélectionnez Gérer les variables.
5. La procédure de configuration de la variable d'environnement varie selon que vous activez ou désactivez les versions de backend basées sur les différences.
   - Pour activer les builds de backend basés sur les différences
     a. Dans la section Gérer les variables, sous Variable, entrez`AMPLIFY_DIFF_BACKEND`.

b. Pour le champ Valeur, saisissez `true`.

- Pour désactiver les builds de backend basés sur les différences

  - Effectuez l'une des actions suivantes :

    - Dans la section Gérer les variables, recherchez`AMPLIFY_DIFF_BACKEND`. Pour le champ Valeur, saisissez `false`.
    - Supprimez la variable d'`AMPLIFY_DIFF_BACKEND`environnement.

6. Choisissez Enregistrer.

Configuration des paramètres de compilation de monorepo

Lorsque vous stockez plusieurs projets ou microservices dans un seul référentiel, cela s'appelle un monorepo. Vous pouvez utiliser Amplify Hosting pour déployer des applications dans un monorepo sans créer plusieurs configurations de build ou de branche.

Amplify prend en charge les applications en monorepos génériques ainsi que les applications en monorepos créées à l'aide de npm workspace, pnpm workspace, Yarn workspace, Nx et Turborepo. Lorsque vous déployez votre application, Amplify détecte automatiquement l'outil de génération monorepo que vous utilisez. Amplify applique automatiquement les paramètres de génération pour les applications dans un espace de travail npm, un espace de travail Yarn ou Nx. Les applications Turborepo et pnpm nécessitent une configuration supplémentaire. Pour de plus amples informations, veuillez consulter Configuration des applications Turborepo et pnpm monorepo.

Vous pouvez enregistrer les paramètres de compilation d'un monorepo dans la console Amplify ou vous pouvez télécharger le `amplify.yml` fichier et l'ajouter à la racine de votre référentiel. Amplify applique les paramètres enregistrés dans la console à toutes vos branches sauf s'il trouve un `amplify.yml` fichier dans votre référentiel. Lorsqu'un `amplify.yml` fichier est présent, ses paramètres remplacent tous les paramètres de compilation enregistrés dans la console Amplify.

Référence syntaxique YAML de la spécification de build Monorepo

La syntaxe YAML pour une spécification de build monorepo est différente de la syntaxe YAML pour un dépôt contenant une seule application. Pour un monorepo, vous déclarez chaque projet dans une liste d'applications. Vous devez fournir la `appRoot` clé supplémentaire suivante pour chaque application que vous déclarez dans la spécification de construction de votre monorepo :

par Root

Racine, au sein du référentiel, dans laquelle l'application démarre. Cette clé doit exister et avoir la même valeur que la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement. Pour obtenir des instructions sur la définition de cette variable d'environnement, consultezConfiguration de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT.

L'exemple de spécification de construction de monorepo suivant montre comment déclarer plusieurs applications Amplify dans le même dépôt. Les deux applications`react-app`, et `angular-app` sont déclarées dans la `applications` liste. La `appRoot` clé de chaque application indique que l'application se trouve dans le dossier `apps` racine du dépôt.

L'`buildpath`attribut est défini pour exécuter et créer l'application `/` à partir de la racine du projet monorepo. L'`baseDirectory`attribut est le chemin relatif de`buildpath`.

Syntaxe YAML de la spécification de construction Monorepo

```
version: 1
applications:
  - appRoot: apps/react-app
    env:
      variables:
        key: value
    backend:
      phases:
        preBuild:
          commands:
            - *enter command*
        build:
          commands:
            - *enter command*
        postBuild:
            commands:
            - *enter command*
    frontend:
      buildPath: / # Run install and build from the monorepo project root
      phases:
        preBuild:
          commands:
            - *enter command*
            - *enter command*
        build:
```

```
        commands:
          - *enter command*
    artifacts:
      files:
          - location
          - location
      discard-paths: yes
      baseDirectory: location
    cache:
      paths:
          - path
          - path
  test:
    phases:
      preTest:
        commands:
          - *enter command*
      test:
        commands:
          - *enter command*
      postTest:
        commands:
          - *enter command*
    artifacts:
      files:
          - location
          - location
      configFilePath: *location*
      baseDirectory: *location*
- appRoot: apps/angular-app
  env:
    variables:
      key: value
  backend:
    phases:
      preBuild:
        commands:
          - *enter command*
      build:
        commands:
          - *enter command*
      postBuild:
          commands:
          - *enter command*
```

```
    frontend:
      phases:
        preBuild:
          commands:
            - *enter command*
            - *enter command*
        build:
          commands:
            - *enter command*
      artifacts:
        files:
            - location
            - location
        discard-paths: yes
        baseDirectory: location
      cache:
        paths:
            - path
            - path
    test:
      phases:
        preTest:
          commands:
            - *enter command*
        test:
          commands:
            - *enter command*
        postTest:
          commands:
            - *enter command*
      artifacts:
        files:
            - location
            - location
        configFilePath: *location*
        baseDirectory: *location*
```

Une application utilisant l'exemple de spécification de construction suivant sera créée sous la racine du projet et les artefacts de construction seront situés à l'adresse `/packages/nextjs-app/.next`.

```
applications:
  - frontend:
```

```
      buildPath: '/'  # run install and build from monorepo project root
      phases:
        preBuild:
          commands:
            - npm install
        build:
          commands:
            - npm run build --workspace=nextjs-app
      artifacts:
        baseDirectory: packages/nextjs-app/.next
        files:
          - '**/*'
      cache:
        paths:
          - node_modules/**/*
    appRoot: packages/nextjs-app
```

Configuration de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT

Lorsque vous déployez une application stockée dans un monorepo, la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement de l'application doit avoir la même valeur que le chemin de la racine de l'application, par rapport à la racine de votre dépôt. Par exemple, un monorepo nommé `ExampleMonorepo` avec un dossier racine nommé`apps`, qui contient, `app1``app2`, et `app3` possède la structure de répertoires suivante :

```
ExampleMonorepo
  apps
    app1
    app2
    app3
```

Dans cet exemple, la valeur de la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement for `app1` est`apps/app1`.

Lorsque vous déployez une application monorepo à l'aide de la console Amplify, celle-ci définit automatiquement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement en utilisant la valeur que vous spécifiez pour le chemin d'accès à la racine de l'application. Toutefois, si votre application monorepo existe déjà dans Amplify ou est déployée à l'aide de AWS CloudFormation, vous devez définir manuellement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement dans la section Variables d'environnement de la console Amplify.

Définition automatique de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT lors du déploiement

Les instructions suivantes montrent comment déployer une application monorepo avec la console Amplify. Amplify définit automatiquement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement à l'aide du dossier racine de l'application que vous spécifiez dans la console.

Pour déployer une application monorepo avec la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez Créer une nouvelle application dans le coin supérieur droit.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur Git, puis Next.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :
   a. Choisissez le nom de votre dépôt dans la liste.
   b. Choisissez le nom de la branche à utiliser.
   c. Sélectionnez Mon application est un monorepo
   d. Entrez le chemin d'accès à votre application dans votre monorepo, par exemple,. **`apps/app1`**
   e. Choisissez Suivant.
5. Sur la page des paramètres de l'application, vous pouvez utiliser les paramètres par défaut ou personnaliser les paramètres de génération de votre application. Dans la section Variables d'environnement, Amplify définit le chemin que vous avez spécifié `AMPLIFY_MONOREPO_APP_ROOT` à l'étape 4d.
6. Choisissez Suivant.
7. Sur la page Révision, choisissez Enregistrer et déployer.

Définition de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT pour une application existante

Utilisez les instructions suivantes pour définir manuellement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement pour une application déjà déployée sur Amplify ou créée à l'aide de. CloudFormation

Pour définir la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT pour une application existante

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez le nom de l'application pour laquelle définir la variable d'environnement.
3. Dans le volet de navigation, choisissez Hosting, puis choisissez Environment variables.
4. Sur la page Variables d'environnement, sélectionnez Gérer les variables.
5. Dans la section Gérer les variables, procédez comme suit :
   a. Choisissez Add new (Ajouter nouveau).
   b. Pour Variable, entrez la cléAMPLIFY_MONOREPO_APP_ROOT.
   c. Pour Value, entrez le chemin d'accès à l'application, par exemple**apps/app1**.
   d. Pour Branch, Amplify applique par défaut la variable d'environnement à toutes les branches.
6. Choisissez Enregistrer.

Configuration des applications Turborepo et pnpm monorepo

Les outils de génération monorepo de Turborepo et pnpm workspace obtiennent des informations de configuration à partir de fichiers. `.npmrc` Lorsque vous déployez une application monorepo créée avec l'un de ces outils, vous devez avoir un `.npmrc` fichier dans le répertoire racine de votre projet.

Dans le `.npmrc` fichier, définissez l'éditeur de liens pour l'installation des packages Node sur`hoisted`. Vous pouvez copier la ligne suivante dans votre fichier.

```
node-linker=hoisted
```

Pour plus d'informations sur *`.npmrc`* les fichiers et les paramètres, consultez pnpm .npmrc dans la documentation de pnpm.

Pnpm n'est pas inclus dans le conteneur de build par défaut d'Amplify. Pour les applications pnpm workspace et Turborepo, vous devez ajouter une commande pour installer pnpm dans la `preBuild` phase des paramètres de compilation de votre application.

L'exemple d'extrait suivant d'une spécification de construction montre une `preBuild` phase avec une commande pour installer pnpm.

```
version: 1
```

```
applications:
  - frontend:
      phases:
        preBuild:
          commands:
            - npm install -g pnpm
```

Personnalisation de l'image de construction

Vous pouvez utiliser une image de construction personnalisée pour fournir un environnement de génération personnalisé pour une application Amplify. Si vous avez des dépendances spécifiques dont l'installation prend du temps lors d'une compilation à l'aide du conteneur par défaut d'Amplify, vous pouvez créer votre propre image Docker et la référencer lors d'une compilation. Les images peuvent être hébergées sur Amazon Elastic Container Registry Public.

Pour qu'une image de construction personnalisée fonctionne comme une image de génération Amplify, elle doit répondre aux exigences suivantes.

Exigences relatives aux images de construction personnalisées

1. Une distribution Linux qui supporte la bibliothèque GNU C (glibc), telle qu'Amazon Linux, compilée pour l'architecture x86-64.
2. cURL : lorsque vous lancez l'image personnalisée, l'exécuteur de build est téléchargé dans votre conteneur. C'est pourquoi il est nécessaire d'indiquer l'attribut cURL. Si cette dépendance est absente, le build échoue instantanément sans aucune sortie car notre build-runner n'est pas en mesure de produire de sortie.
3. Git : afin de pouvoir cloner le référentiel Git, Git doit être installé dans l'image. Si cette dépendance est absente, l'étape de clonage du référentiel échouera.
4. OpenSSH : afin de cloner votre dépôt en toute sécurité, nous avons besoin qu'OpenSSH configure temporairement la clé SSH pendant la compilation. Le package OpenSSH fournit les commandes dont le lanceur de compilation a besoin pour ce faire.
5. Bash et The Bourne Shell : ces deux utilitaires sont utilisés pour exécuter des commandes au moment de la construction. S'ils ne sont pas installés, vos builds risquent d'échouer avant de démarrer.
6. Node.js+npm : Notre lanceur de build n'installe pas Node. Il repose plutôt sur l'installation de Node et de NPM dans l'image. Cette exigence ne s'applique que pour les builds impliquant des packages NPM ou des commandes Node spécifiques. Cependant, nous vous recommandons

vivement de les installer car lorsqu'ils sont présents, le lanceur de build Amplify peut utiliser ces outils pour améliorer l'exécution de la compilation. La fonction de remplacement de package d'Amplify utilise NPM pour installer le package Hugo-Extended lorsque vous définissez une dérogation pour Hugo.

Les packages suivants ne sont pas obligatoires, mais nous vous recommandons vivement de les installer.

1. NVM (Node Version Manager): Nous vous recommandons d'installer ce gestionnaire de version si vous devez gérer différentes versions deNode. Lorsque vous définissez une dérogation, la fonction de remplacement de package d'Amplify permet de modifier les versions NVM de Node.js avant chaque build.
2. Wget: Amplify peut utiliser l'Wgetutilitaire pour télécharger des fichiers pendant le processus de compilation. Nous vous recommandons de l'installer dans votre image personnalisée.
3. Tar : Amplify peut utiliser l'Tarutilitaire pour décompresser les fichiers téléchargés pendant le processus de compilation. Nous vous recommandons de l'installer dans votre image personnalisée.

Configuration d'une image de build personnalisée pour une application

Utilisez la procédure suivante pour configurer une image de build personnalisée pour une application dans la console Amplify.

Pour configurer une image de build personnalisée hébergée dans Amazon ECR

1. Consultez Getting started dans le guide de l'utilisateur Amazon ECR Public pour configurer un référentiel Amazon ECR Public avec une image Docker.
2. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
3. Choisissez l'application pour laquelle vous souhaitez configurer une image de build personnalisée.
4. Dans le volet de navigation, choisissez Hosting, Build settings.
5. Sur la page Paramètres de création, dans la section Paramètres de création d'image, choisissez Modifier.
6. Sur la page Modifier les paramètres de l'image de construction, développez le menu Créer une image et choisissez Custom Build Image.

7. Entrez le nom du dépôt Amazon ECR Public que vous avez créé à la première étape. C'est ici que votre image de build est hébergée. Par exemple, si le nom de votre dépôt est ecr-examplerepo, vous devez entrer. **`public.ecr.aws/xxxxxxxx/ecr-examplerepo`**
8. Choisissez Enregistrer.

Utilisation de versions de package et de dépendance spécifiques dans l'image de construction

Les mises à jour des packages en direct vous permettent de spécifier les versions des packages et des dépendances à utiliser dans l'image de compilation par défaut d'Amplify. L'image de construction par défaut est fournie avec plusieurs packages et dépendances préinstallés (par exemple Hugo, Amplify CLI, Yarn, etc.). Avec les mises à jour de packages en direct, vous pouvez remplacer la version de ces dépendances et spécifier une version spécifique ou vous assurer que la dernière version est toujours installée.

Si les mises à jour des packages en direct sont activées, avant l'exécution de votre build, le build runner met d'abord à jour (ou rétrograde) les dépendances spécifiées. Cela augmente le temps de création proportionnellement au temps nécessaire pour mettre à jour les dépendances, mais l'avantage est que vous pouvez vous assurer que la même version d'une dépendance est utilisée pour créer votre application.

> **Warning**
>
> La configuration de la version de Node.js sur la dernière version entraîne l'échec des builds. Au lieu de cela, vous devez spécifier une version exacte de Node.js18, telle que`21.5`, ou`0.1.2`.

Pour configurer les mises à jour de packages en direct

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez configurer les mises à jour des packages en direct.
3. Dans le volet de navigation, choisissez Hosting, Build settings.
4. Sur la page Paramètres de création, dans la section Paramètres de création d'image, choisissez Modifier.

5. Sur la page Modifier les paramètres de l'image de compilation, liste des mises à jour du package Live, choisissez Ajouter un nouveau.
6. Pour Package, sélectionnez la dépendance à remplacer.
7. Pour Version, conservez la dernière version par défaut ou entrez une version spécifique de la dépendance. Si vous utilisez la dernière version, la dépendance sera toujours mise à niveau vers la dernière version disponible.
8. Choisissez Enregistrer.

Configuration de l'instance de build pour une application Amplify

Amplify Hosting propose des tailles d'instance de build configurables qui vous permettent de fournir à l'instance de build de votre application les ressources de processeur, de mémoire et d'espace disque dont elle a besoin. Avant la sortie de cette fonctionnalité, Amplify fournissait une configuration d'instance de build de taille fixe de 8 GiB de mémoire et 4 v. CPUs

Amplify prend en charge trois types d'instances de build : StandardLarge, et. XLarge Si vous ne spécifiez aucun type d'instance, Amplify utilise l'instance par défautStandard. Vous pouvez configurer le type d'instance de build pour une application à l'aide de la console Amplify, du AWS CLI, ou du. SDKs

Le coût de chaque type d'instance de construction est calculé par minute de construction. Pour plus d'informations sur la tarification, consultez la page AWS Amplify Pricing (Tarification).

Le tableau suivant décrit les spécifications de calcul pour chaque type d'instance de build :

Type d'instance de construction	v CPUs	Mémoire	Espace disque
Standard	4 v CPUs	8 GiO	128 Go
Large	8 v CPUs	16 GiO	128 Go
XLarge	36 v CPUs	72 GiB	256 Go

Rubriques

- Comprendre les types d'instances de build
- Configuration du type d'instance de build dans la console Amplify

- Configuration de la mémoire de segment d'une application pour utiliser de grands types d'instances

Comprendre les types d'instances de build

Le paramètre du type d'instance de build est configuré au niveau de l'application et s'étend à toutes les branches de l'application. Les principaux détails suivants s'appliquent aux types d'instances de build :

- Le type d'instance de build que vous configurez pour une application s'applique automatiquement aux branches créées automatiquement et aux aperçus des pull requests.
- Le quota du service de tâches simultanées s'applique à tous les types d'instances de build de votre Compte AWS. Par exemple, si votre limite de tâches simultanées est de cinq, vous pouvez exécuter jusqu'à 5 builds pour tous les types d'instances de votre Compte AWS.
- Le coût de chaque type d'instance de construction est calculé par minute de construction. Le processus d'allocation des instances de build peut nécessiter des frais supplémentaires avant le début de votre build. Pour les instances plus importantes, en particulier XLarge, votre build peut rencontrer une latence avant le début de la compilation, en raison de ce temps de surcharge. Cependant, vous n'êtes facturé que pour le temps de construction réel, et non pour les frais généraux.

Vous pouvez configurer le type d'instance de build lorsque vous créez une nouvelle application ou vous pouvez mettre à jour le type d'instance sur une application existante. Pour obtenir des instructions sur la configuration de ce paramètre dans la console Amplify, consultez. Configuration du type d'instance de build dans la console Amplify Vous pouvez également mettre à jour ce paramètre à l'aide du SDKs. Pour plus d'informations, consultez le CreateApp, et UpdateApp APIs dans la référence de l'API Amplify.

Si votre compte contient déjà des applications créées avant la sortie de la fonctionnalité de type d'instance de génération personnalisable, elles utilisent le type d'Standardinstance par défaut. Lorsque vous mettez à jour le type d'instance de build pour une application existante, toutes les versions mises en file d'attente ou en cours avant votre mise à jour utiliseront le type d'instance de build configuré précédemment. Par exemple, si vous avez une application existante dont la `main` branche est déployée sur Amplify et que vous mettez à jour son type d'instance de build de Standard à Large, toutes les nouvelles versions que vous initierez depuis la `main` branche utiliseront le type d'instance de build Large. Toutefois, toutes les versions en cours au moment de la mise à jour du type d'instance de construction continueront de s'exécuter sur l'instance standard.

Configuration du type d'instance de build dans la console Amplify

Utilisez la procédure suivante pour configurer le type d'instance de build lorsque vous créez une nouvelle application Amplify.

Pour configurer le type d'instance de build pour une nouvelle application

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :
   a. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.
   b. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.
   c. Choisissez Suivant.
5. Sur la page des paramètres de l'application, ouvrez la section Paramètres avancés.
6. Pour le type d'instance de génération, choisissez le type d'instance souhaité dans la liste.
7. Si vous déployez une application basée sur l'exécution Node.js, configurez la taille de mémoire du segment de mémoire pour utiliser efficacement un type d'instance de grande taille. Vous pouvez le faire sur la page des paramètres de l'application en définissant une variable d'environnement ou en mettant à jour les paramètres de compilation. Pour de plus amples informations, veuillez consulter Configuration de la mémoire de segment d'une application pour utiliser de grands types d'instances.
   - Définir une variable d'environnement
     a. Dans la section Paramètres avancés, Variables d'environnement, choisissez Ajouter un nouveau.
     b. Pour Key, entrez**`NODE_OPTIONS`**.
     c. Pour le champ Valeur, saisissez `--max-old-space-size=`*`memory_size_in_mb`*. Remplacez *`memory_size_in_mb`* par la taille de mémoire de tas souhaitée en mégaoctets.
   - Mettre à jour les paramètres de compilation
     a. Dans la section Paramètres de génération, choisissez Modifier le fichier YML.

b. Ajoutez la commande suivante à la `preBuild` phase. Remplacez *`memory_size_in_mb`* par la taille de mémoire de tas souhaitée en mégaoctets.

```
export NODE_OPTIONS='--max-old-space-size=memory_size_in_mb'
```

c. Choisissez Enregistrer.

8. Choisissez Next.
9. Sur la page Révision, choisissez Enregistrer et déployer.

Utilisez la procédure suivante pour configurer le type d'instance de build pour une application Amplify existante.

Pour configurer le type d'instance de build pour une application existante

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez configurer le type d'instance de build.
3. Dans le volet de navigation, choisissez Hosting, puis choisissez Build settings.
4. Sur la page des paramètres de compilation, dans la section Paramètres avancés, choisissez Modifier.
5. Sur la page Modifier les paramètres, pour Créer le type d'instance, choisissez le type d'instance souhaité dans la liste.
6. Choisissez Enregistrer. Cette modification prendra effet lors du prochain déploiement de l'application.
7. (Facultatif) Pour déployer immédiatement l'application mise à jour, procédez comme suit :

   a. Dans le panneau de navigation, sélectionnez Présentation.

   b. Sur la page de présentation de votre application, choisissez la branche à redéployer.

   c. Sur la page Déploiement, choisissez un déploiement, tel que le déploiement le plus récent. Choisissez ensuite Redéployer cette version. Un nouveau déploiement va commencer.

   d. Une fois le déploiement terminé, les paramètres de compilation de l'application indiquent que la branche utilise le type d'instance de build mis à jour.

Configuration de la mémoire de segment d'une application pour utiliser de grands types d'instances

Si vous créez des applications gourmandes en mémoire, consultez cette section pour comprendre comment configurer votre application pour utiliser des types d'instances de grande taille. Les langages de programmation et les frameworks s'appuient souvent sur l'allocation de mémoire dynamique, également appelée mémoire en tas, pendant l'exécution pour gérer les besoins en mémoire des applications. La mémoire en tas est demandée par l'environnement d'exécution et allouée par le système d'exploitation hôte. Par défaut, les environnements d'exécution imposent une limite de taille de segment maximale disponible pour l'application. Cela signifie qu'aucune mémoire supplémentaire ne sera disponible pour l'application au-delà de la taille du segment de mémoire, même si le système d'exploitation ou le conteneur hôte dispose d'une plus grande quantité de mémoire disponible.

Par exemple, l'environnement d'exécution JavaScript Node.js v8 impose une limite de taille de segment par défaut qui dépend de plusieurs facteurs, notamment de la taille de la mémoire de l'hôte. Par conséquent, Standard les instances de Large build ont une taille de tas Node.js par défaut de 2 096 Mo et l'XLargeinstance a une taille de tas par défaut de 4 144 Mo. Par conséquent, la création d'une application nécessitant 6 000 Mo de mémoire en utilisant la taille de mémoire par défaut de Node.js sur n'importe quel type d'instance de build Amplify entraînera l'échec de la génération en raison d' out-of-memoryune erreur.

Pour contourner les limites de mémoire par défaut du fichier Node.js en matière de mémoire, vous pouvez effectuer l'une des opérations suivantes :

- Définissez la variable d'`NODE_OPTIONS`environnement de votre application Amplify sur la valeur. `--max-old-space-size=`*`memory_size_in_mb`* Pour`memory_size_in_mb`, spécifiez la taille de mémoire de segment de mémoire souhaitée en mégaoctets.

  Pour obtenir des instructions, veuillez consulter Définition de variables d'environnement.

- Ajoutez la commande suivante à la `preBuild` phase dans la spécification de construction de votre application Amplify.

  ```
  export NODE_OPTIONS='--max-old-space-size=memory_size_in_mb'
  ```

  Vous pouvez mettre à jour les spécifications de construction dans la console Amplify ou dans le `amplify.yml` fichier de votre application dans le référentiel de votre projet. Pour obtenir des

instructions, veuillez consulter Configuration des paramètres de compilation pour une application Amplify.

L'exemple de spécification de construction Amplify suivant définit une taille de mémoire de tas Node.js à 7 000 Mo pour créer une application frontale React :

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        # Set the heap size to 7000 MB
        - export NODE_OPTIONS='--max-old-space-size=7000'
        # To check the heap size memory limit in MB
        - node -e "console.log('Total available heap size (MB):',
 v8.getHeapStatistics().heap_size_limit / 1024 / 1024)"
        - npm ci --cache .npm --prefer-offline
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: build
    files:
      - '**/*'
  cache:
    paths:
      - .npm/**/*
```

Pour utiliser efficacement les types d'instances de grande taille, il est important de configurer une taille de mémoire de segment suffisante. La configuration d'un segment de mémoire réduit pour une application gourmande en mémoire risque d'entraîner un échec de compilation. Il est possible que les journaux de compilation de l'application n'indiquent pas directement une out-of-memory erreur, car le moteur d'exécution de l'application peut se bloquer de manière inattendue. La configuration d'une taille de segment aussi grande que la mémoire de l'hôte peut entraîner le remplacement ou l'arrêt d'autres processus par le système d'exploitation hôte, ce qui peut perturber votre processus de compilation. À titre de référence, Node.js recommande de définir une taille de mémoire maximale de 1 536 Mo sur une machine disposant d'environ 2 000 Mo de mémoire afin de laisser de la mémoire pour d'autres utilisations.

La taille de tas optimale dépend des besoins de votre application et de l'utilisation des ressources. Si vous rencontrez out-of-memory des erreurs, commencez par une taille de tas modérée, puis augmentez-la progressivement selon les besoins. À titre indicatif, nous recommandons de commencer avec 6 000 Mo pour un type d'Standardinstance, 12 000 Mo pour un type d'Largeinstance et 60 000 Mo pour un type d'XLargeinstance.

Création d'un webhook entrant pour démarrer une construction

Configurez un webhook entrant dans la console Amplify pour démarrer une compilation sans envoyer de code dans votre dépôt Git. Vous pouvez utiliser des webhooks avec des outils CMS headless (tels que Contentful ou GraphCMS) pour démarrer un build chaque fois que le contenu change, ou pour effectuer des builds quotidiens à l'aide de services tels que Zapier.

Pour créer un webhook entrant

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez créer un webhook.
3. Dans le volet de navigation, choisissez Hosting, puis Build settings.
4. Sur la page des paramètres de création, faites défiler la page jusqu'à la section Webhooks entrants et choisissez Create Webhook.
5. Dans la boîte de dialogue Créer un webhook, procédez comme suit :
    a. Pour le nom du webhook, entrez le nom du webhook.
    b. Pour Branch to build, sélectionnez la branche à compiler sur les requêtes webhook entrantes.
    c. Choisissez Create webhook.
6. Dans la section Webhooks entrants, effectuez l'une des opérations suivantes :
    - Copiez l'URL du webhook et fournissez-la à un outil CMS headless ou à un autre service pour lancer les builds.
    - Exécutez la commande curl dans une fenêtre de terminal pour démarrer une nouvelle version.

Configuration des notifications par e-mail pour les builds

Vous pouvez configurer des notifications par e-mail pour une AWS Amplify application afin d'avertir les parties prenantes ou les membres de l'équipe en cas de réussite ou d'échec d'un build. Amplify Hosting crée une rubrique Amazon Simple Notification Service (SNS) dans votre compte et l'utilise pour configurer les notifications par e-mail. Les notifications peuvent être configurées pour s'appliquer à toutes les branches ou à des branches spécifiques d'une application Amplify.

Configuration des notifications par e-mail

Utilisez les procédures suivantes pour configurer les notifications par e-mail pour toutes les branches ou pour des branches spécifiques d'une application Amplify.

Pour configurer les notifications par e-mail pour une application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez configurer les notifications par e-mail.
3. Dans le volet de navigation, choisissez Hosting, Build notifications. Sur la page Créer des notifications, choisissez Gérer les notifications.
4. Sur la page Gérer les notifications, choisissez Ajouter un nouveau.
5. Effectuez l'une des actions suivantes :
   - Pour envoyer des notifications pour une seule succursale, dans le champ E-mail, entrez l'adresse e-mail à laquelle envoyer les notifications. Pour Branche, sélectionnez le nom de la succursale pour laquelle vous souhaitez envoyer des notifications.
   - Pour envoyer des notifications à toutes les succursales connectées, dans le champ E-mail, entrez l'adresse e-mail à laquelle envoyer les notifications. Pour Branche, sélectionnez Toutes les branches.
6. Choisissez Enregistrer.

Connexion d'un domaine personnalisé

Vous pouvez connecter une application que vous avez déployée avec Amplify Hosting à un domaine personnalisé. Lorsque vous utilisez Amplify pour déployer votre application Web, Amplify l'héberge pour vous sur le `amplifyapp.com` domaine par défaut avec une URL telle que. `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com` Lorsque vous connectez votre application à un domaine personnalisé, les utilisateurs voient que votre application est hébergée sur une URL personnalisée, telle que`https://www.example.com`.

Vous pouvez acheter un domaine personnalisé auprès d'un bureau d'enregistrement de domaines accrédité tel qu'Amazon Route 53 ou GoDaddy. Route 53 est le service Web DNS (Domain Name System) d'Amazon. Pour plus d'informations sur l'utilisation de Route 53, consultez Qu'est-ce qu'Amazon Route 53 ? Pour obtenir la liste des bureaux d'enregistrement de domaines accrédités tiers, consultez le répertoire des bureaux d'enregistrement accrédités sur le site Web de l'ICANN.

Lorsque vous configurez votre domaine personnalisé, vous pouvez utiliser le certificat géré par défaut fourni par Amplify pour vous ou vous pouvez utiliser votre propre certificat personnalisé. Vous pouvez modifier le certificat utilisé pour le domaine à tout moment. Pour obtenir des informations détaillées sur la gestion des certificats, consultezUtilisation de SSL/TLS certificats.

Avant de procéder à la configuration d'un domaine personnalisé, vérifiez que vous remplissez les conditions préalables suivantes.

- Vous possédez un nom de domaine enregistré.
- Vous avez un certificat émis par ou importé dans AWS Certificate Manager.
- Vous avez déployé votre application sur Amplify Hosting.

  Pour plus d'informations sur la réalisation de cette étape, consultezCommencer à déployer une application sur Amplify Hosting.
- Vous avez une connaissance de base des domaines et de la terminologie du DNS.

  Pour plus d'informations sur les domaines et le DNS, consultezComprendre la terminologie et les concepts du DNS.

> **Warning**
>
> Lorsque vous DomainAssociation lancez une demande pour une application Amplify avec un domaine qui est déjà ou a été précédemment associé à différentes applications Amplify dans d'autres comptes AWS de la même région, cela est considéré comme une association de domaines entre comptes. Les demandes d'association de domaines entre comptes nécessitent une vérification manuelle. Si vous souhaitez créer une association de domaines entre comptes, contactez le support AWS pour obtenir de l'aide.

Rubriques

Comprendre la terminologie et les concepts du DNS

Si vous ne connaissez pas les termes et concepts associés au système de noms de domaine (DNS), les rubriques suivantes peuvent vous aider à comprendre les procédures d'ajout de domaines personnalisés.

Terminologie DNS

Vous trouverez ci-dessous une liste de termes communs au DNS. Ils peuvent vous aider à comprendre les procédures d'ajout de domaines personnalisés.

CNAME

Un nom d'enregistrement canonique (CNAME) est un type d'enregistrement DNS qui masque le domaine d'un ensemble de pages Web et les fait apparaître comme si elles se trouvaient ailleurs. Un CNAME pointe un sous-domaine vers un nom de domaine complet (FQDN). Par exemple, vous pouvez créer un nouvel enregistrement CNAME pour mapper le sous-domaine www.example.com, où www est le sous-domaine, au domaine FQDN branch-name.d1m7bkiki6tdw1.cloudfront.net attribué à votre application dans la console Amplify.

ANAME

Un enregistrement ANAME est similaire à un enregistrement CNAME, mais au niveau de la racine. Un ANAME pointe la racine de votre domaine vers un FQDN. Ce FQDN pointe vers une adresse IP.

Serveur de noms

Un serveur de noms est un serveur sur Internet spécialisé dans le traitement des requêtes concernant l'emplacement des différents services d'un nom de domaine. Si vous configurez votre domaine dans Amazon Route 53, une liste de serveurs de noms est déjà attribuée à votre domaine.

Enregistrement NS

Un enregistrement NS pointe vers des serveurs de noms qui consultent les détails de votre domaine.

Vérification DNS

Un système de noms de domaine (DNS) est comme un annuaire téléphonique qui traduit des noms de domaine lisibles par l'homme en adresses IP adaptées aux ordinateurs. Lorsque vous tapez **`https://google.com`** dans un navigateur, une opération de recherche est effectuée dans le fournisseur DNS pour trouver l'adresse IP du serveur qui héberge le site Web.

Les fournisseurs DNS contiennent des enregistrements de domaines et leurs adresses IP correspondantes. Les enregistrements DNS les plus couramment utilisés sont les enregistrements CNAME, ANAME et NS.

Amplify utilise un enregistrement CNAME pour vérifier que vous êtes bien le propriétaire de votre domaine personnalisé. Si vous hébergez votre domaine avec Route 53, la vérification est effectuée automatiquement en votre nom. Toutefois, si vous hébergez votre domaine chez un fournisseur tiers

tel que GoDaddy, vous devez mettre à jour manuellement les paramètres DNS de votre domaine et ajouter un nouvel enregistrement CNAME fourni par Amplify.

Processus d'activation de domaine personnalisé

> **Warning**
>
> Lorsque vous DomainAssociation lancez une demande pour une application Amplify avec un domaine qui est déjà ou a été précédemment associé à différentes applications Amplify dans d'autres comptes AWS de la même région, cela est considéré comme une association de domaines entre comptes. Les demandes d'association de domaines entre comptes nécessitent une vérification manuelle. Si vous souhaitez créer une association de domaines entre comptes, contactez le support AWS pour obtenir de l'aide.

Lorsque vous connectez votre application Amplify à un domaine personnalisé dans la console Amplify, Amplify doit effectuer plusieurs étapes avant de pouvoir afficher votre application à l'aide de votre domaine personnalisé. La liste suivante décrit chaque étape du processus de configuration et d'activation du domaine.

Création de SSL/TLS

Si vous utilisez un certificat géré, AWS Amplify émet un certificat SSL/TLS pour configurer un domaine personnalisé sécurisé.

Configuration et vérification SSL/TLS

Avant d'émettre un certificat géré, Amplify vérifie que vous êtes le propriétaire du domaine. Pour les domaines gérés par Amazon Route 53, Amplify met automatiquement à jour l'enregistrement de vérification DNS. Pour les domaines gérés en dehors de Route 53, vous devez ajouter manuellement l'enregistrement de vérification DNS fourni dans la console Amplify dans votre domaine auprès d'un fournisseur DNS tiers.

Si vous utilisez un certificat personnalisé, vous êtes responsable de la validation de la propriété du domaine.

Activation du domaine

Le domaine a été vérifié avec succès. Pour les domaines gérés en dehors de Route 53, vous devez ajouter manuellement les enregistrements CNAME fournis dans la console Amplify dans votre domaine auprès d'un fournisseur DNS tiers.

Utilisation de SSL/TLS certificats

Un SSL/TLS certificat est un document numérique qui permet aux navigateurs Web d'identifier et d'établir des connexions réseau cryptées avec des sites Web à l'aide du SSL/TLS protocole sécurisé. Lorsque vous configurez votre domaine personnalisé, vous pouvez utiliser le certificat géré par défaut fourni par Amplify pour vous ou vous pouvez utiliser votre propre certificat personnalisé.

Avec un certificat géré, Amplify émet un SSL/TLS certificat pour tous les domaines connectés à votre application afin que tout le trafic soit sécurisé via HTTPS/2. Le certificat par défaut généré par AWS Certificate Manager (ACM) est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify.

> **Warning**
>
> Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé dans les paramètres DNS de votre fournisseur de domaine. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

Pour utiliser un certificat personnalisé, vous devez d'abord obtenir un certificat auprès de l'autorité de certification tierce de votre choix. Amplify Hosting prend en charge deux types de certificats : RSA (Rivest-Shamir-Adleman) et ECDSA (Elliptic Curve Digital Signature Algorithm). Chaque type de certificat doit être conforme aux exigences suivantes.

Certificats RSA

- Amplify Hosting prend en charge les clés RSA 1024 bits, 2048 bits, 3072 bits et 4096 bits.
- AWS Certificate Manager (ACM) émet des certificats RSA avec des clés allant jusqu'à 2 048 bits.
- Pour utiliser un certificat RSA 3072 bits ou 4096 bits, procurez-vous le certificat en externe et importez-le dans ACM. Il pourra ensuite être utilisé avec Amplify Hosting.

Certificats ECDSA

- Amplify Hosting prend en charge les clés 256 bits.
- Utilisez la courbe elliptique prime256v1 pour obtenir un certificat ECDSA pour Amplify Hosting.

Après avoir obtenu un certificat, importez-le dans AWS Certificate Manager. ACM est un service qui vous permet d'approvisionner, de gérer et de déployer facilement des SSL/TLS certificats publics et privés à utiliser avec Services AWS vos ressources internes connectées. Assurez-vous de demander ou d'importer le certificat dans la région USA Est (Virginie du Nord) (us-east-1).

Assurez-vous que votre certificat personnalisé couvre tous les sous-domaines que vous prévoyez d'ajouter. Vous pouvez utiliser un caractère générique au début de votre nom de domaine pour couvrir plusieurs sous-domaines. Par exemple, si votre domaine l'est`example.com`, vous pouvez inclure le domaine `*.example.com` générique. Cela couvrira les sous-domaines tels que `product.example.com` et. `api.example.com`

Une fois que votre certificat personnalisé sera disponible dans ACM, vous pourrez le sélectionner lors du processus de configuration du domaine. Pour obtenir des instructions sur l'importation de certificats dans AWS Certificate Manager, consultez la section Importation de certificats AWS Certificate Manager dans le guide de AWS Certificate Manager l'utilisateur.

Si vous renouvelez ou réimportez votre certificat personnalisé dans ACM, Amplify actualise les données du certificat associées à votre domaine personnalisé. Dans le cas de certificats importés, ACM ne gère pas les renouvellements automatiquement. Vous êtes responsable du renouvellement de vos certificats personnalisés et de leur réimportation.

Vous pouvez modifier le certificat utilisé pour un domaine à tout moment. Par exemple, vous pouvez passer du certificat géré par défaut à un certificat personnalisé ou passer d'un certificat personnalisé à un certificat géré. En outre, vous pouvez remplacer le certificat personnalisé utilisé par un autre certificat personnalisé. Pour obtenir des instructions sur la mise à jour des certificats, voir Mettre à jour le SSL/TLS certificat d'un domaine.

Ajouter un domaine personnalisé géré par Amazon Route 53

Amazon Route 53 est un service DNS hautement disponible et évolutif. Pour plus d'informations, consultez Amazon Route 53 dans le manuel du développeur Amazon Route 53. Si vous possédez déjà un domaine Route 53, suivez les instructions suivantes pour connecter votre domaine personnalisé à votre application Amplify.

Pour ajouter un domaine personnalisé géré par Route 53

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application que vous souhaitez connecter à un domaine personnalisé.

3. Dans le volet de navigation, choisissez Hosting, Custom domains.
4. Sur la page Domaines personnalisés, choisissez Ajouter un domaine.
5. Entrez le nom de votre domaine racine. Par exemple, si le nom de votre domaine est https://example.com, entrez**`example.com`**.

   Lorsque vous commencez à taper, tous les domaines racines que vous gérez déjà dans Route 53 apparaissent dans la liste. Vous pouvez choisir le domaine que vous souhaitez utiliser dans la liste. Si vous ne possédez pas encore le domaine et qu'il est disponible, vous pouvez l'acheter sur Amazon Route 53.
6. Après avoir saisi votre nom de domaine, choisissez Configurer le domaine.
7. Par défaut, Amplify crée automatiquement deux entrées de sous-domaine pour votre domaine. Par exemple, si votre nom de domaine est exemple.com, vous verrez les sous-domaines https://www.example.comet https://example.comune redirection sera configurée depuis le domaine racine vers le sous-domaine www.

   (Facultatif) Vous pouvez modifier la configuration par défaut si vous souhaitez uniquement ajouter des sous-domaines. Pour modifier la configuration par défaut, choisissez Réécritures et redirections dans le volet de navigation, puis configurez votre domaine.
8. Choisissez le SSL/TLS certificat à utiliser. Vous pouvez soit utiliser le certificat géré par défaut fourni par Amplify pour vous, soit un certificat tiers personnalisé dans lequel vous avez importé. AWS Certificate Manager
   - Utilisez le certificat géré par Amplify par défaut.
     - Choisissez le certificat géré Amplify.
   - Utilisez un certificat tiers personnalisé.
     a. Choisissez un certificat SSL personnalisé.
     b. Sélectionnez le certificat à utiliser dans la liste.
9. Choisissez Ajouter un domaine.

> **Note**
>
> La propagation du DNS et l'émission du certificat peuvent prendre jusqu'à 24 heures. Pour obtenir de l'aide sur la résolution des erreurs qui se produisent, consultezRésolution des problèmes liés aux domaines personnalisés.

Ajouter un domaine personnalisé géré par un fournisseur DNS tiers

Si vous n'utilisez pas Amazon Route 53 pour gérer votre domaine, vous pouvez ajouter un domaine personnalisé géré par un fournisseur DNS tiers à votre application déployée avec Amplify.

Si vous l'utilisez GoDaddy, consultez the section called "Mise à jour des enregistrements DNS pour un domaine géré par GoDaddy" les instructions spécifiques à ce fournisseur.

Pour ajouter un domaine personnalisé géré par un fournisseur DNS tiers

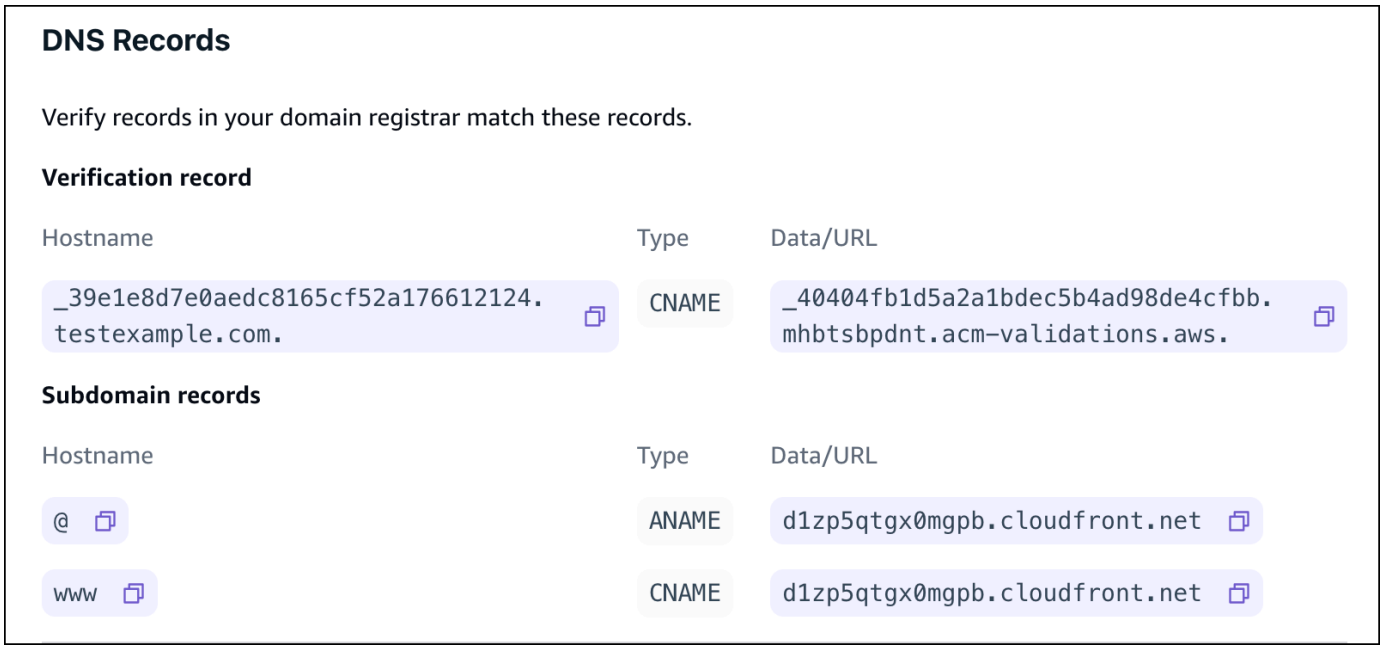
1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez ajouter un domaine personnalisé.
3. Dans le volet de navigation, choisissez Hosting, Custom domains.
4. Sur la page Domaines personnalisés, choisissez Ajouter un domaine.
5. Entrez le nom de votre domaine racine. Par exemple, si le nom de votre domaine est https://example.com, entrez**`example.com`**.
6. Amplify détecte que vous n'utilisez pas de domaine Route 53 et vous donne la possibilité de créer une zone hébergée dans Route 53.
   - Pour créer une zone hébergée dans Route 53
     a. Choisissez Créer une zone hébergée sur Route 53.
     b. Choisissez Configurer le domaine.
     c. Les serveurs de noms de zone hébergés sont affichés dans la console. Accédez au site Web de votre fournisseur DNS et ajoutez les serveurs de noms à vos paramètres DNS.
     d. Sélectionnez J'ai ajouté les serveurs de noms ci-dessus à mon registre de domaines.
     e. Passez à l'étape 7.
   - Pour poursuivre la configuration manuelle
     a. Choisissez Configuration manuelle
     b. Choisissez Configurer le domaine.
     c. Passez à l'étape 7.
7. Par défaut, Amplify crée automatiquement deux entrées de sous-domaine pour votre domaine. Par exemple, si votre nom de domaine est exemple.com, vous verrez les sous-domaines https://www.example.comet https://example.comune redirection sera configurée depuis le domaine racine vers le sous-domaine www.

(Facultatif) Vous pouvez modifier la configuration par défaut si vous souhaitez uniquement ajouter des sous-domaines. Pour modifier la configuration par défaut, choisissez Réécritures et redirections dans le volet de navigation et configurez votre domaine.

8. Choisissez le SSL/TLS certificat à utiliser. Vous pouvez soit utiliser le certificat géré par défaut fourni par Amplify pour vous, soit un certificat tiers personnalisé dans lequel vous avez importé. AWS Certificate Manager
   - Utilisez le certificat géré par Amplify par défaut.
     - Choisissez le certificat géré Amplify.
   - Utilisez un certificat tiers personnalisé.
     a. Choisissez un certificat SSL personnalisé.
     b. Sélectionnez le certificat à utiliser dans la liste.
9. Choisissez Ajouter un domaine.
10. Si vous avez choisi Créer une zone hébergée sur Route 53 à l'étape 6, passez à l'étape 15.

    Si vous avez choisi Configuration manuelle, à l'étape 6, vous devez mettre à jour vos enregistrements DNS auprès de votre fournisseur de domaine tiers.

    Dans le menu Actions, choisissez Afficher les enregistrements DNS. La capture d'écran suivante montre les enregistrements DNS affichés dans la console.



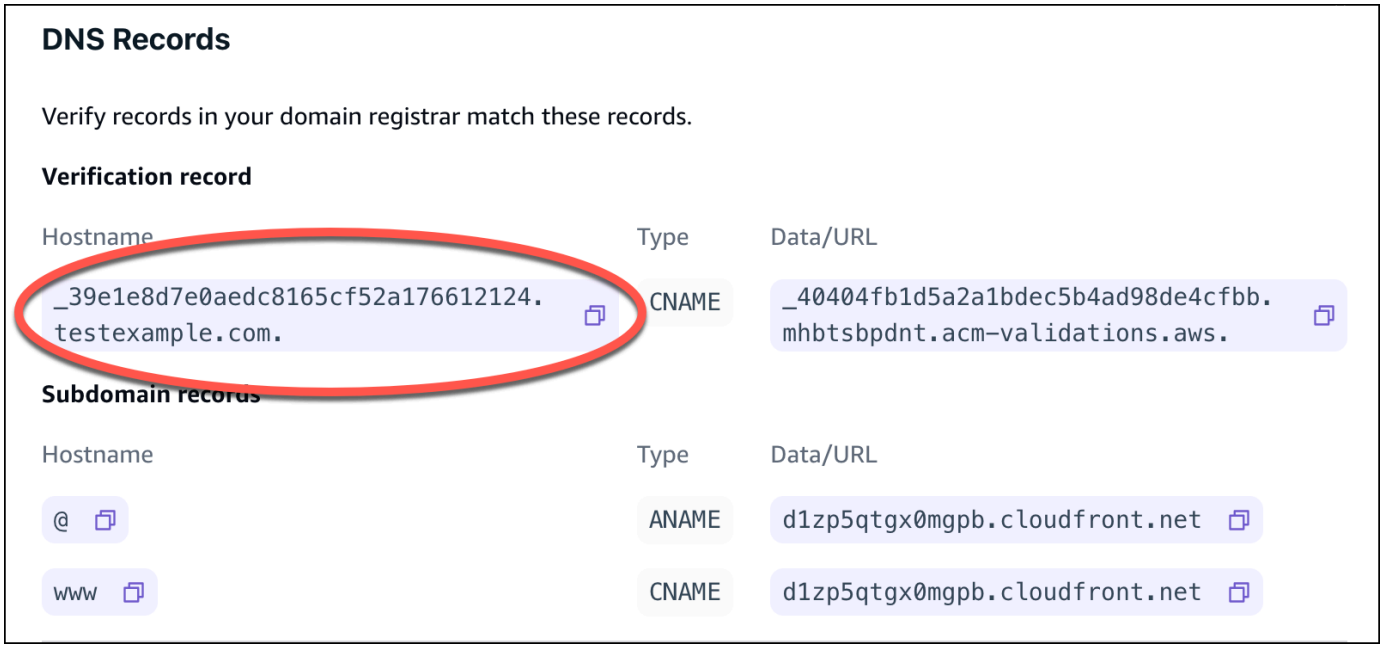
11. Effectuez l'une des actions suivantes :

- Si vous utilisez GoDaddy, rendez-vous surMise à jour des enregistrements DNS pour un domaine géré par GoDaddy.
- Si vous utilisez un autre fournisseur DNS tiers, passez à l'étape suivante de cette procédure.

12. Accédez au site Web de votre fournisseur DNS, connectez-vous à votre compte et recherchez les paramètres de gestion DNS de votre domaine. Vous allez configurer deux enregistrements CNAME.
13. Configurez le premier enregistrement CNAME pour qu'il pointe votre sous-domaine vers le serveur de AWS validation.

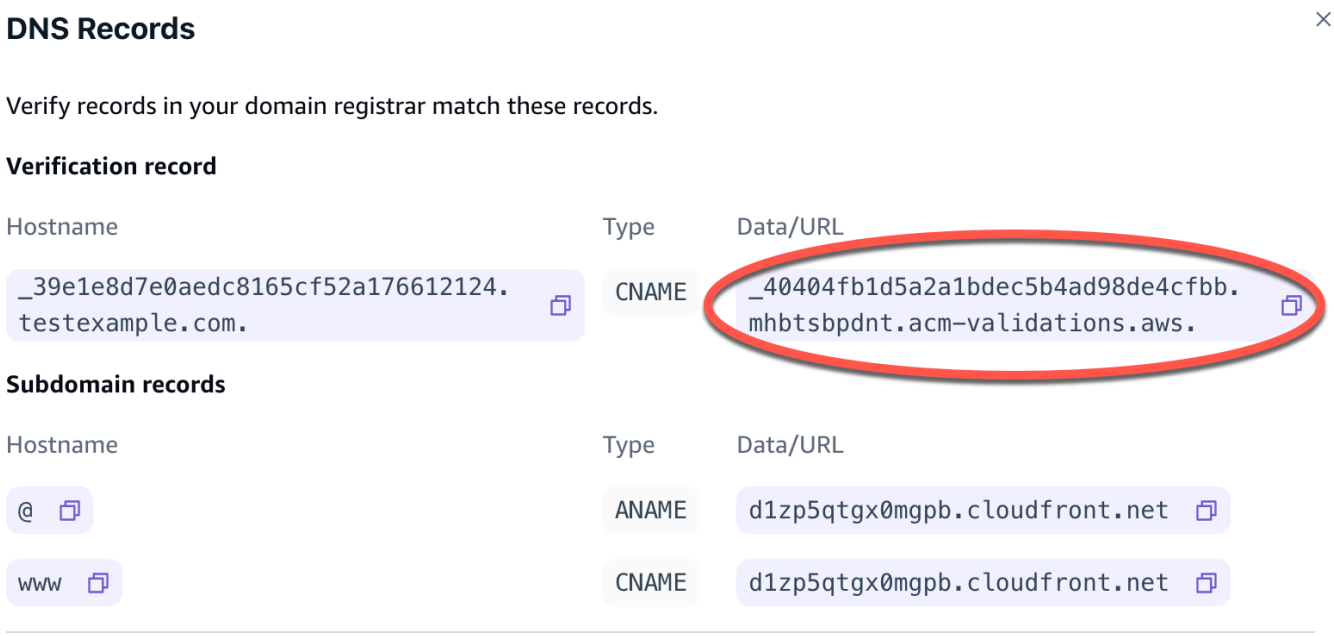
    Si la console Amplify affiche un enregistrement DNS permettant de vérifier la propriété de votre sous-domaine, tel que _c3e2d7eaf1e656b73f46cd6980fdc0e.example.com, entrez uniquement le nom du sous-domaine de l'enregistrement CNAME. **`_c3e2d7eaf1e656b73f46cd6980fdc0e`**

    La capture d'écran suivante montre l'emplacement de l'enregistrement de vérification à utiliser.



    Si la console Amplify affiche un enregistrement du serveur de validation ACM tel que _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws, entrez la valeur de l'enregistrement CNAME. **`_cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws`**

    La capture d'écran suivante montre l'emplacement de l'enregistrement de vérification ACM à utiliser.



Amplify utilise ces informations pour vérifier la propriété de votre domaine et générer un SSL/TLS certificat pour votre domaine. Une fois qu'Amplify aura validé la propriété de votre domaine, tout le trafic sera diffusé via HTTPS/2.

> **Note**
>
> Le certificat Amplify par défaut généré par AWS Certificate Manager (ACM) est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify. Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

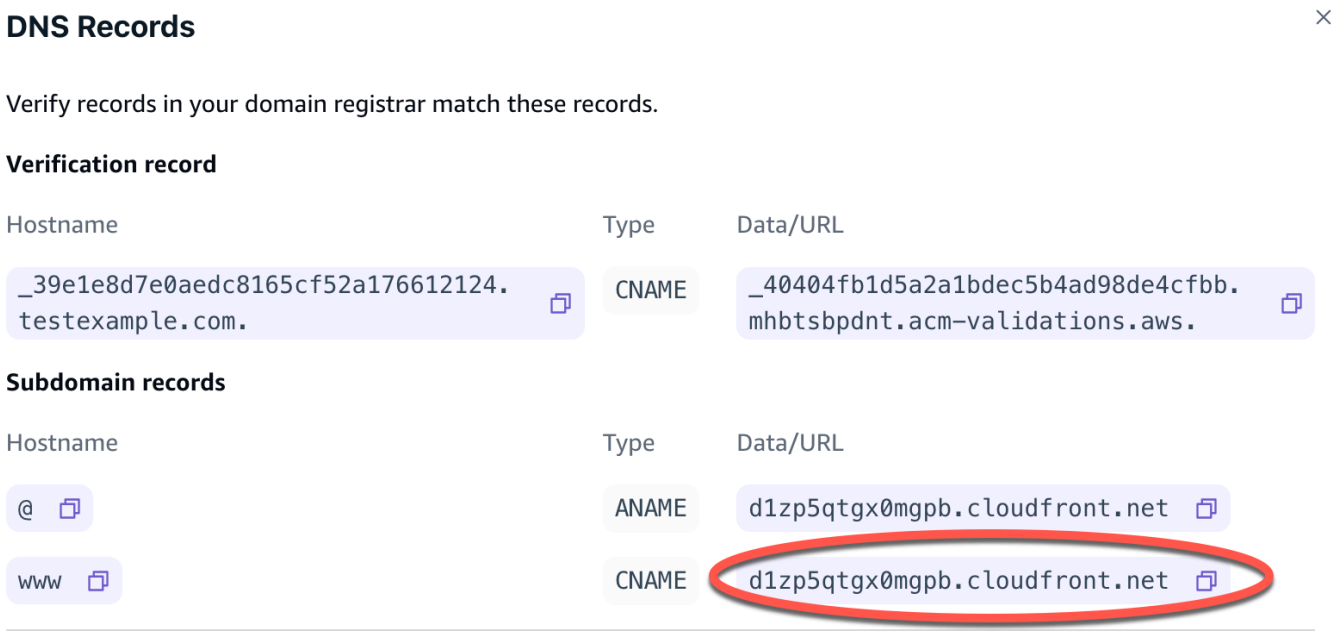
> **Important**
>
> Il est important que vous exécutiez cette étape peu après avoir ajouté votre domaine personnalisé dans la console Amplify. L' AWS Certificate Manager (ACM) commence immédiatement à essayer de vérifier la propriété. Au fil du temps, les contrôles deviennent moins fréquents. Si vous ajoutez ou mettez à jour vos enregistrements CNAME quelques heures après avoir créé votre application, celle-ci peut rester bloquée dans l'état d'attente de vérification.

14. Configurez un deuxième enregistrement CNAME pour faire pointer vos sous-domaines vers le domaine Amplify. Par exemple, si votre sous-domaine est www.exemple.com, entrez www comme nom de sous-domaine.

    Si la console Amplify affiche le domaine de votre application sous la forme d111111abcdef8.cloudfront.net, entrez le domaine Amplify.
    `d111111abcdef8.cloudfront.net`

    Si vous avez du trafic de production, nous vous recommandons de mettre à jour cet enregistrement CNAME une fois que le statut de votre domaine sera indiqué comme DISPONIBLE dans la console Amplify.

    La capture d'écran suivante montre l'emplacement de l'enregistrement de nom de domaine à utiliser.



15. Configurez l' ANAME/ALIAS enregistrement pour qu'il pointe vers le domaine racine de votre application (par exemple https://example.com). Un enregistrement ANAME pointe la racine de votre domaine vers un nom d'hôte. Si vous avez du trafic de production, nous vous recommandons de mettre à jour votre enregistrement ANAME une fois que le statut de votre domaine sera indiqué comme DISPONIBLE dans la console. Pour les fournisseurs de DNS qui n'ont pas de ANAME/ALIAS support, nous recommandons vivement de migrer votre DNS vers Route 53. Pour plus d'informations, consultez Configuration d'Amazon Route 53 en tant que service DNS.

> **Note**
>
> La vérification de la propriété du domaine et la propagation du DNS pour les domaines tiers peuvent prendre jusqu'à 48 heures. Pour obtenir de l'aide pour résoudre les erreurs qui se produisent, consultez la section Résolution des problèmes liés aux domaines personnalisés.

Mise à jour des enregistrements DNS pour un domaine géré par GoDaddy

S'il s' GoDaddy agit de votre fournisseur DNS, suivez les instructions suivantes pour mettre à jour vos enregistrements DNS dans l' GoDaddy interface utilisateur afin de terminer la connexion de votre application Amplify à votre GoDaddy domaine.

Pour ajouter un domaine personnalisé géré par GoDaddy

1. Avant de mettre à jour vos enregistrements DNS avec GoDaddy, effectuez les étapes 1 à 9 de la procéduresection called "Ajouter un domaine personnalisé géré par un fournisseur DNS tiers".
2. Connectez-vous à votre GoDaddy compte.
3. Dans votre liste de domaines, recherchez le domaine à ajouter et choisissez Gérer le DNS.
4. Sur la page DNS, GoDaddy affiche la liste des enregistrements de votre domaine dans la section Enregistrements DNS. Vous devez ajouter deux nouveaux enregistrements CNAME.
5. Créez le premier enregistrement CNAME pour faire pointer vos sous-domaines vers le domaine Amplify.

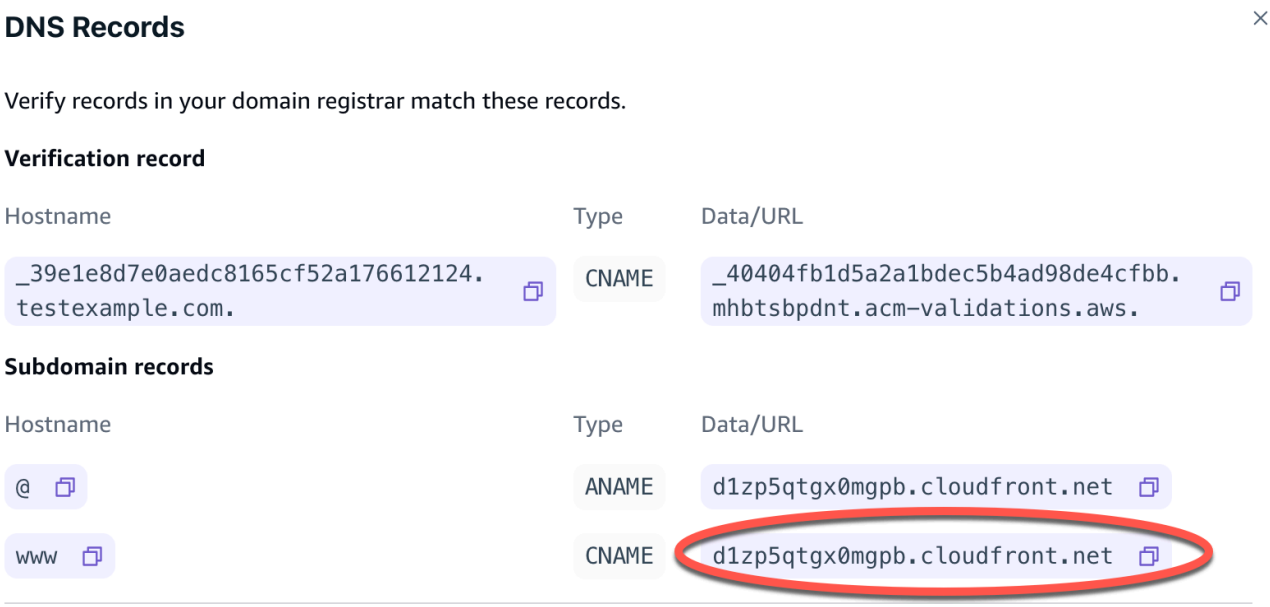
   a. Dans la section Enregistrements DNS, choisissez Ajouter un nouvel enregistrement.

   b. Pour Type, choisissez CNAME.

   c. Dans Nom, entrez uniquement le sous-domaine. Par exemple, si votre sous-domaine est www.exemple.com, entrez www pour Nom.

   d. Pour Value, examinez vos enregistrements DNS dans la console Amplify, puis entrez la valeur. Si la console Amplify affiche le domaine de votre application sous la forme d111111abcdef8.cloudfront.net, entrez Value. **`d111111abcdef8.cloudfront.net`**

      La capture d'écran suivante montre l'emplacement de l'enregistrement de nom de domaine à utiliser.



e. Choisissez Enregistrer.

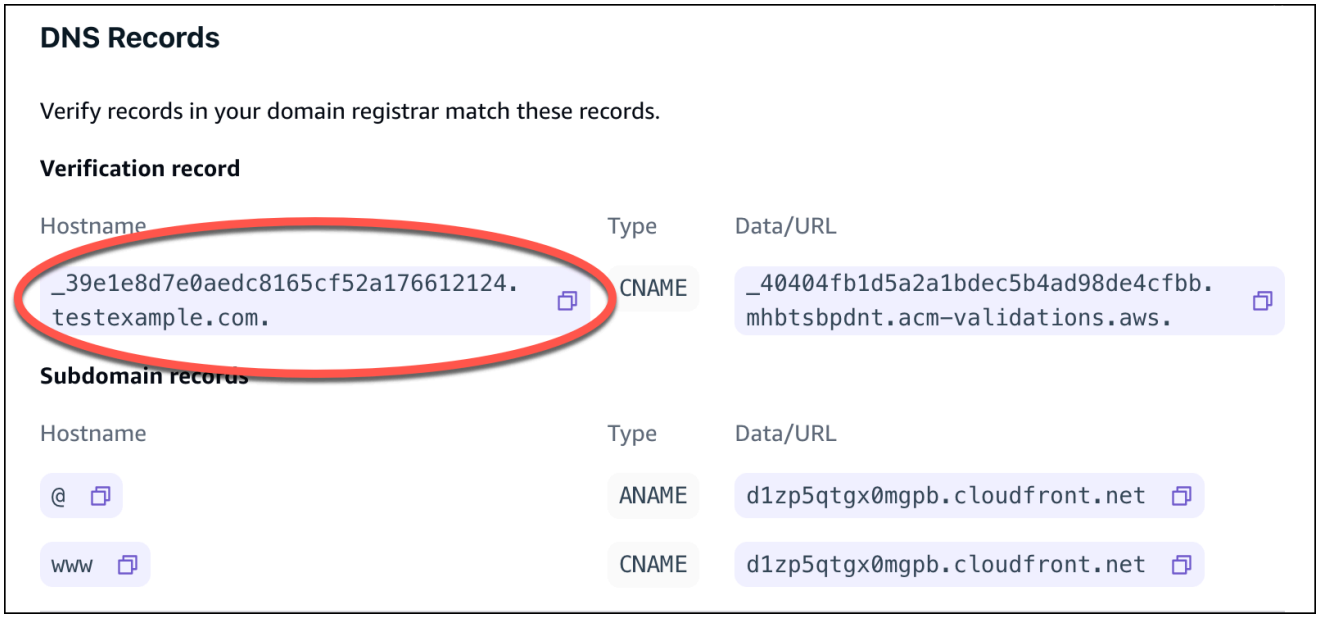
6. Créez le deuxième enregistrement CNAME qui pointe vers le serveur de validation AWS Certificate Manager (ACM). Un seul ACM validé génère un SSL/TLS certificat pour votre domaine.

   a. Pour Type, choisissez CNAME.

   b. Dans Nom, entrez le sous-domaine.

      Par exemple, si l'enregistrement DNS de la console Amplify permettant de vérifier la propriété de votre sous-domaine est _c3e2d7eaf1e656b73f46cd6980fdc0e.example.com, entrez uniquement le nom. **`_c3e2d7eaf1e656b73f46cd6980fdc0e`**

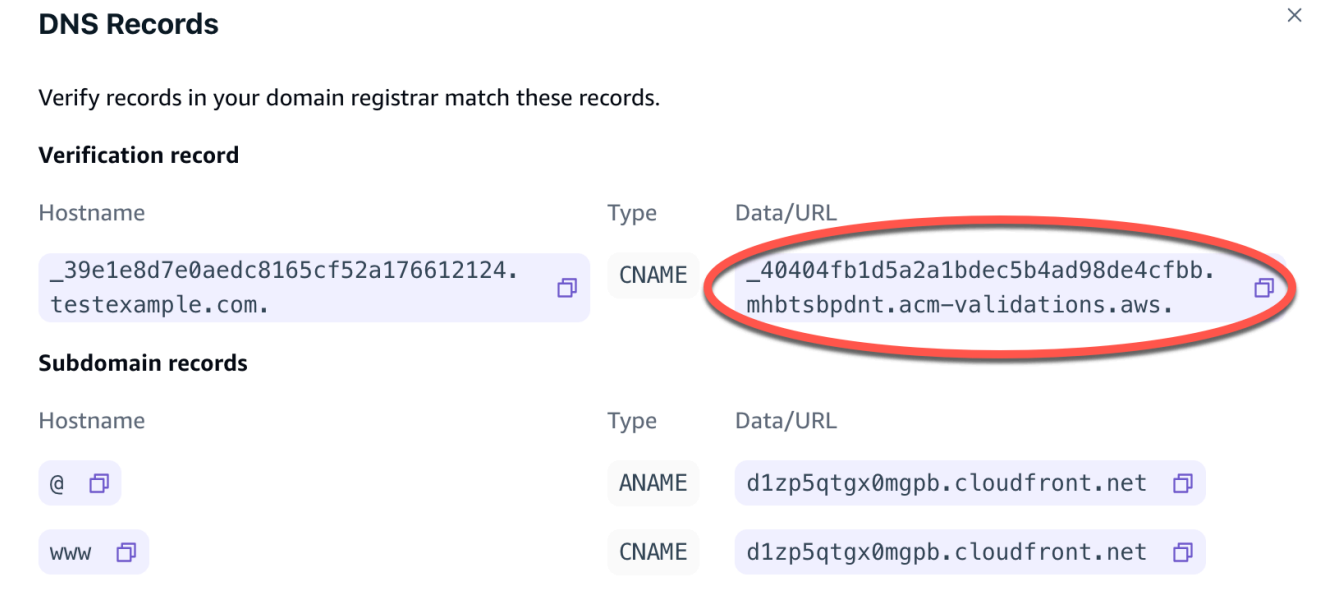
      La capture d'écran suivante montre l'emplacement de l'enregistrement de vérification à utiliser.



c. Pour Value, entrez le certificat de validation ACM.

Par exemple, si le serveur de validation est _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws, entrez _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws pour Value.

La capture d'écran suivante montre l'emplacement de l'enregistrement de vérification ACM à utiliser.



d. Choisissez Enregistrer.

> **Note**
>
> Le certificat Amplify par défaut généré par AWS Certificate Manager (ACM) est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify. Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

7. Cette étape n'est pas obligatoire pour les sous-domaines. GoDaddy ne prend pas en ANAME/ALIAS records. For DNS providers that do not have ANAME/ALIAS charge le support, nous vous recommandons vivement de migrer votre DNS vers Amazon Route 53. Pour plus d'informations, consultez Configuration d'Amazon Route 53 en tant que service DNS.

   Si vous souhaitez rester GoDaddy votre fournisseur et mettre à jour le domaine racine, ajoutez le transfert et configurez un transfert de domaine :

   a. Sur la page DNS, recherchez le menu en haut de la page et choisissez Forwarding.
   b. Dans la section Domaine, choisissez Ajouter un transfert.
   c. Choisissez http ://, puis entrez le nom du sous-domaine vers lequel vous souhaitez transférer (par exemple, www.example.com) pour l'URL de destination.
   d. Pour le type de transfert, choisissez Temporary (302).
   e. Choisissez Enregistrer.

Mettre à jour le SSL/TLS certificat d'un domaine

Vous pouvez modifier le SSL/TLS certificat utilisé pour un domaine à tout moment. Par exemple, vous pouvez passer d'un certificat géré à un certificat personnalisé. Cela est utile si vous souhaitez gérer le certificat et ses notifications d'expiration. Vous pouvez également modifier le certificat personnalisé utilisé pour le domaine. Les modifications apportées au certificat SSL n'entraîneront aucune interruption de service pour votre domaine actif. Pour plus d'informations sur les certificats, consultez la section Utilisation de certificats SSL/TLS.

Suivez la procédure ci-dessous pour mettre à jour le type de certificat ou le certificat personnalisé utilisé pour un domaine.

Pour mettre à jour le certificat d'un domaine

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application que vous souhaitez mettre à jour.
3. Dans le volet de navigation, choisissez Hosting, Custom domains.
4. Sur la page Domaines personnalisés, choisissez Configuration du domaine.
5. Sur la page de détails de votre domaine, recherchez la section Certificat SSL personnalisé. La procédure de mise à jour de votre certificat varie en fonction du type de modification que vous souhaitez apporter.
   - Pour passer d'un certificat personnalisé au certificat géré par Amplify par défaut
     - Choisissez le certificat géré Amplify.
   - Pour passer d'un certificat géré à un certificat personnalisé
     a. Choisissez un certificat SSL personnalisé.
     b. Sélectionnez le certificat à utiliser dans la liste.
   - Pour remplacer un certificat personnalisé par un autre certificat personnalisé
     - Pour le certificat SSL personnalisé, sélectionnez le nouveau certificat à utiliser dans la liste.
6. Choisissez Enregistrer. Les détails du statut du domaine indiqueront qu'Amplify a lancé le processus de création SSL pour un certificat géré ou le processus de configuration pour un certificat personnalisé.

Gestion des sous-domaines

Un sous-domaine est la partie de votre URL qui apparaît avant votre nom de domaine. Par exemple, www est le sous-domaine de www.amazon.com et aws est le sous-domaine de aws.amazon.com. Si vous possédez déjà un site Web de production, vous souhaiterez peut-être connecter uniquement un sous-domaine. Les sous-domaines peuvent également être à plusieurs niveaux, par exemple beta.alpha.example.com possède le sous-domaine à plusieurs niveaux beta.alpha.

Pour ajouter un sous-domaine uniquement

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez ajouter un sous-domaine.

3. Dans le volet de navigation, choisissez Hosting, puis Custom domains.
4. Sur la page Domaines personnalisés, choisissez Ajouter un domaine.
5. Entrez le nom de votre domaine racine, puis choisissez Configurer le domaine. Par exemple, si le nom de votre domaine est https://example.com, entrez exemple.com.
6. Choisissez Exclure la racine et modifiez le nom du sous-domaine. Par exemple, si le domaine est exemple.com, vous pouvez le modifier pour ajouter uniquement le sous-domaine alpha.
7. Choisissez Ajouter un domaine.

Pour ajouter un sous-domaine à plusieurs niveaux

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez ajouter un sous-domaine multiniveaux.
3. Dans le volet de navigation, choisissez Hosting, puis Custom domains.
4. Sur la page Domaines personnalisés, choisissez Ajouter un domaine.
5. Entrez le nom d'un domaine avec un sous-domaine, choisissez Exclure la racine et modifiez le sous-domaine pour ajouter un nouveau niveau.

   Par exemple, si vous avez un domaine appelé alpha.exemple.com et que vous souhaitez créer un sous-domaine à plusieurs niveaux beta.alpha.example.com, vous devez saisir beta comme valeur du sous-domaine.
6. Choisissez Ajouter un domaine.

Pour ajouter ou modifier un sous-domaine

Après avoir ajouté un domaine personnalisé à une application, vous pouvez modifier un sous-domaine existant ou en ajouter un nouveau.

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez gérer les sous-domaines.
3. Dans le volet de navigation, choisissez Hosting, puis Custom domains.
4. Sur la page Domaines personnalisés, choisissez Configuration du domaine.
5. Dans la section Sous-domaines, vous pouvez modifier vos sous-domaines existants selon vos besoins.
6. (Facultatif) Pour ajouter un nouveau sous-domaine, choisissez Ajouter un nouveau.

7. Choisissez Enregistrer.

Configuration de sous-domaines génériques

Amplify Hosting prend désormais en charge les sous-domaines génériques. Un sous-domaine générique est un sous-domaine fourre-tout qui vous permet de rediriger des sous-domaines existants et inexistants vers une branche spécifique de votre application. Lorsque vous utilisez un caractère générique pour associer tous les sous-domaines d'une application à une branche spécifique, vous pouvez proposer le même contenu aux utilisateurs de votre application dans n'importe quel sous-domaine et éviter de configurer chaque sous-domaine individuellement.

Pour créer un sous-domaine générique, spécifiez un astérisque (*) comme nom de sous-domaine. Par exemple, si vous spécifiez le sous-domaine générique `*.example.com` pour une branche spécifique de votre application, toute URL se terminant par exemple.com sera acheminée vers la branche. Dans ce cas, les demandes concernant `dev.example.com` et `prod.example.com` seront acheminées vers le `*.example.com` sous-domaine.

Notez qu'Amplify prend en charge les sous-domaines génériques uniquement pour un domaine personnalisé. Vous ne pouvez pas utiliser cette fonctionnalité avec le `amplifyapp.com` domaine par défaut.

Les exigences suivantes s'appliquent aux sous-domaines génériques :

- Le nom du sous-domaine doit être indiqué uniquement par un astérisque (*).
- Vous ne pouvez pas utiliser un caractère générique pour remplacer une partie du nom d'un sous-domaine, comme ceci : *domain.example.com.
- Vous ne pouvez pas remplacer un sous-domaine au milieu d'un nom de domaine, comme ceci : sous-domain.*.example.com.
- Par défaut, tous les certificats fournis par Amplify couvrent tous les sous-domaines d'un domaine personnalisé.

Pour ajouter ou supprimer un sous-domaine générique

Après avoir ajouté un domaine personnalisé à une application, vous pouvez ajouter un sous-domaine générique pour une branche d'application.

1. Connectez-vous à la console Amplify Hosting AWS Management Console et ouvrez-la.

2. Choisissez l'application pour laquelle vous souhaitez gérer les sous-domaines génériques.
3. Dans le volet de navigation, choisissez Hosting, puis Custom domains.
4. Sur la page Domaines personnalisés, choisissez Configuration du domaine.
5. Dans la section Sous-domaines, vous pouvez ajouter ou supprimer des sous-domaines génériques.
   - Pour ajouter un nouveau sous-domaine générique
     a. Choisissez Add new (Ajouter nouveau).
     b. Pour le sous-domaine, entrez un *****.
     c. Pour la branche de votre application, sélectionnez un nom de branche dans la liste.
     d. Choisissez Enregistrer.
   - Pour supprimer un sous-domaine générique
     a. Choisissez Supprimer à côté du nom du sous-domaine. Le trafic vers un sous-domaine qui n'est pas configuré explicitement s'arrête et Amplify Hosting renvoie un code d'état 404 à ces demandes.
     b. Choisissez Enregistrer.

Configuration de sous-domaines automatiques pour un domaine personnalisé Amazon Route 53

Une fois qu'une application est connectée à un domaine personnalisé dans Route 53, Amplify vous permet de créer automatiquement des sous-domaines pour les succursales nouvellement connectées. Par exemple, si vous connectez votre branche de développement, Amplify peut créer automatiquement dev.exampledomain.com. Lorsque vous supprimez une branche, tous les sous-domaines associés sont automatiquement supprimés.

Pour configurer la création automatique de sous-domaines pour les succursales nouvellement connectées

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez une application connectée à un domaine personnalisé géré dans Route 53.
3. Dans le volet de navigation, choisissez Hosting, puis Custom domains.
4. Sur la page Domaines personnalisés, choisissez Configuration du domaine.

5. Dans la section Création automatique de sous-domaines, activez la fonctionnalité.

> **Note**
>
> Cette fonctionnalité n'est disponible que pour les domaines racine, par exemple exempledomain.com. La console Amplify n'affiche pas cette case à cocher si votre domaine est déjà un sous-domaine, tel que dev.exampledomain.com.

Aperçus Web avec sous-domaines

Une fois que vous avez activé la création automatique de sous-domaines en suivant les instructions précédentes, les aperçus Web par pull request de votre application seront également accessibles avec les sous-domaines créés automatiquement. Lorsqu'une pull request est fermée, la branche et le sous-domaine associés sont automatiquement supprimés. Pour plus d'informations sur la configuration des aperçus Web pour les pull requests, consultezAperçus Web pour les pull requests.

Résolution des problèmes liés aux domaines personnalisés

Si vous rencontrez des problèmes lors de l'ajout d'un domaine personnalisé à une application dans la AWS Amplify console, consultez Résolution des problèmes liés aux domaines personnalisés le chapitre de résolution des problèmes d'Amplify. Si vous n'y trouvez pas de solution à votre problème, contactez Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .

Support de pare-feu pour les sites hébergés par Amplify

La prise en charge du pare-feu pour les sites hébergés par Amplify vous permet de protéger vos applications Web grâce à une intégration directe avec. AWS WAF AWS WAF vous permet de configurer un ensemble de règles, appelé liste de contrôle d'accès Web (ACL Web), qui autorisent, bloquent ou surveillent (comptent) les requêtes Web sur la base de règles et de conditions de sécurité Web personnalisables que vous définissez. Lorsque vous intégrez votre application Amplify à AWS WAF, vous gagnez en contrôle et en visibilité sur le trafic HTTP accepté par votre application. Pour en savoir plus AWS WAF, consultez la section Comment AWS WAF fonctionne dans le guide du AWS WAF développeur.

La prise en charge du pare-feu est disponible dans tous les Régions AWS établissements dans lesquels Amplify Hosting fonctionne. Cette intégration s'inscrit dans le cadre d'une ressource AWS WAF globale, similaire à CloudFront. Le Web ACLs peut être connecté à plusieurs applications d'hébergement Amplify, mais elles doivent résider dans la même région.

Vous pouvez l'utiliser AWS WAF pour protéger votre application Amplify contre les exploits Web courants, tels que l'injection SQL et les scripts intersites. Cela peut affecter la disponibilité et les performances de votre application, compromettre la sécurité ou consommer des ressources excessives. Par exemple, vous pouvez créer des règles pour autoriser ou bloquer les demandes provenant de plages d'adresses IP spécifiées, les demandes provenant de blocs CIDR, les demandes provenant d'un pays ou d'une région spécifique, ou les demandes contenant du code SQL ou des scripts inattendus.

Vous pouvez également créer des règles qui correspondent à une chaîne spécifiée ou un modèle d'expression régulière dans les en-têtes HTTP, la méthode, la chaîne de requête, l'URI et le corps de la demande (limité aux 8 premiers Ko). En outre, vous pouvez créer des règles pour bloquer les événements provenant d'agents utilisateurs, de robots ou de scrapers de contenu spécifiques. Par exemple, vous pouvez utiliser des règles basées sur le débit pour spécifier le nombre de requêtes web que chaque adresse IP du client est autorisée à envoyer au cours d'une période de 5 minutes mise à jour en continu.

Pour en savoir plus sur les types de règles pris en charge et sur les AWS WAF fonctionnalités supplémentaires, consultez le guide du AWS WAF développeur et le guide de référence des AWS WAF API.

> **Important**
>
> La sécurité est une responsabilité partagée entre vous AWS et vous. AWS WAF n'est pas la solution à tous les problèmes de sécurité Internet et vous devez le configurer pour répondre à vos objectifs de sécurité et de conformité. Pour vous aider à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation AWS WAF, consultez la section Sécurité lors de votre utilisation du AWS WAF service.

Rubriques

Activation AWS WAF d'une application Amplify dans AWS Management Console

Vous pouvez activer AWS WAF les protections pour une application Amplify dans la console Amplify ou dans la console. AWS WAF

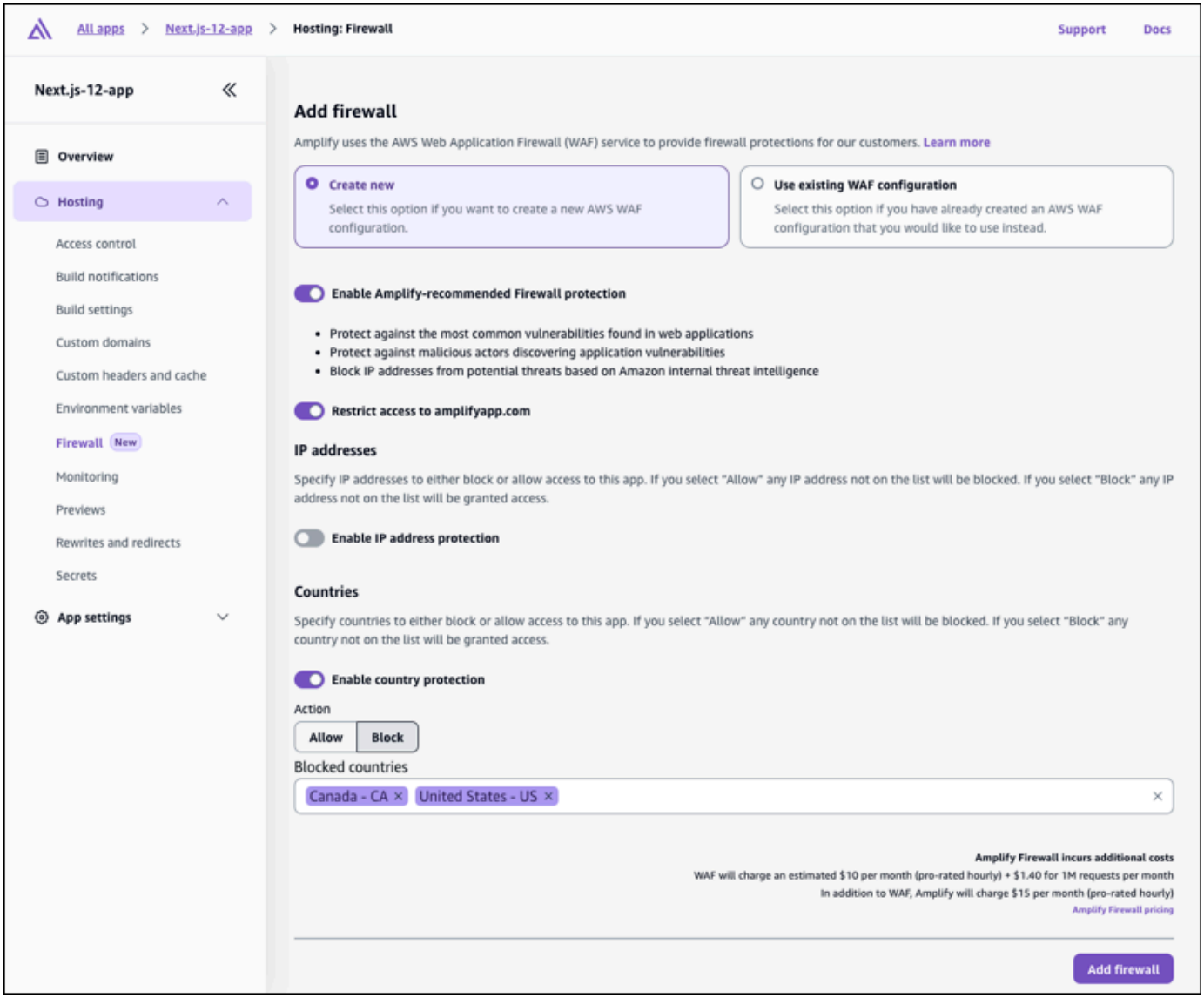
- Console Amplify : vous pouvez activer les fonctionnalités de pare-feu pour une application Amplify existante en associant une ACL AWS WAF Web à votre application dans la console Amplify. Utilisez la protection en un clic pour créer une ACL Web avec des règles préconfigurées que nous considérons comme les meilleures pratiques pour la plupart des applications. Vous avez la possibilité de personnaliser l'accès par adresse IP et par pays. Les instructions de cette section décrivent la configuration des protections en un clic.
- AWS WAF console — Utilisez une ACL Web préconfigurée que vous créez dans la AWS WAF console ou à l'aide du AWS WAF APIs. Vous devez créer un site Web ACLs que vous souhaitez associer à une application Amplify dans la région Global (CloudFront). Le Web régional existe ACLs peut-être déjà dans votre environnement Compte AWS, mais il n'est pas compatible avec Amplify. Pour les instructions de démarrage, consultez la section Configuration AWS WAF et ses composants dans le Guide du AWS WAF développeur.

Utilisez la procédure suivante AWS WAF pour activer une application existante dans la console Amplify.

Activer AWS WAF pour une application Amplify existante

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez le nom de l'application déployée pour activer la fonctionnalité Pare-feu.
3. Dans le volet de navigation, choisissez Hosting, puis Firewall.

   La capture d'écran suivante montre comment accéder à la page Ajouter un pare-feu dans la console Amplify.



4. Sur la page Ajouter un pare-feu, vos actions varient selon que vous souhaitez créer une nouvelle AWS WAF configuration ou utiliser une configuration existante.

   - Créez une nouvelle AWS WAF configuration.

     a. Choisissez Créer.

     b. Activez éventuellement l'une des configurations suivantes :

        i. Activez Activer la protection par pare-feu recommandée par Amplify.

        ii. Activez Restreindre l'accès à amplifyapp.com pour empêcher l'accès à votre application sur le domaine Amplify par défaut.

        iii. Pour les adresses IP, activez Activer la protection des adresses IP.

           A. Pour Action, choisissez Autoriser si vous souhaitez spécifier les adresses IP qui y auront accès et toutes les autres seront bloquées. Choisissez Bloquer si vous souhaitez spécifier les adresses IP qui seront bloquées et auxquelles toutes les autres auront accès.

           B. Pour la version IP, sélectionnez IPV4soit IPV6.

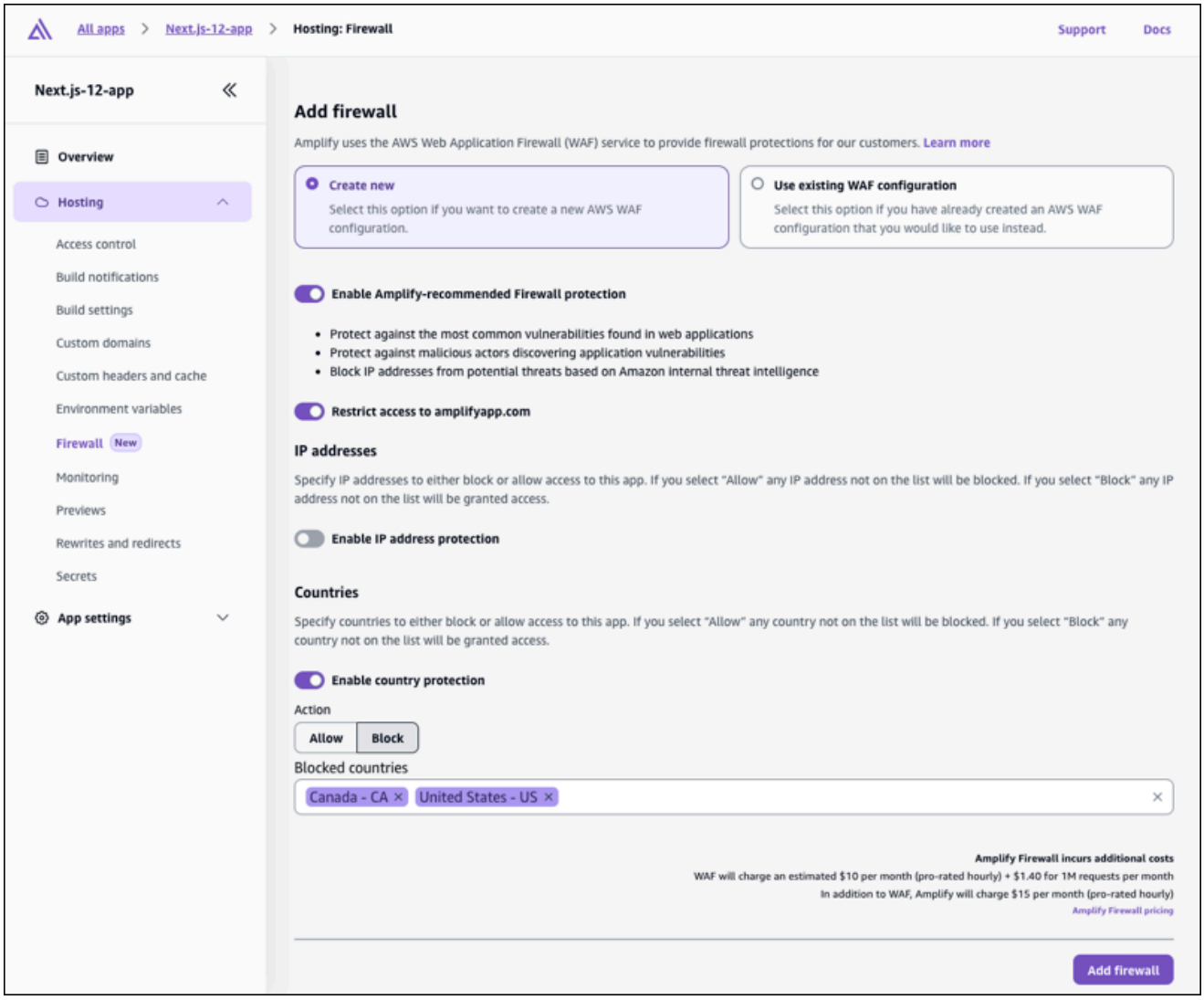
           C. Dans la zone de texte Adresses IP, entrez vos adresses IP autorisées ou bloquées, une par ligne, au format CIDR.

        iv. Pour les pays, activez Activer la protection du pays.

           A. Pour Action, choisissez Autoriser si vous souhaitez spécifier les pays qui y auront accès et tous les autres seront bloqués. Choisissez Bloquer si vous souhaitez spécifier les pays qui seront bloqués et auxquels tous les autres auront accès.

           B. Pour les pays, sélectionnez les pays autorisés ou bloqués dans la liste.

     La capture d'écran suivante montre comment activer une nouvelle AWS WAF configuration pour une application.



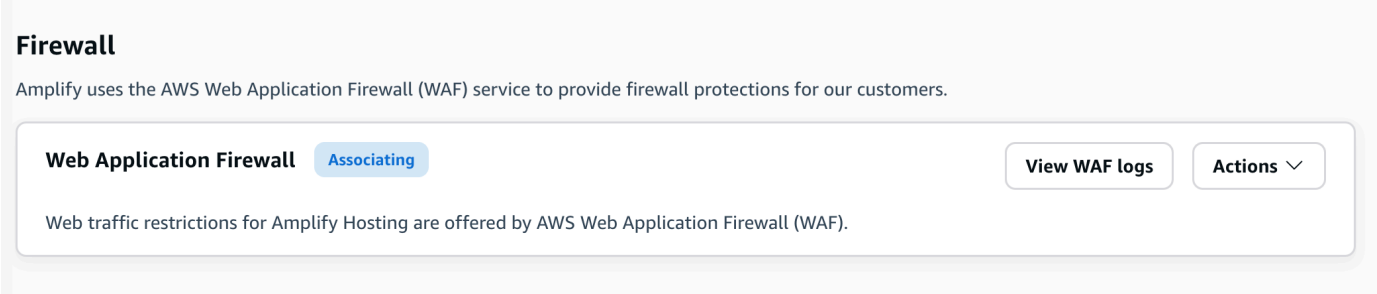
- Utilisez une AWS WAF configuration existante.

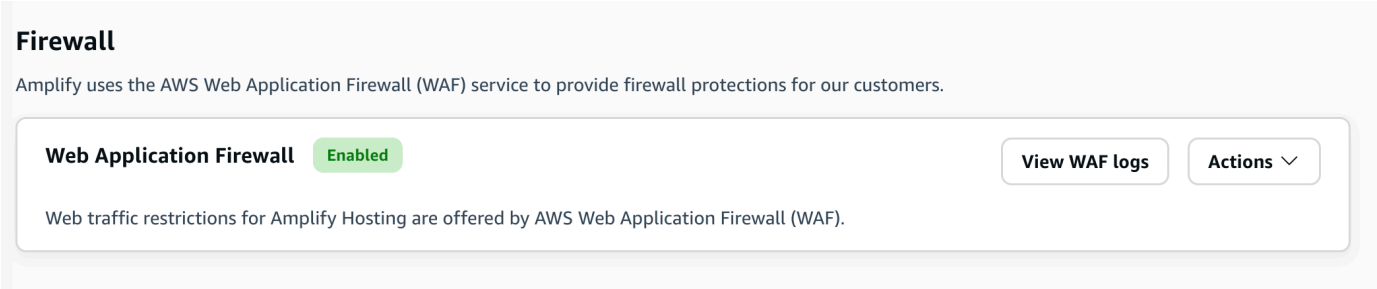
  a. Choisissez Utiliser la AWS WAF configuration existante.

  b. Sélectionnez une configuration enregistrée dans la liste des sites Web ACLs AWS WAF de votre Compte AWS. L'ACL Web que vous associez à votre application Amplify doit être créée dans la région Global (CloudFront). Le Web régional existe ACLs peut-être déjà dans votre environnement Compte AWS, mais il n'est pas compatible avec Amplify.

5. Choisissez Ajouter un pare-feu.
6. Sur la page Pare-feu, l'état d'association est affiché pour indiquer que les AWS WAF paramètres sont en cours de propagation. Lorsque le processus est terminé, le statut passe à Activé.

Les captures d'écran suivantes montrent l'état de progression du pare-feu dans la console Amplify, indiquant à quel moment la AWS WAF configuration est associée et activée.





Dissocier une ACL Web d'une application Amplify

Vous ne pouvez pas supprimer une ACL Web associée à une application Amplify. Vous devez d'abord dissocier l'ACL Web de l'application dans la console Amplify. Vous pouvez ensuite le supprimer dans la AWS WAF console.

Pour dissocier une ACL Web d'une application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez le nom de l'application dont vous souhaitez dissocier une ACL Web.
3. Dans le volet de navigation, choisissez Hosting, puis Firewall.
4. Sur la page Pare-feu, choisissez Actions, puis Dissocier le pare-feu.
5. Dans le mode de confirmation, entrez**`disassociate`**, puis choisissez Dissocier le pare-feu.
6. Sur la page Pare-feu, l'état de dissociation est affiché pour indiquer que les AWS WAF paramètres sont en cours de propagation.

Lorsque le processus est terminé, vous pouvez supprimer l'ACL Web dans la AWS WAF console.

Activation AWS WAF d'une application Amplify à l'aide du AWS CDK

Vous pouvez utiliser le AWS Cloud Development Kit (AWS CDK) AWS WAF pour activer une application Amplify. Pour en savoir plus sur l'utilisation du CDK, voir Qu'est-ce que le CDK ? dans le Guide AWS Cloud Development Kit (AWS CDK) du développeur.

L'exemple de TypeScript code suivant montre comment créer une AWS CDK application avec deux piles de CDK : une pour Amplify et une pour. AWS WAF Notez que la AWS WAF pile doit être déployée dans la région USA Est (Virginie du Nord) (us-east-1). La pile d'applications Amplify peut être déployée dans une autre région. Vous devez créer l'ACL Web que vous souhaitez associer à l'application Amplify dans la région Global (CloudFront). Le Web régional existe ACLs peut-être déjà dans votre environnement Compte AWS, mais il n'est pas compatible avec Amplify.

```
import * as cdk from "aws-cdk-lib";
import { Construct } from "constructs";
import * as wafv2 from "aws-cdk-lib/aws-wafv2";
import * as amplify from "aws-cdk-lib/aws-amplify";

interface WafStackProps extends cdk.StackProps {
  appArn: string;
}

export class AmplifyStack extends cdk.Stack {
  public readonly appArn: string;
  constructor(scope: Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);
    const amplifyApp = new amplify.CfnApp(this, "AmplifyApp", {
      name: "MyApp",
    });
    this.appArn = amplifyApp.attrArn;
  }
}

export class WAFStack extends cdk.Stack {
  constructor(scope: Construct, id: string, props: WafStackProps) {
```

```
    super(scope, id, props);
    const webAcl = new wafv2.CfnWebACL(this, "WebACL", {
      defaultAction: { allow: {} },
      scope: "CLOUDFRONT",
      rules: [
        // Add your own rules here.
      ],
      visibilityConfig: {
        cloudWatchMetricsEnabled: true,
        metricName: "my-metric-name",
        sampledRequestsEnabled: true,
      },
    });

    new wafv2.CfnWebACLAssociation(this, "WebACLAssociation", {
      resourceArn: props.appArn,
      webAclArn: webAcl.attrArn,
    });
  }
}

const app = new cdk.App();

// Create AmplifyStack in your desired Region.
const amplifyStack = new AmplifyStack(app, 'AmplifyStack', {
  env: { region: 'us-west-2' },
});

// Create WAFStack in IAD region, passing appArn from AmplifyStack.
new WAFStack(app, 'WAFStack', {
  env: { region: 'us-east-1' },
  crossRegionReferences: true,

  appArn: amplifyStack.appArn,  // Pass appArn from AmplifyStack.
});
```

Comment Amplify s'intègre à AWS WAF

La liste suivante fournit des informations spécifiques sur la manière dont le support du pare-feu est intégré AWS WAF et sur les contraintes à prendre en compte lors de la création de sites Web ACLs et de leur association aux applications Amplify.

- Vous pouvez l'activer AWS WAF pour n'importe quel type d'application Amplify. Cela inclut tous les frameworks pris en charge, les applications de rendu côté serveur (SSR) et les sites entièrement statiques. AWS WAF est compatible avec les applications Amplify Gen 1 et Gen 2.
- Vous devez créer un site Web ACLs que vous souhaitez associer à une application Amplify dans la région Global (CloudFront). Le Web régional existe ACLs peut-être déjà dans votre environnement Compte AWS, mais il n'est pas compatible avec Amplify.
- L'ACL Web et l'application Amplify doivent être créés de la même manière. Compte AWS Vous pouvez l'utiliser AWS Firewall Manager pour répliquer AWS WAF les règles Comptes AWS, afin de simplifier la centralisation et la distribution des règles d'entreprise sur plusieurs Comptes AWS sites. Pour plus d'informations, consultez AWS Firewall Manager dans le Guide du développeur AWS WAF .
- Vous pouvez partager la même ACL Web sur plusieurs applications Amplify de la même manière. Compte AWS Toutes les applications doivent se trouver dans la même région.
- Lorsque vous associez une ACL Web à une application Amplify, l'ACL Web est attachée par défaut à chaque branche de l'application. Lorsque vous créez de nouvelles branches, elles disposeront de l'ACL Web.
- Lorsque vous associez une ACL Web à une application Amplify, elle est automatiquement associée à tous les domaines de l'application. Cependant, vous pouvez configurer des règles qui s'appliquent à un seul nom de domaine à l'aide des règles de correspondance des en-têtes d'hôte.
- Vous ne pouvez pas supprimer une ACL Web associée à une application Amplify. Avant de supprimer une ACL Web dans la AWS WAF console, vous devez la dissocier de l'application.

Politique de ressources d'Amplify Web ACL

Pour permettre à Amplify d'accéder à votre ACL Web, une politique de ressources est attachée à l'ACL Web lors de l'association. Amplify construit automatiquement cette politique de ressources, mais vous pouvez la consulter à l'aide de l'API. AWS WAFV2 GetPermissionPolicy Les autorisations IAM suivantes sont requises pour associer une ACL Web à une application Amplify.

- amplifier : ACL AssociateWeb
- wafv2 : ACL AssociateWeb
- wafv2 : PutPermissionPolicy
- wafv2 : GetPermissionPolicy

Tarification du pare-feu pour les applications Amplify

Le coût de mise en œuvre AWS WAF sur une application Amplify est calculé sur la base des deux composants suivants :

- AWS WAF utilisation — Votre AWS WAF utilisation vous sera facturée conformément au modèle de AWS WAF tarification. AWS WAF les frais sont basés sur les listes de contrôle d'accès Web (Web ACLs) que vous créez, le nombre de règles que vous ajoutez par ACL Web et le nombre de demandes Web que vous recevez. Pour plus d'informations sur la tarification, consultez la page AWS WAF Pricing (Tarification).
- Coût d'intégration d'Amplify Hosting — Des frais de 15$ par mois et par application sont facturés lorsque vous attachez une ACL Web à une application Amplify. Ce montant est calculé au prorata toutes les heures.

Flux de travail de déploiements de branches de fonctionnalités et flux de travail d'équipe

Amplify Hosting est conçu pour fonctionner avec les branches de fonctionnalités et GitFlow les flux de travail. Amplify utilise les branches Git pour créer un nouveau déploiement chaque fois que vous connectez une nouvelle branche dans votre dépôt. Après avoir connecté votre première branche, vous créez des branches de fonctionnalités supplémentaires.

Pour ajouter une branche à une application

1. Choisissez l'application à laquelle vous souhaitez ajouter une branche.
2. Choisissez Paramètres de l'application, puis Paramètres de la succursale.
3. Sur la page des paramètres de la branche, choisissez Ajouter une branche.
4. Sélectionnez une branche dans votre référentiel.
5. Choisissez Ajouter une branche.
6. Redéployez votre application.

Une fois que vous avez ajouté une branche, votre application dispose de deux déploiements disponibles sur les domaines par défaut d'Amplify, tels que https://main.appid.amplifyapp.comet. https://dev.appid.amplifyapp.com Cela peut être différent de team-to-team, mais c'est généralement la branche principale qui suit le code de publication et constitue votre branche de production. La branche de développement sert de branche d'intégration pour tester les nouvelles fonctionnalités. Cela permet aux bêta-testeurs de tester des fonctionnalités inédites lors du déploiement de la branche de développement, sans affecter les utilisateurs finaux de production participant au déploiement de la branche principale.

Rubriques

- Utiliser les backends Amplify dans toutes les applications (applications de première génération uniquement)

Flux de travail d'équipe avec les applications Amplify Gen 2 intégrales

AWS Amplify Gen 2 introduit une expérience de développement basée sur le code, TypeScript basée sur le code, pour définir les backends. Pour en savoir plus sur les flux de travail Fullstack avec les applications Amplify Gen 2, voir Workflows Fullstack dans la documentation Amplify.

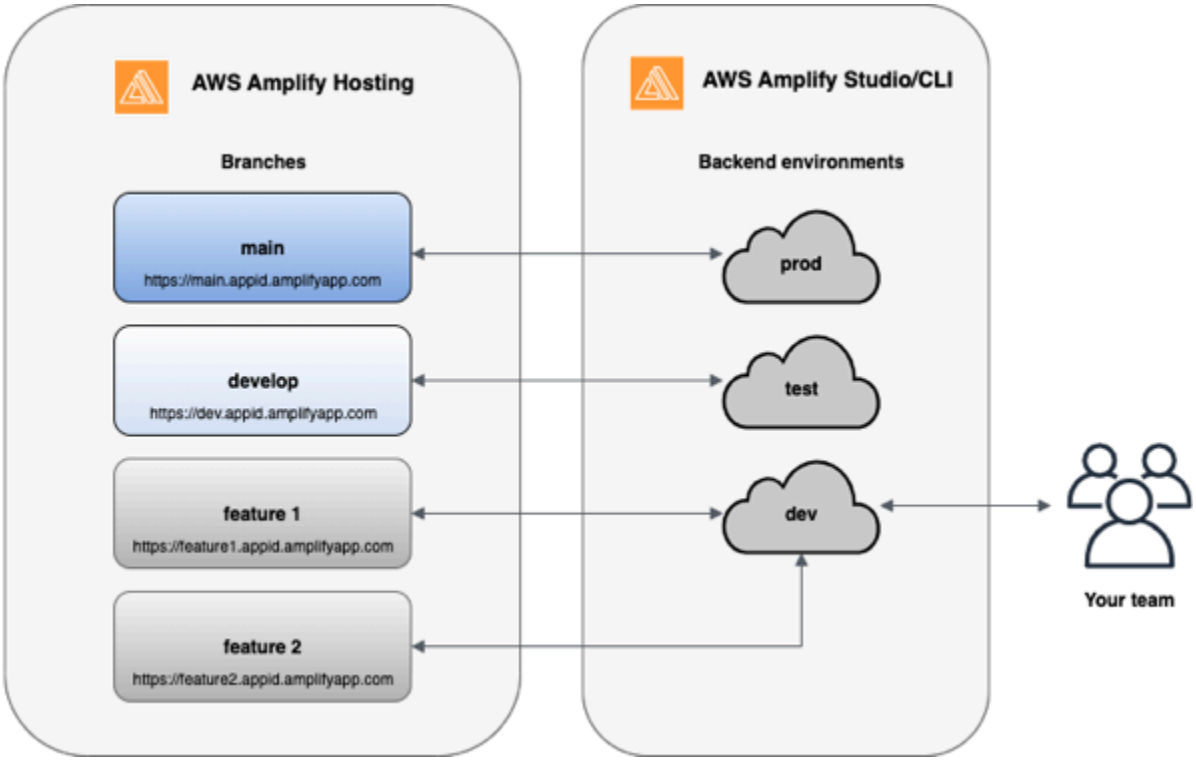
Flux de travail d'équipe avec les applications Amplify Gen 1 intégrales

Le déploiement d'une branche de fonctionnalités comprend un frontend et un environnement principal optionnel. Le frontend est créé et déployé sur un réseau mondial de diffusion de contenu (CDN), tandis que le backend est déployé par Amplify Studio ou la CLI Amplify pour. AWS Pour savoir comment configurer ce scénario de déploiement, consultezCréation d'un backend pour une application.

Amplify Hosting déploie en permanence des ressources dorsales telles que les fonctions GraphQL APIs et Lambda dans le cadre de vos déploiements de branches fonctionnelles. Vous pouvez utiliser les modèles de branchement suivants pour déployer votre backend et votre frontend avec Amplify Hosting.

Flux de travail de branche de fonctionnalités

- Créez des environnements de backend de production, de test et de développement avec Amplify Studio ou l'Amplify CLI.
- Mappez le backend de production à la branche principale.
- Mappez le backend de test à la branche de développement.
- Les membres de l'équipe peuvent utiliser l'environnement de développement principal pour tester des branches de fonctionnalités individuelles.



1. Installez l'interface de ligne de commande Amplify pour lancer un nouveau projet Amplify.

```
npm install -g @aws-amplify/cli
```

2. Créez un environnement backend prod pour votre projet. Si vous n'avez pas de projet, créez-en un à l'aide d'outils bootstrap tels que create-react-app Gatsby.

```
create-react-app next-unicorn
cd next-unicorn
amplify init
 ? Do you want to use an existing environment? (Y/n): n
 ? Enter a name for the environment: prod
...
amplify push
```

3. Ajoutez des environnements backen de test et de développement.

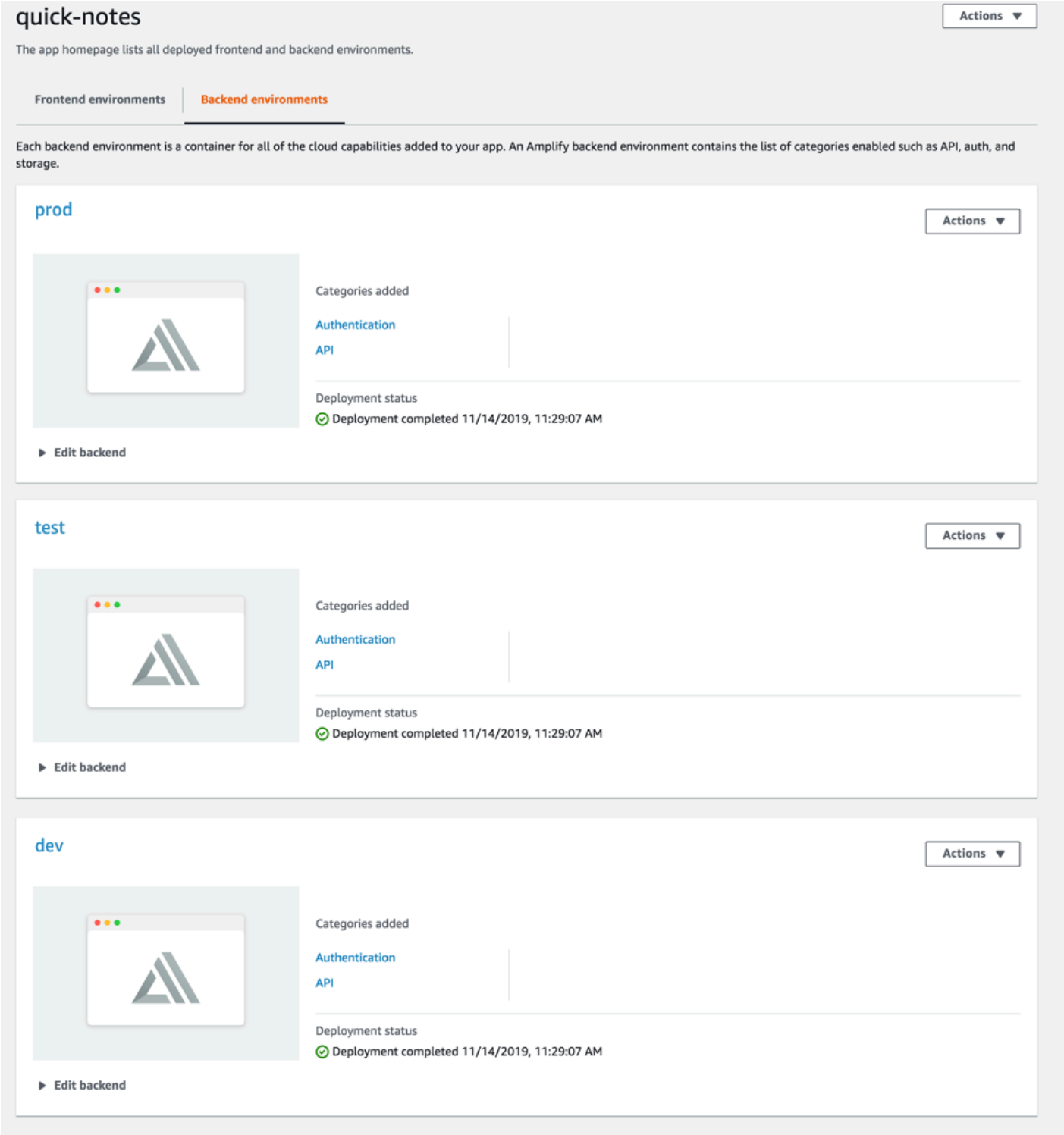
```
amplify env add
 ? Do you want to use an existing environment? (Y/n): n
 ? Enter a name for the environment: test
...
amplify push
```

```
amplify env add
 ? Do you want to use an existing environment? (Y/n): n
 ? Enter a name for the environment: dev
...
amplify push
```

4. Envoyez le code vers le dépôt Git de votre choix (dans cet exemple, nous supposerons que vous avez envoyé le code vers le dépôt principal).

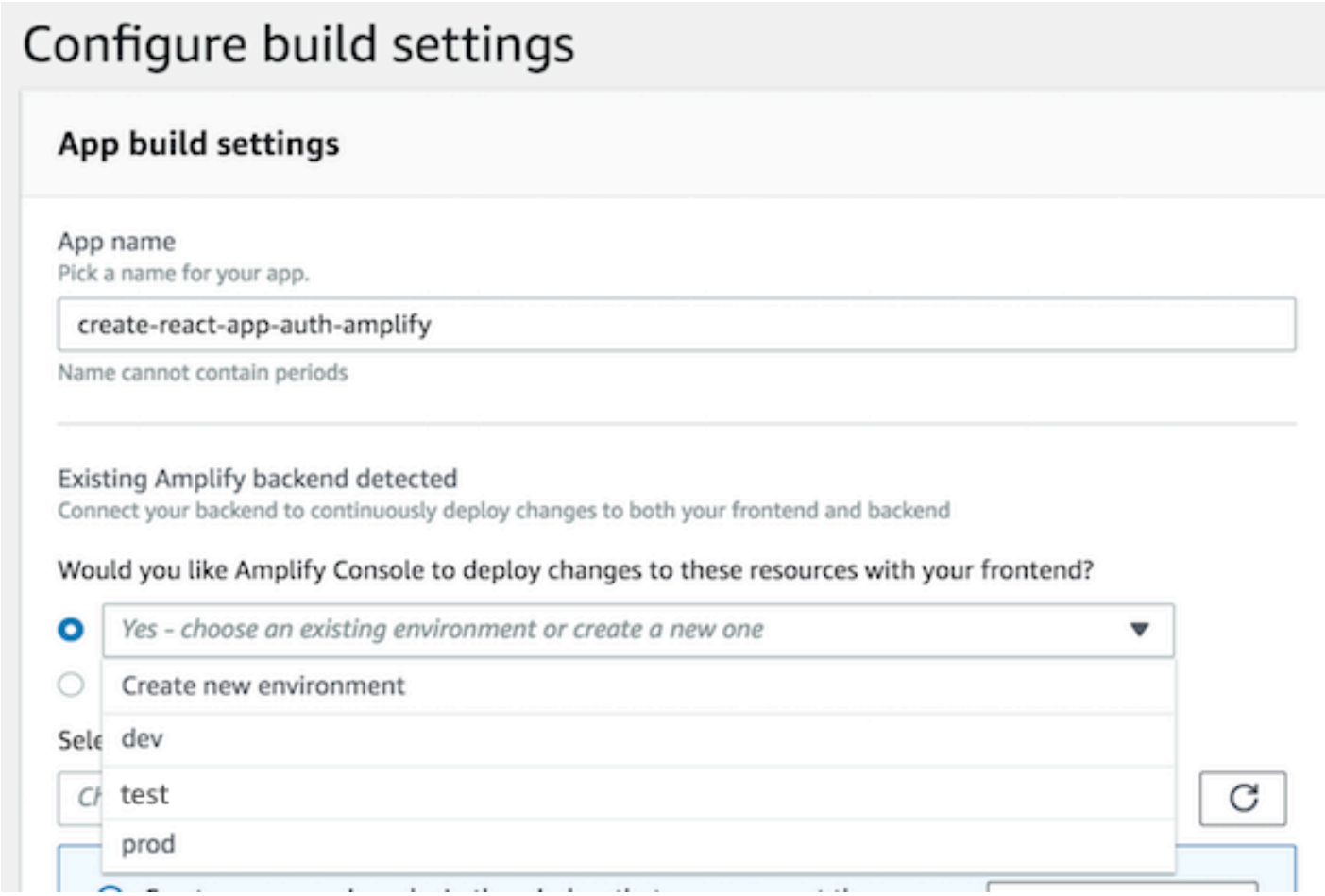
```
git commit -am 'Added dev, test, and prod environments'
git push origin main
```

5. Visitez Amplify dans le AWS Management Console pour voir votre environnement principal actuel. Naviguez d'un niveau supérieur depuis le fil d'Ariane pour afficher la liste de tous les environnements de backend créés dans l'onglet Environnements de backend.

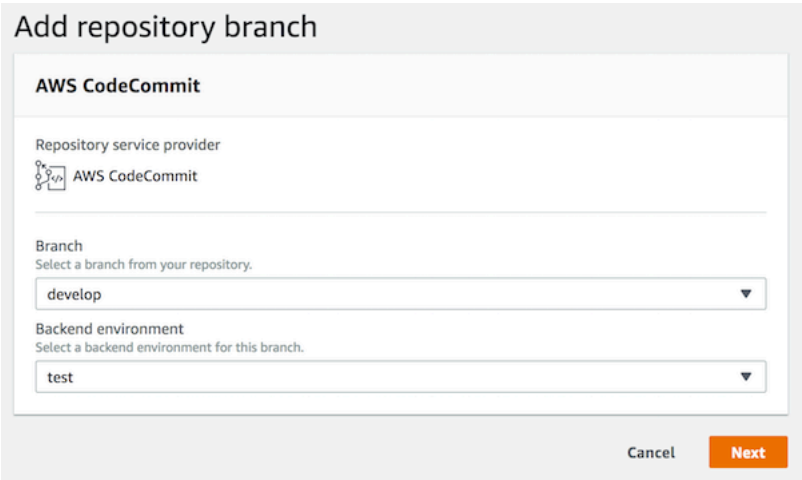


6. Accédez à l'onglet Environnements frontaux et connectez votre fournisseur de référentiel à votre branche principale.
7. Sur la page des paramètres de génération, sélectionnez un environnement principal existant pour configurer un déploiement continu avec la branche principale. Choisissez prod dans la liste et

accordez le rôle de service à Amplify. Choisissez Save and deploy (Enregistrer et déployer). Une fois la compilation terminée, vous pourrez accéder au déploiement de la branche principale à l'adresse https://main.appid.amplifyapp.com.



8. Connectez la branche de développement dans Amplify (supposons que le développement et la branche principale sont identiques à ce stade). Choisissez l'environnement backend test.



9. Amplify est maintenant configuré. Vous pouvez commencer à travailler sur les nouvelles fonctionnalités d'une branche. Pour ajouter une fonctionnalité backend, utilisez l'environnement backend dev à partir de votre poste de travail local.

```
git checkout -b newinternet
amplify env checkout dev
amplify add api
...
amplify push
```

10. Une fois que vous avez fini de travailler sur cette fonctionnalité, validez le code et créez une demande d'extraction afin d'effectuer un examen en interne.

```
git commit -am 'Decentralized internet v0.1'
git push origin newinternet
```

11. Pour prévisualiser à quoi ressembleront les modifications, accédez à la console Amplify et connectez votre branche de fonctionnalités. Remarque : Si vous l'avez AWS CLI installé sur votre système (pas l'Amplify CLI), vous pouvez connecter une branche directement depuis votre terminal. Pour localiser l'ID de votre application, accédez à App Settings (Paramètres de l'application) > General (Général) > AppARN : arn:aws:amplify:<région>:<région>:apps/<IDapp>.

```
aws amplify create-branch --app-id <appid> --branch-name <branchname>
aws amplify start-job --app-id <appid> --branch-name <branchname> --job-type RELEASE
```

12. Votre fonctionnalité sera accessible à l'adresse suivante https://newinternet.appid.amplifyapp.compour la partager avec vos coéquipiers. Si tout semble correct, fusionnez la branche PR avec la branche de développement.

```
git checkout develop
git merge newinternet
git push
```

13. Cela lancera une version qui mettra à jour le backend ainsi que le frontend dans Amplify avec un déploiement de branche sur. https://dev.appid.amplifyapp.com Vous pouvez partager ce lien avec les intervenants internes afin qu'ils puissent examiner cette nouvelle fonctionnalité.

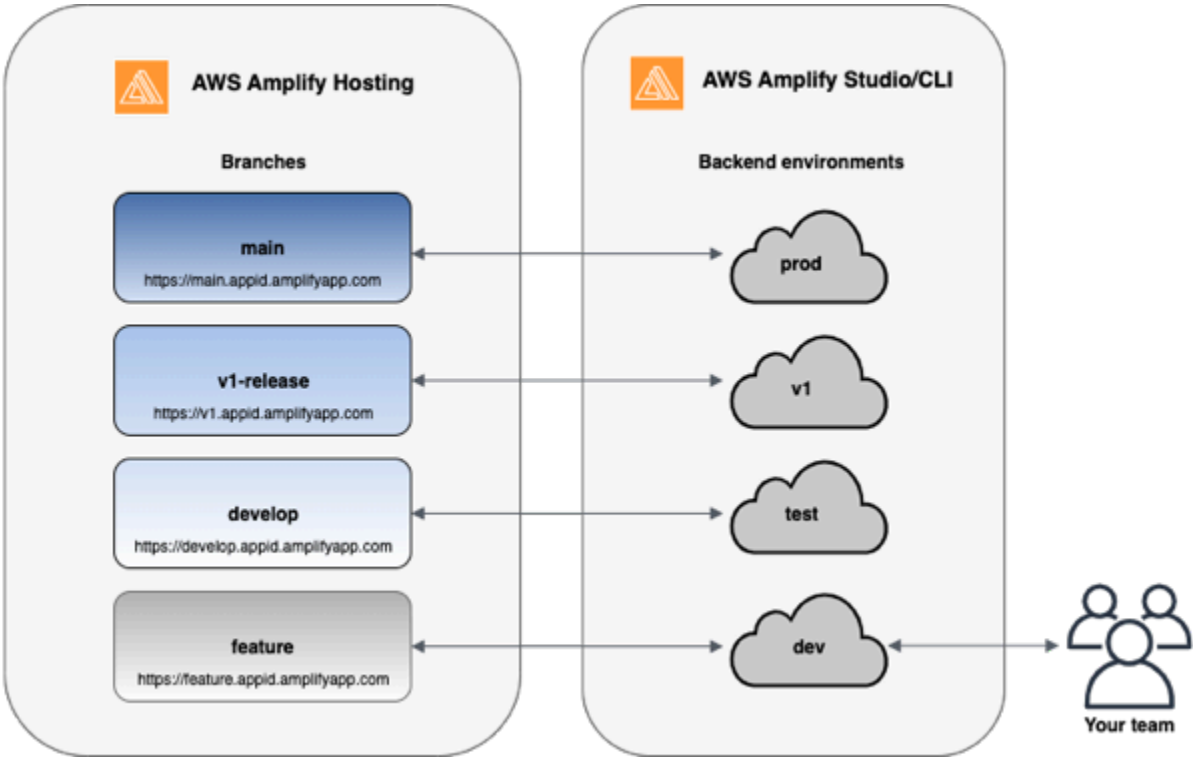
14. Supprimez votre branche de fonctionnalités de Git, Amplify et supprimez l'environnement principal du cloud (vous pouvez toujours en créer une nouvelle en exécutant « amplify env checkout prod » et « amplify env add »).

```
git push origin --delete newinternet
aws amplify delete-branch --app-id <appid> --branch-name <branchname>
amplify env remove dev
```

GitFlow flux de travail

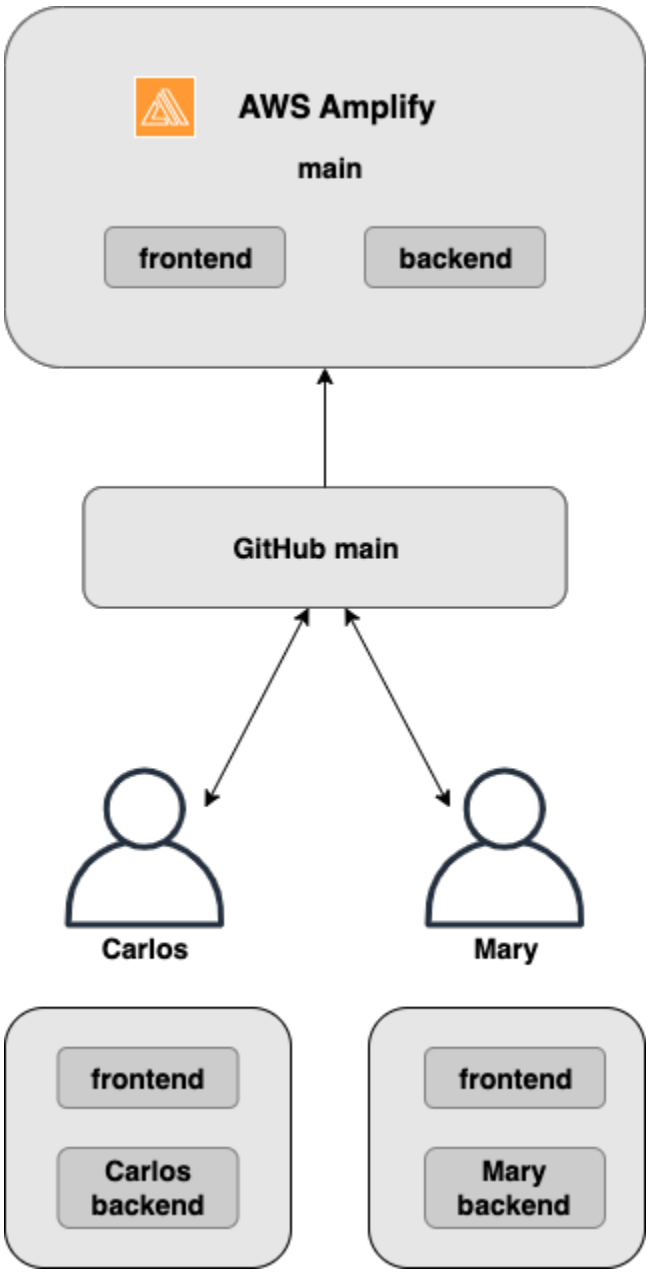
GitFlow utilise deux branches pour enregistrer l'historique du projet. La branche principale suit uniquement le code de version, et la branche de développement est utilisée comme branche d'intégration pour les nouvelles fonctionnalités. GitFlow simplifie le développement parallèle en isolant les nouveaux développements des travaux achevés. Le nouveau développement (comme les fonctionnalités et les correctifs de bogues non urgents) s'effectue dans les branches de fonctionnalités. Lorsque le développeur juge que le code peut être publié, la branche de fonctionnalités est fusionnée dans la branche de développement pour l'intégration. Les seules validations vers la branche principale sont les fusions depuis les branches de version et les branches de correctifs (pour corriger les bogues d'urgence).

Le schéma ci-dessous montre une configuration recommandée avec GitFlow. Vous pouvez suivre le même processus, comme décrit dans la section ci-dessus sur les flux de travail des branches de fonctionnalités.



Un sandbox pour chaque développeur

- Chaque développeur d'une équipe crée un environnement de test (sandbox) dans le cloud qui est distinct de son ordinateur local. Cela permet aux développeurs de travailler indépendamment les uns des autres sans remplacer les modifications apportées par les autres membres de l'équipe.
- Chaque branche d'Amplify possède son propre backend. Cela garantit qu'Amplify utilise le référentiel Git comme source fiable unique à partir de laquelle déployer les modifications, plutôt que de compter sur les développeurs de l'équipe pour transférer manuellement leur backend ou leur front-end en production depuis leurs ordinateurs locaux.



1. Installez l'interface de ligne de commande Amplify pour lancer un nouveau projet Amplify.

```
npm install -g @aws-amplify/cli
```

2. Initialisez un environnement principal Mary pour votre projet. Si vous n'avez pas de projet, créez-en un à l'aide d'outils bootstrap tels que create-react-app Gatsby.

```
cd next-unicorn
amplify init
 ? Do you want to use an existing environment? (Y/n): n
```

```
 ? Enter a name for the environment: mary
...
amplify push
```

3. Envoyez le code vers le dépôt Git de votre choix (dans cet exemple, nous supposerons que vous avez envoyé le code au dépôt principal).

```
git commit -am 'Added mary sandbox'
git push origin main
```

4. Connectez votre repo **>** main à Amplify.
5. La console Amplify détectera les environnements principaux créés par la CLI Amplify. Choisissez Créer un nouvel environnement dans la liste déroulante et accordez le rôle de service à Amplify. Choisissez Save and deploy (Enregistrer et déployer). Une fois la construction terminée, vous obtiendrez un déploiement de branche principale disponible sur https://main.appid.amplifyapp.comavec un nouvel environnement principal lié à la branche.
6. Connectez la branche de développement dans Amplify (supposons que le développement et la branche principale sont identiques à ce stade) et choisissez Create

Déploiements de branches de fonctionnalités basés sur des modèles

Les déploiements de branches basés sur des modèles vous permettent de déployer automatiquement des branches correspondant à un modèle spécifique dans Amplify. Les équipes produit qui utilisent des branches de fonctionnalités ou des GitFlow flux de travail pour leurs versions peuvent désormais définir des modèles tels **`release**`** que le déploiement automatique de branches Git commençant par « release » vers une URL partageable.

1. Choisissez Paramètres de l'application, puis Paramètres de la succursale.
2. Sur la page des paramètres de la branche, choisissez Modifier.
3. Sélectionnez Détection automatique des branches pour connecter automatiquement les branches à Amplify qui correspondent à un ensemble de modèles.
4. Dans le champ Détection automatique des branches - modèles, entrez les modèles pour le déploiement automatique des branches.
   - **`*`**— Déploie toutes les branches de votre dépôt.
   - **`release*`**— Déploie toutes les branches commençant par le mot « release ».

- **`release*/`**— Déploie toutes les branches correspondant au modèle « release/».
- Spécifiez plusieurs modèles dans une liste séparée par des virgules. Par exemple, **`release*, feature*`**.

5. Configurez la protection automatique par mot de passe pour toutes les succursales créées automatiquement en sélectionnant Contrôle d'accès par détection automatique des succursales.
6. Pour les applications Gen 1 créées avec un backend Amplify, vous pouvez choisir de créer un nouvel environnement pour chaque branche connectée ou de pointer toutes les branches vers un backend existant.
7. Choisissez Enregistrer.

Déploiements de branches de fonctionnalités basés sur des modèles pour une application connectée à un domaine personnalisé

Vous pouvez utiliser des déploiements de branches de fonctionnalités basés sur des modèles pour une application connectée à un domaine personnalisé Amazon Route 53.

- Pour obtenir des instructions sur la configuration de déploiements de branches de fonctionnalités basés sur des modèles, voir Configuration de sous-domaines automatiques pour un domaine personnalisé Amazon Route 53
- Pour obtenir des instructions sur la connexion d'une application Amplify à un domaine personnalisé géré dans Route 53, voir Ajouter un domaine personnalisé géré par Amazon Route 53
- Pour plus d'informations sur l'utilisation de Route 53, consultez Qu'est-ce qu'Amazon Route 53 ?

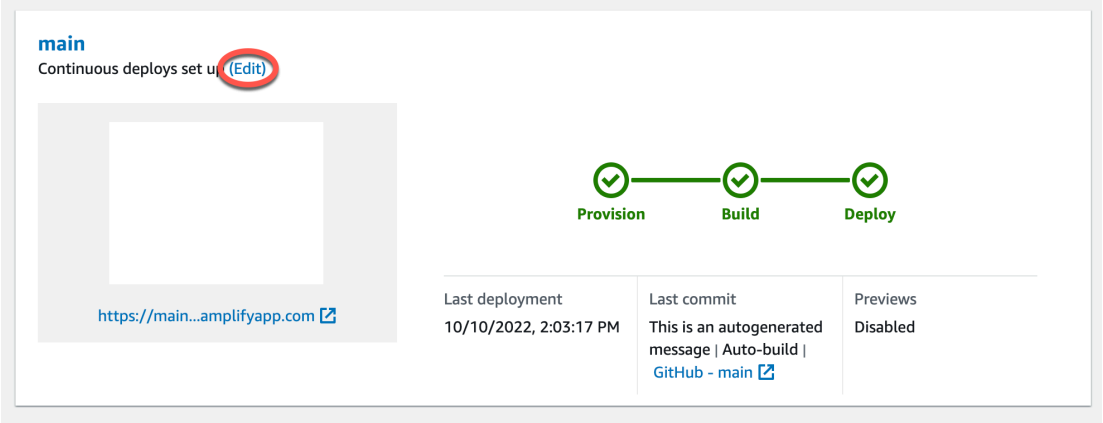
Génération automatique de la configuration Amplify au moment de la construction (applications Gen 1 uniquement)

> **Note**
>
> Les informations contenues dans cette section concernent uniquement les applications de première génération. Si vous souhaitez déployer automatiquement les modifications d'infrastructure et de code d'application à partir des branches de fonctionnalités d'une application de deuxième génération, consultez la section Déploiements de branches Fullstack dans la documentation Amplify

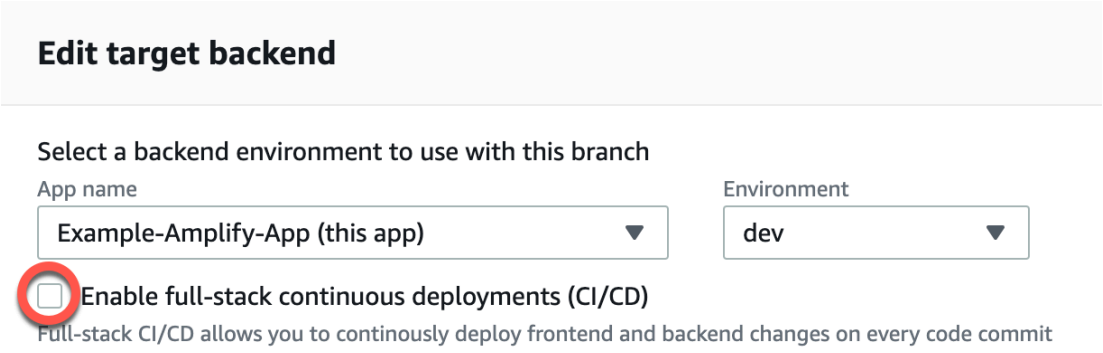
Amplify prend en charge la génération automatique au moment de la génération du fichier de configuration `aws-exports.js` Amplify pour les applications de première génération. En désactivant les déploiements CI/CD complets, vous permettez à votre application de générer automatiquement le `aws-exports.js` fichier et de vous assurer qu'aucune mise à jour n'est apportée à votre backend au moment de la création.

Pour générer automatiquement au **`aws-exports.js`** moment de la construction

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à modifier.
3. Choisissez l'onglet Environnements d'hébergement.
4. Localisez la branche à modifier et choisissez Modifier.



5. Sur la page Modifier le backend cible, décochez Activer les déploiements continus full-stack (CI/CD) pour désactiver le CI/CD full stack pour ce backend.



6. Sélectionnez un rôle de service existant pour accorder à Amplify les autorisations dont il a besoin pour apporter des modifications au backend de votre application. Si vous devez créer un rôle de service, choisissez Créer un nouveau rôle. Pour plus d'informations sur la création d'un rôle de

service, consultez la section Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend.

7. Choisissez Enregistrer. Amplify appliquera ces modifications lors de la prochaine création de l'application.

Constructions de backend conditionnelles (applications Gen 1 uniquement)

> **Note**
>
> Les informations contenues dans cette section concernent uniquement les applications de première génération. Amplify Gen 2 introduit une expérience de développement TypeScript basée sur le code et axée sur le code. Par conséquent, cette fonctionnalité n'est pas nécessaire pour les backends de 2e génération.

Amplify prend en charge les builds de backend conditionnels sur toutes les branches d'une application Gen 1. Pour configurer les versions conditionnelles du backend, définissez la variable d'`AMPLIFY_DIFF_BACKEND`environnement sur`true`. L'activation des versions conditionnelles du backend aidera à accélérer les builds où les modifications ne sont apportées qu'au frontend.

Lorsque vous activez les versions de backend basées sur le diff, au début de chaque build, Amplify tente d'exécuter un diff sur le `amplify` dossier de votre référentiel. Si Amplify ne trouve aucune différence, il ignore l'étape de création du backend et ne met pas à jour vos ressources backend. Si votre projet ne contient aucun `amplify` dossier dans votre référentiel, Amplify ignore la valeur de la `AMPLIFY_DIFF_BACKEND` variable d'environnement. Pour obtenir des instructions sur la définition de la variable d'`AMPLIFY_DIFF_BACKEND`environnement, consultezConfiguration de versions de backend basées sur les différences pour une application de première génération.

Si des commandes personnalisées sont actuellement spécifiées dans les paramètres de génération de votre phase de backend, les builds de backend conditionnels ne fonctionneront pas. Si vous souhaitez que ces commandes personnalisées s'exécutent, vous devez les déplacer vers la phase frontale de vos paramètres de génération dans le `amplify.yml` fichier de votre application. Pour plus d'informations sur la mise à jour du `amplify.yml` fichier, consultezRéférence de spécification de construction.

Utiliser les backends Amplify dans toutes les applications (applications de première génération uniquement)

> **Note**
>
> Les informations contenues dans cette section concernent uniquement les applications de première génération. Si vous souhaitez partager les ressources du backend d'une application de deuxième génération, consultez la section Partager les ressources entre les branches dans la documentation Amplify

Amplify vous permet de réutiliser les environnements backend existants dans toutes vos applications de première génération dans une région donnée. Vous pouvez le faire lorsque vous créez une nouvelle application, connectez une nouvelle branche à une application existante ou mettez à jour un frontend existant pour qu'il pointe vers un autre environnement principal.

Réutiliser les backends lors de la création d'une nouvelle application

Pour réutiliser un backend lors de la création d'une nouvelle application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Pour créer un nouveau backend à utiliser dans cet exemple, procédez comme suit :
   a. Dans le volet de navigation, sélectionnez Toutes les applications.
   b. Choisissez Nouvelle application, puis Créez une application.
   c. Entrez un nom pour votre application, par exemple**`Example-Amplify-App`**.
   d. Choisissez Confirmer le déploiement.
3. Pour connecter un frontend à votre nouveau backend, choisissez l'onglet Environnements d'hébergement.
4. Choisissez votre fournisseur Git, puis sélectionnez Connect branch.
5. Sur la page Ajouter une branche de référentiel, pour Référentiels récemment mis à jour, choisissez le nom de votre référentiel. Pour Branch, sélectionnez la branche de votre référentiel à connecter.
6. Sur la page des paramètres de compilation, effectuez les opérations suivantes :

a. Dans Nom de l'application, sélectionnez l'application à utiliser pour ajouter un environnement principal. Vous pouvez choisir l'application actuelle ou n'importe quelle autre application de la région actuelle.

b. Pour Environnement, sélectionnez le nom de l'environnement principal à ajouter. Vous pouvez utiliser un environnement existant ou en créer un nouveau.

c. Par défaut, la fonction full stack CI/CD est désactivée. La désactivation du CI/CD complet entraîne l'exécution de l'application en mode pull uniquement. Au moment de la création, Amplify générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier votre environnement principal.

d. Sélectionnez un rôle de service existant pour accorder à Amplify les autorisations dont il a besoin pour apporter des modifications au backend de votre application. Si vous devez créer un rôle de service, choisissez Créer un nouveau rôle. Pour plus d'informations sur la création d'un rôle de service, consultez la section Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend.

e. Choisissez Suivant.

7. Choisissez Save and deploy (Enregistrer et déployer).

Réutiliser les backends lors de la connexion d'une branche à une application existante

Pour réutiliser un backend lors de la connexion d'une branche à une application Amplify existante

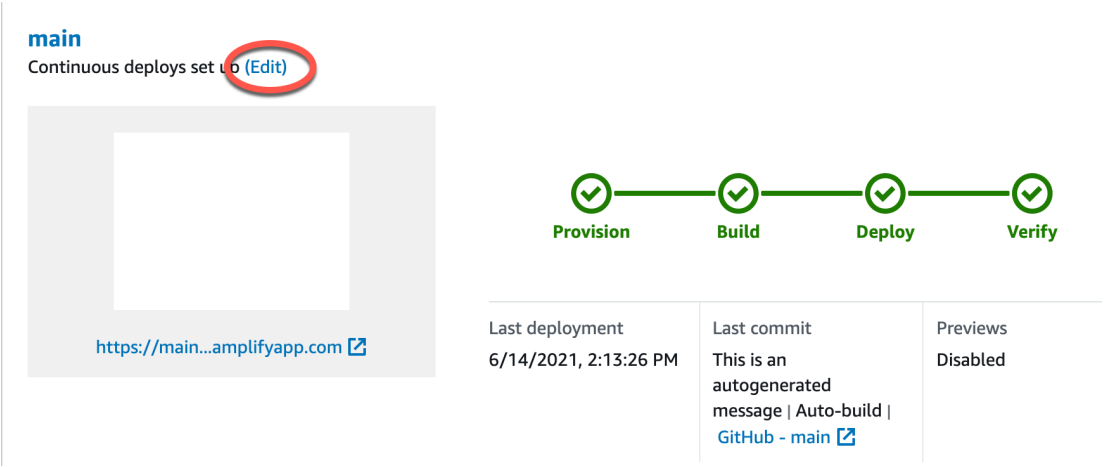
1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle connecter une nouvelle succursale.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
4. Dans la section Branches, choisissez Connect a branch.
5. Sur la page Ajouter une branche de dépôt, pour Branch, sélectionnez la branche de votre référentiel à connecter.
6. Dans Nom de l'application, sélectionnez l'application à utiliser pour ajouter un environnement principal. Vous pouvez choisir l'application actuelle ou n'importe quelle autre application de la région actuelle.
7. Pour Environnement, sélectionnez le nom de l'environnement principal à ajouter. Vous pouvez utiliser un environnement existant ou en créer un nouveau.

8. Si vous devez configurer un rôle de service pour accorder à Amplify les autorisations nécessaires pour apporter des modifications au backend de votre application, la console vous invite à effectuer cette tâche. Pour plus d'informations sur la création d'un rôle de service, consultez la section Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend.
9. Par défaut, la fonction full stack CI/CD est désactivée. La désactivation de Full-Stack CI/CD entraîne l'exécution de l'application en mode pull uniquement. Au moment de la création, Amplify générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier votre environnement principal.
10. Choisissez Suivant.
11. Choisissez Save and deploy (Enregistrer et déployer).

Modifier un frontend existant pour qu'il pointe vers un autre backend

Pour modifier un frontend, l'application Amplify afin qu'elle pointe vers un autre backend

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application dont vous souhaitez modifier le backend.
3. Choisissez l'onglet Environnements d'hébergement.
4. Localisez la branche à modifier et choisissez Modifier.



5. Sur la page Sélectionnez un environnement principal à utiliser avec cette branche, dans Nom de l'application, sélectionnez l'application frontale dont vous souhaitez modifier l'environnement principal. Vous pouvez choisir l'application actuelle ou n'importe quelle autre application de la région actuelle.
6. Pour l'environnement principal, sélectionnez le nom de l'environnement principal à ajouter.

7. Par défaut, Full-Stack CI/CD est activé. Décochez cette option pour désactiver le full stack CI/CD pour ce backend. La désactivation de Full-Stack CI/CD entraîne l'exécution de l'application en mode pull uniquement. Au moment de la création, Amplify générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier l'environnement principal.
8. Choisissez Enregistrer. Amplify appliquera ces modifications lors de la prochaine création de l'application.

Création d'un backend pour une application

Avec, AWS Amplify vous pouvez créer une application complète avec des données, une authentification, un stockage et un hébergement frontal déployée sur. AWS

AWS Amplify Gen 2 introduit une expérience de développement basée sur le code, TypeScript basée sur le code, pour définir les backends. Pour savoir comment utiliser Amplify Gen 2 pour créer et connecter un backend à votre application, consultez la section Créer et connecter un backend dans la documentation Amplify.

Si vous recherchez la documentation pour créer un backend pour une application Gen 1, à l'aide de la CLI et d'Amplify Studio, voir Build & connect backend dans la documentation Amplify Gen 1.

Rubriques

Création d'un backend pour une application de 2e génération

Pour un didacticiel qui vous guide à travers les étapes de création d'une application Amplify Gen 2 fullstack avec un backend TypeScript basé, voir Commencer dans la documentation Amplify.

Création d'un backend pour une application de première génération

Dans ce didacticiel, vous allez configurer un CI/CD flux de travail complet avec Amplify. Vous allez déployer une application frontale sur Amplify Hosting. Vous allez ensuite créer un backend à l'aide d'Amplify Studio. Enfin, vous allez connecter le backend cloud à l'application frontale.

Conditions préalables

Avant de commencer ce didacticiel, remplissez les conditions préalables suivantes.

Inscrivez-vous pour un Compte AWS

> Si vous n'êtes pas encore AWS client, vous devez en créer un Compte AWS en suivant les instructions en ligne. L'inscription vous permet d'accéder à Amplify et à d'autres AWS services que vous pouvez utiliser avec votre application.

Création d'un dépôt Git

Amplify prend en charge GitHub Bitbucket, et GitLab. AWS CodeCommit Transférez votre application vers votre dépôt Git.

Installation de l'interface de ligne de commande (CLI) Amplify

Pour obtenir des instructions, voir Installer la CLI Amplify dans la documentation Amplify Framework.

Étape 1 : Déployer un frontend

Si vous avez une application frontale existante dans un dépôt git que vous souhaitez utiliser pour cet exemple, vous pouvez suivre les instructions de déploiement d'une application frontale.

Si vous devez créer une nouvelle application frontale à utiliser pour cet exemple, vous pouvez suivre les instructions de création d'application React contenues dans la documentation de création d'application React.

Pour déployer une application frontale

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web dans le coin supérieur droit.
3. Sélectionnez votre fournisseur GitHub, Bitbucket ou de AWS CodeCommit dépôt GitLab, puis choisissez Continuer.
4. Amplify autorise l'accès à votre dépôt git. Pour les GitHub référentiels, Amplify utilise désormais la fonctionnalité Apps pour autoriser GitHub l'accès à Amplify.

   Pour plus d'informations sur l'installation et l'autorisation de l' GitHub application, consultezConfiguration de l'accès d'Amplify aux référentiels GitHub.
5. Sur la page Ajouter une branche de référentiel, procédez comme suit :

   a. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.
   b. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.
   c. Choisissez Suivant.
6. Sur la page Configurer les paramètres de build, choisissez Next.

7. Sur la page Révision, choisissez Enregistrer et déployer. Lorsque le déploiement est terminé, vous pouvez afficher votre application sur le domaine `amplifyapp.com` par défaut.

> **Note**
>
> Pour renforcer la sécurité de vos applications Amplify, le domaine amplifyapp.com est enregistré dans la liste des suffixes publics (PSL). Pour plus de sécurité, nous vous recommandons d'utiliser des cookies avec un `__Host-` préfixe si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Amplify. Cette pratique vous aidera à protéger votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Étape 2 : créer un backend

Maintenant que vous avez déployé une application frontale sur Amplify Hosting, vous pouvez créer un backend. Utilisez les instructions suivantes pour créer un backend avec une base de données simple et un point de terminaison d'API GraphQL.

Pour créer un backend

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, sélectionnez l'application que vous avez créée à l'étape 1.
3. Sur la page d'accueil de l'application, choisissez l'onglet Environnements principaux, puis choisissez Commencer. Cela lance le processus de configuration d'un environnement intermédiaire par défaut.
4. Une fois la configuration terminée, choisissez Launch Studio pour accéder à l'environnement principal de mise en scène dans Amplify Studio.

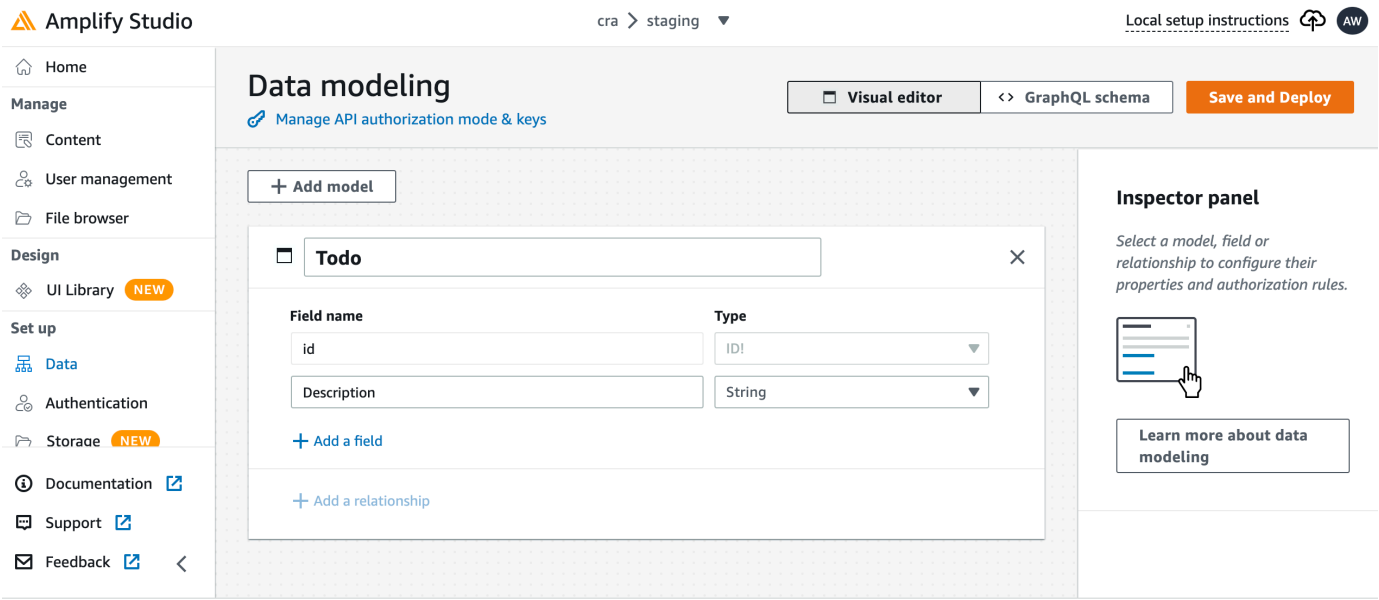
Amplify Studio est une interface visuelle permettant de créer et de gérer votre backend et d'accélérer le développement de votre interface utilisateur frontale. Pour plus d'informations sur Amplify Studio, consultez la documentation d'Amplify Studio.

Utilisez les instructions suivantes pour créer une base de données simple à l'aide de l'interface de création visuelle de backend d'Amplify Studio.

Création d'un modèle de données

1. Sur la page d'accueil de l'environnement intermédiaire de votre application, choisissez Create data model. Cela ouvre le concepteur de modèles de données.
2. Sur la page Modélisation des données, choisissez Ajouter un modèle.
3. Pour le titre, entrez**`Todo`**.
4. Choisissez Ajouter un champ.
5. Dans Nom du champ, entrez**`Description`**.

   La capture d'écran suivante est un exemple de l'apparence de votre modèle de données dans le concepteur.



6. Choisissez Enregistrer et déployer.
7. Retournez à la console Amplify Hosting et le déploiement de l'environnement de préparation sera en cours.

Pendant le déploiement, Amplify Studio crée toutes les AWS ressources requises dans le backend, notamment une API AWS AppSync GraphQL pour accéder aux données et une table Amazon DynamoDB pour héberger les éléments Todo. Amplify utilise CloudFormation pour déployer votre backend, ce qui vous permet de stocker votre définition de backend sous forme de. infrastructure-as-code

Étape 3 : Connectez le backend au frontend

Maintenant que vous avez déployé un frontend et créé un backend cloud contenant un modèle de données, vous devez les connecter. Utilisez les instructions suivantes pour transférer votre définition de backend vers votre projet d'application local à l'aide de la CLI Amplify.

Pour connecter un backend cloud à un frontend local

1. Ouvrez une fenêtre de terminal et accédez au répertoire racine de votre projet local.
2. Exécutez la commande suivante dans la fenêtre du terminal, en remplaçant le texte rouge par l'identifiant unique de l'application et le nom de l'environnement principal de votre projet.

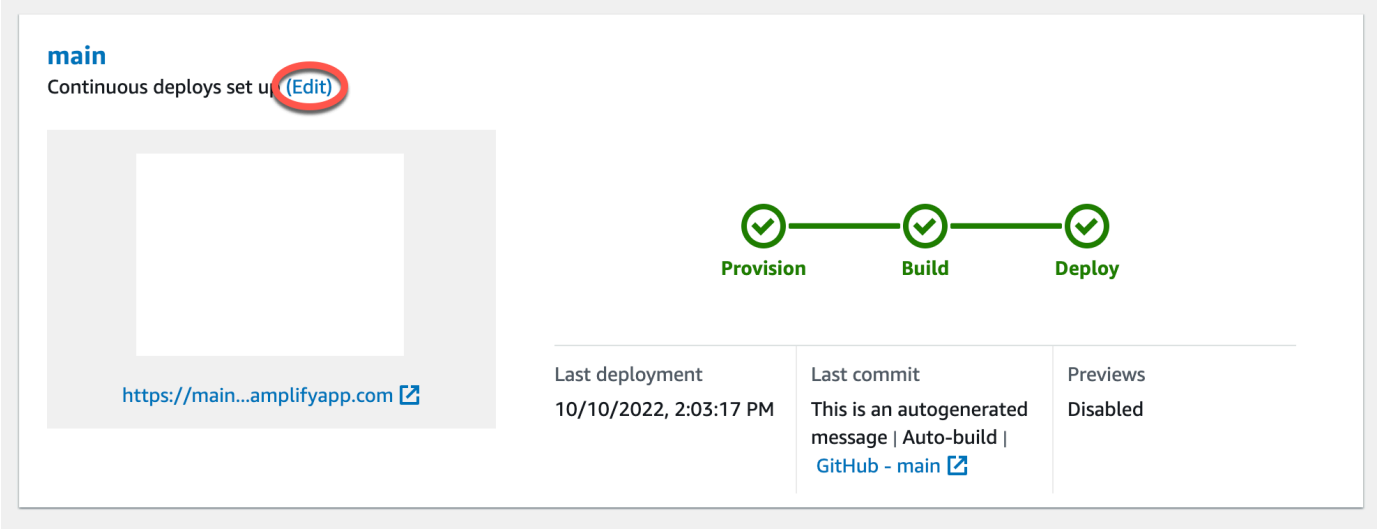
```
amplify pull --appId abcd1234 --envName staging
```

3. Suivez les instructions affichées dans la fenêtre du terminal pour terminer la configuration du projet.

Vous pouvez désormais configurer le processus de création pour ajouter le backend au flux de travail de déploiement continu. Suivez les instructions suivantes pour connecter une branche frontale à un backend dans la console Amplify Hosting.

Pour connecter une branche d'application frontale et un backend cloud

1. Sur la page d'accueil de l'application, choisissez l'onglet Environnements d'hébergement.
2. Localisez la branche principale et choisissez Modifier.



3. Dans la fenêtre Modifier le backend cible, pour Environnement, sélectionnez le nom du backend à connecter. Dans cet exemple, choisissez le backend de mise en scène que vous avez créé à l'étape 2.

   Par défaut, Full-Stack CI/CD est activé. Décochez cette option pour désactiver le full stack CI/CD pour ce backend. La désactivation de Full-Stack CI/CD entraîne l'exécution de l'application en mode pull uniquement. Au moment de la création, Amplify générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier votre environnement principal.
4. Ensuite, vous devez configurer un rôle de service pour accorder à Amplify les autorisations dont il a besoin pour apporter des modifications au backend de votre application. Vous pouvez utiliser un rôle de service existant ou en créer un nouveau. Pour obtenir des instructions, veuillez consulter Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend.
5. Après avoir ajouté un rôle de service, revenez à la fenêtre Modifier le backend cible et choisissez Enregistrer.
6. Pour terminer la connexion du backend intermédiaire à la branche principale de l'application frontale, effectuez une nouvelle version de votre projet.

   Effectuez l'une des actions suivantes :

   - Depuis votre dépôt git, envoyez du code pour lancer une compilation dans la console Amplify.
   - Dans la console Amplify, accédez à la page des détails de compilation de l'application et choisissez Redeploy this version.

Étapes suivantes

Configurer les déploiements de succursales de fonctionnalités

Suivez notre flux de travail recommandé pour configurer des déploiements de branches fonctionnelles avec plusieurs environnements principaux.

Création d'une interface utilisateur frontale dans Amplify Studio

Utilisez Studio pour créer votre interface utilisateur frontale à l'aide d'un ensemble de composants d' ready-to-useinterface utilisateur, puis connectez-la au backend de votre application. Pour plus d'informations et des didacticiels, consultez le guide de l'utilisateur d'Amplify Studio dans la documentation d'Amplify Framework.

Fonctionnalités de déploiement avancées

Ce chapitre couvre les fonctionnalités de déploiement avancées qui améliorent votre flux de travail Amplify Hosting. Ces fonctionnalités fournissent des contrôles et des fonctionnalités supplémentaires pour aider les équipes à gérer les déploiements plus efficacement, à garantir la qualité du code et à maintenir la sécurité tout au long du cycle de développement.

Découvrez comment protéger vos branches de fonctionnalités grâce à l'authentification par mot de passe afin de restreindre l'accès aux fonctionnalités inédites. Activez les aperçus Web pour les pull requests afin de consulter les modifications sur un aperçu unique URLs avant de fusionner le code avec les branches de production. Configurez end-to-end les tests à l'aide du framework Cypress pour détecter les régressions avant de mettre le code en production. Bien que la fonctionnalité du bouton Deploy to Amplify ne soit plus disponible, vous pouvez toujours facilement déployer des applications directement depuis votre référentiel à l'aide d'Amplify Hosting.

Rubriques

Restreindre l'accès aux branches d'une application Amplify

Si vous travaillez sur des fonctionnalités inédites, vous pouvez protéger par mot de passe les branches des fonctionnalités afin de restreindre l'accès à des utilisateurs spécifiques. Lorsque le contrôle d'accès est défini sur une branche, les utilisateurs sont invités à saisir un nom d'utilisateur et un mot de passe lorsqu'ils tentent d'accéder à l'URL de la branche.

Vous pouvez définir un mot de passe qui s'applique à une succursale individuelle ou globalement à toutes les succursales connectées. Lorsque le contrôle d'accès est activé à la fois au niveau de la succursale et au niveau global, le mot de passe au niveau de la succursale a priorité sur le mot de passe global (application).

Amplify limite les demandes échouées qui tentent d'accéder à des ressources protégées par mot de passe. Ce comportement protège les applications contre les attaques par dictionnaires ou autres tentatives de lecture de données via des contrôles d'accès.

Utilisez la procédure suivante pour définir un mot de passe afin de restreindre l'accès aux branches d'une application Amplify.

Pour définir des mots de passe pour les branches de fonctionnalités

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir les mots de passe des branches fonctionnelles.
3. Dans le volet de navigation, choisissez Hosting, puis Access control.
4. Dans la section Paramètres de contrôle d'accès, choisissez Gérer l'accès.
5. Sur la page Gérer le contrôle d'accès, effectuez l'une des opérations suivantes.
   - Pour définir un nom d'utilisateur et un mot de passe applicables à toutes les succursales connectées
     - Activez Gérer l'accès pour toutes les succursales. Par exemple, si des branches main, dev et feature sont connectées, vous pouvez appliquer le même nom d'utilisateur et le même mot de passe à toutes les branches.
   - Pour définir un nom d'utilisateur et un mot de passe applicables à une branche individuelle
     a. Désactivez Gérer l'accès pour toutes les succursales.
     b. Localisez la succursale que vous souhaitez gérer. Pour les paramètres d'accès, sélectionnez Mot de passe restreint requis.
     c. Dans Nom d'utilisateur, entrez un nom d'utilisateur.
     d. Dans Mot de passe, entrez un mot de passe.
   - Choisissez Enregistrer.
6. Si vous gérez le contrôle d'accès pour une application de rendu côté serveur (SSR), redéployez l'application en effectuant une nouvelle compilation à partir de votre dépôt Git. Cette étape est nécessaire pour permettre à Amplify d'appliquer vos paramètres de contrôle d'accès.

Aperçus Web pour les pull requests

Les aperçus Web permettent aux équipes de développement et d'assurance qualité (QA) de prévisualiser les modifications apportées par les pull requests (PRs) avant de fusionner le code dans une branche de production ou d'intégration. Les pull requests vous permettent d'informer les autres des modifications que vous avez apportées à une branche d'un référentiel. Après l'ouverture d'une

pull request, vous pouvez discuter et examiner les modifications potentielles avec les collaborateurs et ajouter des validations de suivi avant que vos modifications ne soient fusionnées dans la branche de base.

Un aperçu Web déploie chaque pull request envoyée à votre référentiel vers une URL de prévisualisation unique, complètement différente de l'URL utilisée par votre site principal. Pour les applications dont les environnements de backend sont provisionnés à l'aide de l'Amplify CLI ou d'Amplify Studio, chaque pull request (référentiels Git privés uniquement) crée un backend temporaire qui est supprimé à la fermeture du PR.

Lorsque les aperçus Web sont activés pour votre application, chaque PR est pris en compte dans le quota Amplify de 50 branches par application. Pour éviter de dépasser ce quota, assurez-vous de fermer votre PRs. Pour plus d'informations sur les quotas, consultez Quotas du service Amplify Hosting.

> **Note**
>
> Actuellement, la variable d'`AWS_PULL_REQUEST_ID`environnement n'est pas disponible lorsque vous l'utilisez en AWS CodeCommit tant que fournisseur de référentiel.

Sécurité de l'aperçu Web

Pour des raisons de sécurité, vous pouvez activer les aperçus Web sur toutes les applications dotées de référentiels privés, mais pas sur toutes les applications dotées de référentiels publics. Si votre référentiel Git est public, vous pouvez configurer des aperçus uniquement pour les applications qui ne nécessitent pas de rôle de service IAM. Par exemple, les applications dotées de backends et les applications déployées sur la plate-forme `WEB_COMPUTE` d'hébergement nécessitent un rôle de service IAM. Par conséquent, vous ne pouvez pas activer les aperçus Web pour ces types d'applications si leur référentiel est public. Amplify applique cette restriction pour empêcher des tiers de soumettre du code arbitraire qui serait exécuté en utilisant les autorisations de rôle IAM de votre application.

Lorsque les aperçus Web sont activés pour une application dans un référentiel public, avec un rôle SSR Compute, vous devez gérer avec soin les branches autorisées à accéder à ce rôle. Nous vous recommandons de ne pas utiliser de rôle au niveau de l'application. Vous devez plutôt associer un rôle de calcul au niveau de la branche. Cela vous permet d'accorder des autorisations uniquement aux succursales qui ont besoin d'accéder à des ressources spécifiques. Pour plus d'informations, consultez Ajout d'un rôle SSR Compute pour autoriser l'accès aux AWS ressources.

Activer les aperçus Web pour les pull requests

Pour les applications stockées dans un GitHub dépôt, les aperçus Web utilisent l' GitHub application Amplify pour accéder au dépôt. Si vous activez les aperçus Web sur une application Amplify existante que vous avez précédemment déployée à partir GitHub d'un dépôt OAuth utilisé pour y accéder, vous devez d'abord migrer l'application pour utiliser l'application Amplify. GitHub Pour les instructions de migration, voirMigration d'une OAuth application existante vers l'application Amplify GitHub.

Pour activer les aperçus Web pour les pull requests

1. Choisissez Hosting, puis Previews.

   > **Note**
   >
   > Les aperçus sont visibles dans le menu des paramètres de l'application uniquement lorsqu'une application est configurée pour un déploiement continu et connectée à un dépôt git. Pour obtenir des instructions sur ce type de déploiement, voir Commencer avec le code existant.

2. Pour les GitHub référentiels uniquement, procédez comme suit pour installer et autoriser l'application GitHub Amplify sur votre compte :

   a. Dans la fenêtre Installer GitHub l'application pour activer les aperçus, choisissez Installer GitHub l'application.
   b. Sélectionnez le GitHub compte sur lequel vous souhaitez configurer l'application Amplify GitHub.
   c. Une page s'ouvre sur GitHub.com pour configurer les autorisations de dépôt pour votre compte.
   d. Effectuez l'une des actions suivantes :
      - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
      - Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le dépôt de l'application pour laquelle vous activez les aperçus Web dans les référentiels que vous sélectionnez.
   e. Choisissez Enregistrer.

3. Après avoir activé les aperçus pour votre dépôt, revenez à la console Amplify pour activer les aperçus pour des branches spécifiques. Sur la page des aperçus, sélectionnez une branche dans la liste et choisissez Modifier les paramètres.

4. Sur la page Gérer les paramètres d'aperçu, activez les aperçus des demandes Pull. Ensuite, choisissez Valider.
5. Pour les applications Fullstack, effectuez l'une des opérations suivantes :
   - Choisissez, créez un nouvel environnement principal pour chaque Pull Request. Cette option vous permet de tester les modifications sans affecter la production.
   - Choisissez Pointer toutes les pull requests pour cette branche vers un environnement existant.
6. Choisissez Confirmer.

La prochaine fois que vous soumettrez une pull request pour la branche, Amplify crée et déploie votre PR vers une URL de prévisualisation. Une fois la pull request fermée, l'URL de prévisualisation est supprimée et tout environnement backend temporaire lié à la pull request est supprimé. Pour les GitHub référentiels uniquement, vous pouvez accéder à un aperçu de votre URL directement à partir de la pull request de votre GitHub compte.

Accès à l'aperçu Web avec des sous-domaines

Les aperçus Web pour les pull requests sont accessibles avec les sous-domaines d'une application Amplify connectée à un domaine personnalisé géré par Amazon Route 53. Lorsque la pull request est fermée, les branches et les sous-domaines associés à la pull request sont automatiquement supprimés. Il s'agit du comportement par défaut pour les aperçus Web une fois que vous avez configuré des déploiements de branches de fonctionnalités basés sur des modèles pour votre application. Pour obtenir des instructions sur la configuration de sous-domaines automatiques, consultezConfiguration de sous-domaines automatiques pour un domaine personnalisé Amazon Route 53.

Configuration des tests end-to-end Cypress pour votre application Amplify

Vous pouvez exécuter des tests end-to-end (E2E) pendant la phase de test de votre application Amplify pour détecter les régressions avant de mettre le code en production. La phase de test peut être configurée dans la spécification de construction YAML. Actuellement, vous ne pouvez exécuter que le framework de test Cypress pendant une compilation.

Cypress est un framework de test JavaScript basé sur lequel vous pouvez exécuter des tests E2E sur un navigateur. Pour un didacticiel expliquant comment configurer les tests E2E, consultez le billet de blog Running end-to-end Cypress tests for your fullstack deployment CI/CD with Amplify.

Ajouter des tests Cypress à une application Amplify existante

Vous pouvez ajouter des tests Cypress à une application existante en mettant à jour les paramètres de compilation de l'application dans la console Amplify. La spécification de construction YAML contient un ensemble de commandes de construction et de paramètres associés qu'Amplify utilise pour exécuter votre build. Utilisez cette `test` étape pour exécuter n'importe quelle commande de test au moment de la construction. Pour les tests E2E, Amplify Hosting propose une intégration plus approfondie avec Cypress qui vous permet de générer un rapport d'interface utilisateur pour vos tests.

La liste suivante décrit les paramètres de test et leur mode d'utilisation.

Prétest

Installez les dépendances requises pour exécuter les tests Cypress. Amplify Hosting utilise mochawesome pour générer un rapport afin de consulter les résultats de vos tests et d'attendre de configurer le serveur localhost pendant la construction.

test

Exécutez les commandes cypress pour effectuer des tests à l'aide de mochawesome.

Après le test

Le rapport mochawesome est généré à partir du JSON de sortie. Notez que si vous utilisez Yarn, vous devez exécuter cette commande en mode silencieux pour générer le rapport mochawesome. Pour Yarn, vous pouvez utiliser la commande suivante.

```
yarn run --silent mochawesome-merge cypress/report/mochawesome-report/
mochawesome*.json > cypress/report/mochawesome.json
```

Artefacts > Répertoire de base

Le répertoire à partir duquel les tests sont exécutés.

artéfacts> configFilePath

Les données du rapport de test générées.

artéfacts > fichiers

Les artefacts générés (captures d'écran et vidéos) sont disponibles au téléchargement.

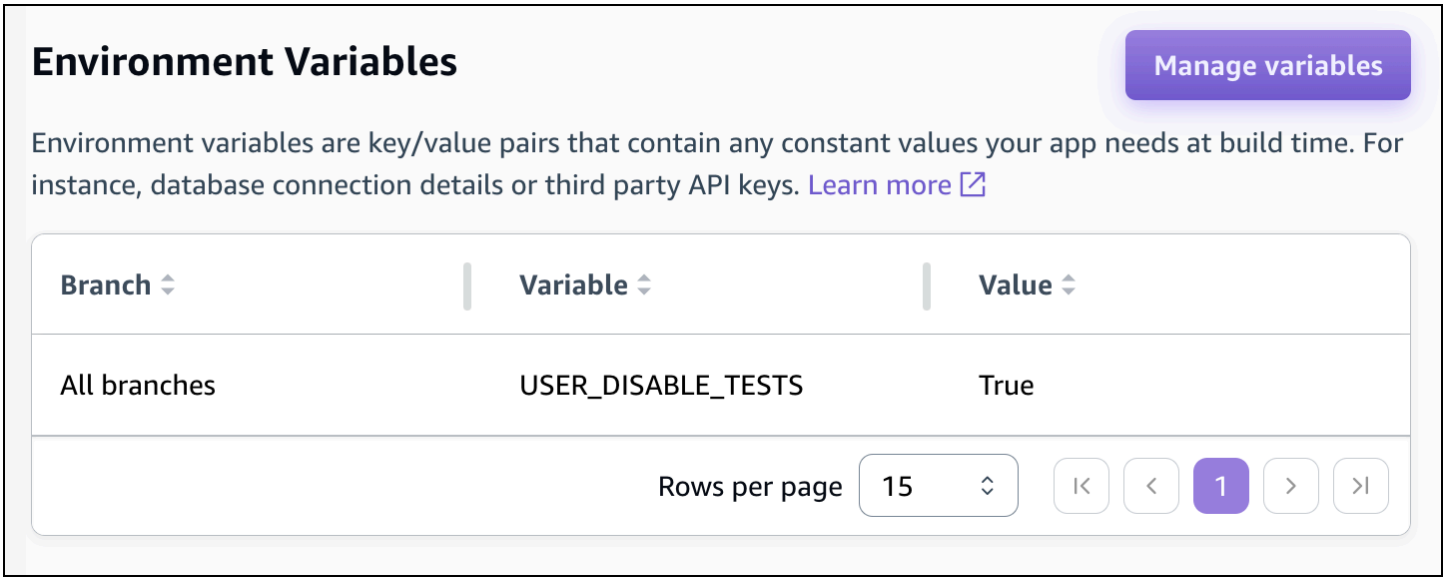
L'exemple d'extrait suivant d'un `amplify.yml` fichier de spécification de build montre comment ajouter des tests Cypress à votre application.

```
test:
  phases:
    preTest:
      commands:
        - npm ci
        - npm install -g pm2
        - npm install -g wait-on
        - npm install mocha mochawesome mochawesome-merge mochawesome-report-generator
        - pm2 start npm -- start
        - wait-on http://localhost:3000
    test:
      commands:
        - 'npx cypress run --reporter mochawesome --reporter-options
 "reportDir=cypress/report/mochawesome-
report,overwrite=false,html=false,json=true,timestamp=mmddyyyy_HHMMss"'
    postTest:
      commands:
        - npx mochawesome-merge cypress/report/mochawesome-report/mochawesome*.json >
 cypress/report/mochawesome.json
        - pm2 kill
  artifacts:
    baseDirectory: cypress
    configFilePath: '**/mochawesome.json'
    files:
      - '**/*.png'
      - '**/*.mp4'
```

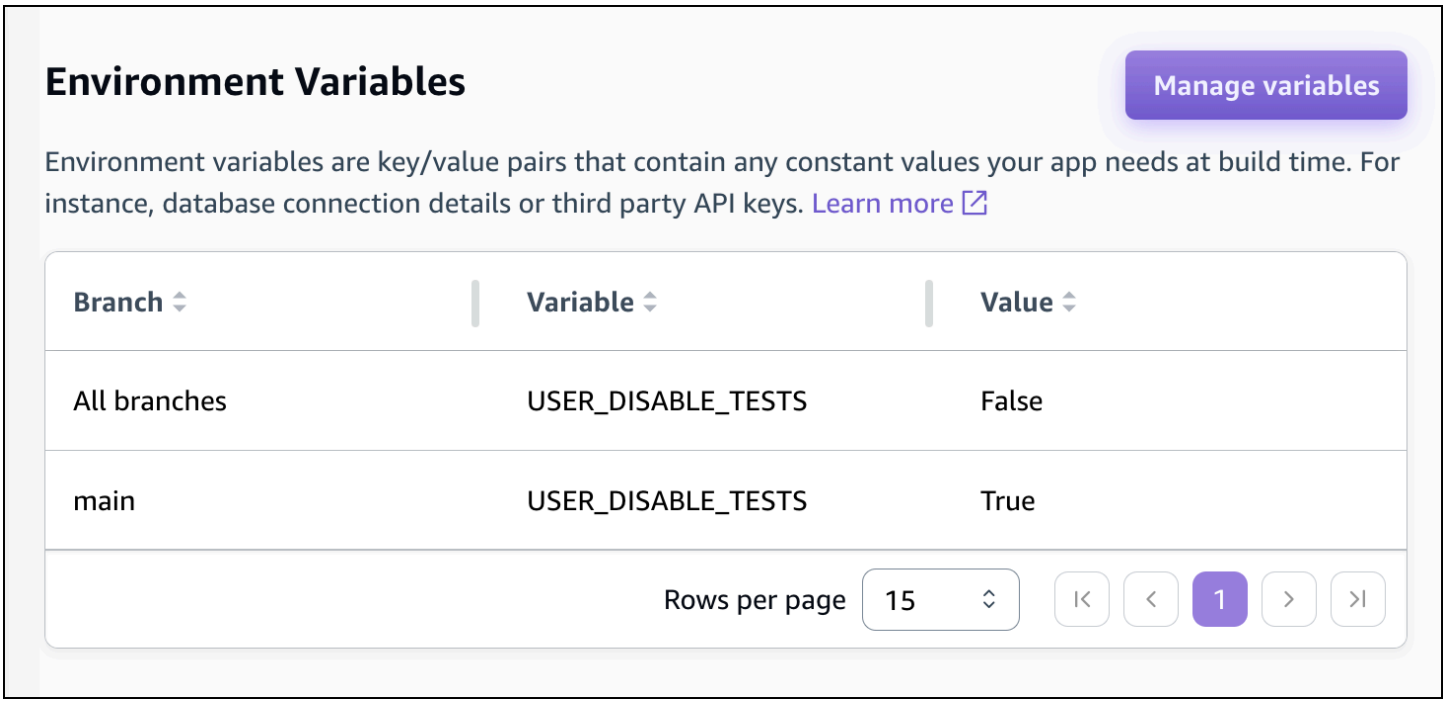
Désactiver les tests pour une application ou une branche Amplify

Une fois que la configuration de test a été ajoutée à vos paramètres de `amplify.yml` build, l'`test`étape s'exécute pour chaque build, sur chaque branche. Si vous souhaitez désactiver globalement l'exécution des tests, ou uniquement exécuter des tests pour des branches spécifiques, vous pouvez utiliser la variable d'USER_DISABLE_TESTSenvironnement sans modifier vos paramètres de compilation.

Pour désactiver globalement les tests pour toutes les branches, ajoutez la variable d'USER_DISABLE_TESTSenvironnement avec une valeur de `true` pour toutes les branches. La capture d'écran suivante montre la section des variables d'environnement de la console Amplify avec les tests désactivés pour toutes les branches.



Pour désactiver les tests pour une branche spécifique, ajoutez la variable d'USER_DISABLE_TESTSenvironnement avec une valeur de `false` pour toutes les branches, puis ajoutez une substitution pour chaque branche que vous souhaitez désactiver avec une valeur de`true`. Dans la capture d'écran suivante, les tests sont désactivés sur la branche principale et activés pour toutes les autres branches.



La désactivation des tests avec cette variable entraînera l'annulation totale de l'étape de test lors d'une compilation. Pour réactiver les tests, définissez cette valeur sur `false` ou supprimez la variable d'environnement.

Utilisation du bouton Deploy to Amplify pour partager un projet GitHub

> **Important**
>
> Le déploiement en un clic à l'aide du bouton Deploy to Amplify Hosting n'est plus disponible. Pour effectuer un déploiement à partir d'un référentiel, créez une nouvelle application dans Amplify Hosting. Pour obtenir des instructions, veuillez consulter Commencer à déployer une application sur Amplify Hosting.

Le bouton Deploy to Amplify Hosting vous permet de partager des GitHub projets publiquement ou au sein de votre équipe. Voici une image du bouton :

Ajouter le bouton Deploy to Amplify Hosting à un référentiel ou à un blog

Ajoutez le bouton à votre fichier GitHub README.md, à votre billet de blog ou à toute autre page de balisage qui affiche du code HTML. Le bouton comporte les deux éléments suivants :

1. Une image SVG située à l'URL `https://oneclick.amplifyapp.com/button.svg`
2. L'URL de la console Amplify avec un lien vers votre GitHub référentiel. Vous pouvez soit copier l'URL de votre dépôt, par exemple`https://github.com/username/repository`, soit fournir un lien profond vers un dossier spécifique, tel que`https://github.com/username/repository/tree/branchname/folder`. Amplify Hosting déploiera la branche par défaut dans votre référentiel. D'autres branches pourront être connectées une fois que l'application sera connectée.

Utilisez l'exemple suivant pour ajouter le bouton à un fichier Markdown, tel que votre fichier GitHub README.md. Remplacez `https://github.com/username/repository` par l'URL de votre dépôt.

```
[![amplifybutton](https://oneclick.amplifyapp.com/button.svg)](https://
console.aws.amazon.com/amplify/home#/deploy?repo=https://github.com/username/
repository)
```

Utilisez l'exemple suivant pour ajouter le bouton à n'importe quel document HTML. Remplacez `https://github.com/username/repository` par l'URL de votre dépôt.

```
<a href="https://console.aws.amazon.com/amplify/home#/deploy?repo=https://github.com/
username/repository">
    <img src="https://oneclick.amplifyapp.com/button.svg" alt="Deploy to Amplify
 Hosting">
</a>
```

Configuration des redirections et des réécritures pour une application Amplify

Les redirections permettent à un serveur web de réacheminer la navigation d'une URL vers une autre. Les redirections sont souvent utilisées pour personnaliser l'apparence d'une URL, pour éviter les liens brisés, pour déplacer l'emplacement d'hébergement d'une application ou d'un site sans modifier son adresse et pour remplacer l'URL demandée par le formulaire requis par une application Web.

Comprendre les redirections prises en charge par Amplify

Amplify prend en charge les types de redirection suivants dans la console.

Redirection permanente (301)

Les redirections 301 sont conçues pour les modifications durables apportées à la destination d'une adresse web. L'historique de classement du moteur de recherche de l'adresse d'origine s'applique à la nouvelle adresse de destination. La redirection se produit côté client, une barre de navigation de navigateur affiche ainsi l'adresse de destination après la redirection.

Les raisons courantes d'utilisation de redirections 301 incluent les suivantes :

- Pour éviter un lien brisé lorsque l'adresse d'une page change.
- Pour éviter un lien brisé lorsqu'un utilisateur fait une faute de frappe prévisible dans une adresse.

Redirection temporaire (302)

Les redirections 302 sont conçues pour les modifications temporaires apportées à la destination d'une adresse web. L'historique de classement de l'adresse d'origine dans les moteurs de recherche ne s'applique pas à la nouvelle adresse de destination. La redirection se produit côté client, une barre de navigation de navigateur affiche ainsi l'adresse de destination après la redirection.

Les raisons courantes d'utilisation de redirections 302 incluent les suivantes :

- Pour fournir une destination de détour lorsque des réparations sont effectuées sur l'adresse d'origine.
- Fournir des pages de test pour la A/B comparaison d'une interface utilisateur.

> **Note**
>
> Si votre application renvoie une réponse 302 inattendue, l'erreur est probablement due aux modifications que vous avez apportées à la redirection et à la configuration personnalisée de l'en-tête de votre application. Pour résoudre ce problème, vérifiez que vos en-têtes personnalisés sont valides, puis réactivez la règle de réécriture 404 par défaut pour votre application.

Réécriture (200)

Les redirections 200 (réécritures) sont conçues pour afficher le contenu de l'adresse de destination comme s'il était servi à partir de l'adresse d'origine. L'historique de classement du moteur de recherche continue à s'appliquer à l'adresse d'origine. La redirection se produit côté serveur, une barre de navigation de navigateur affiche ainsi l'adresse d'origine après la redirection. Les raisons courantes d'utilisation de redirections 200 incluent les suivantes :

- Pour rediriger l'ensemble d'un site vers un nouvel emplacement d'hébergement sans modifier l'adresse du site.
- Pour rediriger l'ensemble du trafic vers une application web monopage (SPA) vers sa page index.html pour traitement par une fonction de routeur côté client.

Introuvable (404)

Les redirections 404 se produisent lorsqu'une demande pointe vers une adresse qui n'existe pas. La page de destination d'une redirection 404 s'affiche au lieu de celle demandée. Les raisons courantes d'une redirection 404 incluent les suivantes :

- Pour éviter un message de lien brisé lorsqu'un utilisateur saisit une URL incorrecte.
- Pour pointer des requêtes de pages inexistantes d'une application web vers sa page index.html pour traitement par une fonction de routeur côté client.

Comprendre l'ordre des redirections

Les redirections sont appliquées du haut de la liste vers le bas. Assurez-vous que votre ordre donne l'effet voulu. Par exemple, avec l'ordre de redirections suivant, toutes les requêtes d'un chemin donné

sous /docs/ sont redirigées vers le même chemin sous /documents/, sauf /docs/specific-filename.html qui redirige vers /documents/different-filename.html :

```
/docs/specific-filename.html /documents/different-filename.html 301
/docs/<*> /documents/<*>
```

L'ordre de redirections suivant ignore la redirection de specific-filename.html vers different-filename.html :

```
/docs/<*> /documents/<*>
/docs/specific-filename.html /documents/different-filename.html 301
```

Comprendre comment Amplify transmet les paramètres de requête

Vous pouvez utiliser les paramètres de requête pour mieux contrôler vos correspondances d'URL. Amplify transmet tous les paramètres de requête vers le chemin de destination pour les redirections 301 et 302, avec les exceptions suivantes :

- Si l'adresse d'origine inclut une chaîne de requête définie sur une valeur spécifique, Amplify ne transmet pas les paramètres de requête. Dans ce cas, la redirection s'applique uniquement aux demandes adressées à l'URL de destination avec la valeur de requête spécifiée.
- Si l'adresse de destination de la règle correspondante comporte des paramètres de requête, ceux-ci ne sont pas transférés. Par exemple, si l'adresse de destination de la redirection est`https://example-target.com?q=someParam`, les paramètres de requête ne sont pas transmis.

Création et modification de redirections dans la console Amplify

Vous pouvez créer et modifier des redirections pour une application dans la console Amplify. Avant de commencer, vous aurez besoin des informations suivantes concernant les différentes parties d'une redirection.

Une adresse originale

Adresse demandée par l'utilisateur.

Une adresse de destination

L'adresse qui diffuse réellement le contenu que l'utilisateur voit.

Un type de redirection

Les types incluent une redirection permanente (301), une redirection temporaire (302), une réécriture (200) ou une redirection introuvable (404).

Un code de pays à deux lettres (facultatif)

Une valeur que vous pouvez inclure pour segmenter l'expérience utilisateur de votre application par région géographique.

Pour créer une redirection dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez créer une redirection.
3. Dans le volet de navigation, choisissez Hosting, puis Rewrites and redirections.
4. Sur la page Réécritures et redirections, choisissez Gérer les redirections.
5. Ajoutez ou mettez à jour manuellement des redirections dans l'éditeur JSON de réécritures et de redirections.
   a. Pour`source`, spécifiez l'adresse d'origine demandée par l'utilisateur.
   b. Pour`status`, spécifiez le type de redirection.
   c. Pour`target`, spécifiez l'adresse de destination qui affiche le contenu à l'utilisateur.
   d. (Facultatif) Pour`condition`, entrez une condition de code de pays à deux lettres.
6. Choisissez Enregistrer.

Exemple de référence pour les redirections et les réécritures

Cette section fournit des exemples de divers scénarios de redirection courants.

> **Important**
>
> Les redirections spécifiques à un domaine ne prennent pas en charge les composants de chemin dans le champ source.
> Pris en charge :
>
> - `"source": "https://example.com"` (chemins ajoutés automatiquement)

Non pris en charge :

- `"source": "https://example.com/specific-path"`
- Les règles comportant des `domain+path` combinaisons ne sont actuellement pas prises en charge.

Motifs alternatifs
Pour les redirections de chemin spécifiques à un domaine, utilisez :

1. Règles distinctes pour les domaines uniquement (les chemins sont automatiquement ajoutés)
2. Règles relatives aux chemins uniquement avec logique conditionnelle
3. Combinaisons de règles multiples

Vous pouvez utiliser ces exemples pour comprendre la syntaxe JSON permettant de créer vos propres redirections et réécritures dans l'éditeur JSON de la console Amplify.

> **Note**
>
> La correspondance du domaine d'adresse d'origine ne fait pas la distinction majuscules/minuscules.

Rubriques

Redirections et réécritures simples

Vous pouvez utiliser l'exemple suivant pour rediriger définitivement une page spécifique vers une nouvelle adresse.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/original.html`	`/destination.html`	`permanent redirect (301)`	

Format JSON

```
[
  {
    "source": "/original.html",
    "status": "301",
    "target": "/destination.html",
    "condition": null
  }
]
```

Vous pouvez utiliser l'exemple suivant pour rediriger n'importe quel chemin situé dans un dossier vers le même chemin dans un autre dossier.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<*>`	`/documents/<*>`	`permanent redirect (301)`	

Format JSON

```
[
  {
    "source": "/docs/<*>",
    "status": "301",
    "target": "/documents/<*>",
```

```
    "condition": null
  }
]
```

Vous pouvez utiliser l'exemple suivant pour rediriger tout le trafic vers index.html sous forme de réécriture. Dans ce scénario, la réécriture permet d'indiquer à l'utilisateur qu'il a accédé à l'adresse d'origine.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/<*>`	`/index.html`	`rewrite (200)`	

Format JSON

```
[
  {
    "source": "/<*>",
    "status": "200",
    "target": "/index.html",
    "condition": null
  }
]
```

Vous pouvez utiliser l'exemple suivant pour utiliser une réécriture afin de modifier le sous-domaine qui apparaît à l'utilisateur.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`https://mydomain.com`	`https://www.mydomain.com`	`rewrite (200)`	

Format JSON

```
[
```

```
  {
    "source": "https://mydomain.com",
    "status": "200", "target": "https://www.mydomain.com",
    "condition": null
  }
]
```

Vous pouvez utiliser l'exemple suivant pour rediriger vers un autre domaine avec un préfixe de chemin.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`https://mydomain.com`	`https://www.mydomain.com/documents`	`temporary redirect (302)`	

Format JSON

```
[
  {
    "source": "https://mydomain.com",
    "status": "302",
    "target": "https://www.mydomain.com/documents/",
    "condition": null
  }
]
```

Vous pouvez utiliser l'exemple suivant pour rediriger les chemins d'un dossier introuvable vers une page 404 personnalisée.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/<*>`	`/404.html`	`not found (404)`	

Format JSON

```
[
  {
    "source": "/<*>",
    "status": "404",
    "target": "/404.html",
    "condition": null
  }
]
```

Important

Les composants de chemin dans les règles source basées sur le domaine (tels que`"https://domain.com/path"`) ne sont pas pris en charge et entraîneront l'ignorance de la règle sans erreur.

Redirections pour les applications Web à page unique (SPA)

La plupart des frameworks SPA prennent en charge HTML5 History.pushState () pour modifier l'emplacement du navigateur sans lancer de requête au serveur. Cela fonctionne pour les utilisateurs qui commencent leur transition à partir de la racine (ou /index.html), mais échoue pour les utilisateurs qui accèdent directement à une autre page.

L'exemple suivant utilise des expressions régulières pour configurer une réécriture 200 pour tous les fichiers dans index.html, à l'exception des extensions de fichier spécifiées dans l'expression régulière.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`</^[^.]+$\|\.(?!(css\|gif\|ico\|jpg\|js\|png\|txt\|svg\|woff\|woff2\|ttf\|map\|json\|webp)$)([^.]+$)/>`	`/index.html`	`200`	

Format JSON

```
[
  {
    "source": "</^[^.]+$|\.(?!(css|gif|ico|jpg|js|png|txt|svg|woff|woff2|ttf|map|json|
webp)$)([^.]+$)/>",
    "status": "200",
    "target": "/index.html",
    "condition": null
  }
]
```

Réécriture du proxy inversé

L'exemple suivant utilise une réécriture pour transférer du contenu provenant d'un autre emplacement afin que l'utilisateur sache que le domaine n'a pas changé. Le protocole HTTPS est le seul protocole pris en charge pour les proxys inverses.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/images/<*>`	`https://images.otherdomain.com/<*>`	`rewrite (200)`	

Format JSON

```
[
  {
    "source": "/images/<*>",
    "status": "200",
    "target": "https://images.otherdomain.com/<*>",
    "condition": null
  }
]
```

Trailing slashes and clean URLs

Pour créer des structures d'URL propres telles que about au lieu de about.html, des générateurs sur site statiques tels que Hugo génèrent des répertoires pour les pages avec un index.html (/about/index.html). Amplify crée automatiquement un nettoyage URLs en ajoutant une barre oblique lorsque cela est nécessaire. Le tableau ci-dessous présente différents scénarios :

Entrées utilisateur dans le navigateur	URL dans la barre d'adresse	Document affiché
`/about`	`/about`	`/about.html`
`/about (when about.html returns 404)`	`/about/`	`/about/index.html`
`/about/`	`/about/`	`/about/index.html`

Espaces réservés

Vous pouvez utiliser l'exemple suivant pour rediriger les chemins d'une structure de dossiers vers une structure correspondante dans un autre dossier.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<year>/<month>/<date>/<itemid>`	`/documents/<year>/<month>/<date>/<itemid>`	`permanent redirect (301)`	

Format JSON

```
[
  {
    "source":  "/docs/<year>/<month>/<date>/<itemid>",
    "status": "301",
    "target": "/documents/<year>/<month>/<date>/<itemid>",
```

```
            "condition": null
    }
]
```

Chaînes de requête et paramètres de chemin

> **Warning**
>
> N'incluez pas de secrets, d'informations d'identification ou de données sensibles dans URLs les paramètres de chemin ou de requête. Ces valeurs sont consultables en texte brut dans les journaux d'accès de votre application Amplify.

Vous pouvez utiliser l'exemple suivant pour rediriger un chemin vers un dossier dont le nom correspond à la valeur d'un élément de chaîne de requête figurant dans l'adresse d'origine :

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs?id=<my-blog-id-value`	`/documents/<my-blog-post-id-value>`	`permanent redirect (301)`	

Format JSON

```
[
  {
    "source": "/docs?id=<my-blog-id-value>",
    "status": "301",
    "target": "/documents/<my-blog-id-value>",
    "condition": null
  }
]
```

> **Note**
>
> Amplify transmet tous les paramètres de chaîne de requête au chemin de destination pour les redirections 301 et 302. Toutefois, si l'adresse d'origine inclut une chaîne de requête définie

sur une valeur spécifique, comme illustré dans cet exemple, Amplify ne transmet pas les paramètres de requête. Dans ce cas, la redirection s'applique uniquement aux demandes adressées à l'adresse de destination avec la valeur de requête spécifiée`eid`.

Vous pouvez utiliser l'exemple suivant pour rediriger tous les chemins introuvables à un niveau donné d'une structure de dossiers vers index.html dans un dossier spécifique.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/documents/<folder>/<child-folder>/<grand-child-folder>`	`/documents/index.html`	`not found (404)`	

Format JSON

```
[
  {
    "source": "/documents/<x>/<y>/<z>",
    "status": "404",
    "target": "/documents/index.html",
    "condition": null
  }
]
```

Redirections basées sur les régions

Vous pouvez utiliser l'exemple suivant pour rediriger les demandes en fonction de la région.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/documents`	`/documents/us/`	`temporary redirect (302)`	`<US>`

Format JSON

```
[
  {
    "source": "/documents",
    "status": "302",
    "target": "/documents/us/",
    "condition": "<US>"
  }
]
```

Utilisation d'expressions génériques dans les redirections et les réécritures

Vous pouvez utiliser l'expression générique<*>, dans l'adresse d'origine pour une redirection ou une réécriture. Vous devez placer l'expression à la fin de l'adresse d'origine et elle doit être unique. Amplify ignore les adresses d'origine qui incluent plusieurs expressions génériques ou les utilise à un emplacement différent.

Voici un exemple de redirection valide avec une expression générique.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<*>`	`/documents/<*>`	`permanent redirect (301)`	

Les deux exemples suivants illustrent des redirections non valides avec des expressions génériques.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<*>/content`	`/documents/<*>/content`	`permanent redirect (301)`	
`/docs/<*>/content/<*>`	`/documents/<*>/content/<*>`	`permanent redirect (301)`	

Utilisation de variables d'environnement dans une application Amplify

Les variables d'environnement sont des paires clé-valeur que vous pouvez ajouter aux paramètres de votre application pour les mettre à la disposition d'Amplify Hosting. La meilleure pratique consiste à utiliser des variables d'environnement pour exposer les données de configuration des applications. Toutes les variables d'environnement que vous ajoutez sont chiffrées pour empêcher tout accès non autorisé.

Amplify applique les contraintes suivantes aux variables d'environnement que vous créez.

- Amplify ne vous permet pas de créer des noms de variables d'environnement avec un `AWS` préfixe. Ce préfixe est réservé à un usage interne d'Amplify uniquement.
- La valeur d'une variable d'environnement ne peut pas dépasser 5 500 caractères.

> **Important**
>
> N'utilisez pas de variables d'environnement pour stocker des secrets. Pour une application Gen 2, utilisez la fonction de gestion des secrets de la console Amplify. Pour plus d'informations, consultez la section Secrets et variables d'environnement dans la documentation Amplify. Pour une application Gen 1, stockez les secrets dans un secret d'environnement créé à l'aide du AWS Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Gestion des secrets environnementaux.

Amplifier la référence des variables d'environnement

Les variables d'environnement suivantes sont accessibles par défaut dans la console Amplify.

Nom de la variable	Description	Exemple de valeur
_BUILD_TIMEOUT	Durée du délai de compilation en minutes. La valeur minimale est 5.	`30`

Nom de la variable	Description	Exemple de valeur
	La valeur maximale est de 120.	
_LIVE_UPDATES	L'outil sera mis à niveau vers la dernière version.	`[{"name":"Amplify CLI","pkg":"@aws-amplify/cli","type":"npm","version":"latest"}]`
USER_DISABLE_TESTS	L'étape de test est ignorée lors d'une génération. Vous pouvez désactiver les tests pour toutes les branches ou pour des branches spécifiques d'une application. Cette variable d'environnement est utilisée pour les applications qui effectuent des tests pendant la phase de création. Pour plus d'informations sur la définition de cette variable, consultezDésactiver les tests pour une application ou une branche Amplify.	`true`
AWS_APP_ID	ID d'application du build actuel	`abcd1234`
AWS_BRANCH	Nom de branche du build actuel	`main`, `develop`, `beta`, `v2.0`
AWS_BRANCH_ARN	La branche Amazon Resource Name (ARN) de la version actuelle	`aws:arn:amplify:us-west-2:123456789012:appname/branch/...`

Nom de la variable	Description	Exemple de valeur
AWS_CLONE_URL	URL clone utilisée pour extraire le contenu du référentiel git	`git@github.com:<user-name>/<repo-name>.git`
AWS_COMMIT_ID	L'ID de validation de la version actuelle « HEAD » pour les reconstructions	`abcd1234`
AWS_JOB_ID	ID de tâche du build actuel. Cela inclut un rembourrage de « 0 » afin qu'il ait toujours la même longueur.	`0000000001`
AWS_PULL_REQUEST_ID	ID de pull request de la version préliminaire Web de pull request. Cette variable d'environnement n'est pas disponible lorsque vous l'utilisez en AWS CodeCommit tant que fournisseur de référentiel.	`1`
AWS_PULL_REQUEST_SOURCE_BRANCH	Nom de la branche de fonctionnalité pour un aperçu de pull request soumis à une branche d'application dans la console Amplify.	`featureA`
AWS_PULL_REQUEST_DESTINATION_BRANCH	Nom de la branche d'application de la console Amplify à laquelle une pull request de branche de fonctionnalité est soumise.	`main`

Nom de la variable	Description	Exemple de valeur
AMPLIFY_AMAZON_CLIENT_ID	L'identifiant du client Amazon	`123456`
AMPLIFY_AMAZON_CLIENT_SECRET	Le secret du client Amazon	`example123456`
AMPLIFY_FACEBOOK_CLIENT_ID	L'identifiant du client Facebook	`123456`
AMPLIFY_FACEBOOK_CLIENT_SECRET	Le secret du client Facebook	`example123456`
AMPLIFY_GOOGLE_CLIENT_ID	L'identifiant du client Google	`123456`
AMPLIFY_GOOGLE_CLIENT_SECRET	Le secret du client de Google	`example123456`
AMPLIFY_DIFF_DEPLOY	Activez ou désactivez le déploiement frontal basé sur les différences. Pour de plus amples informations, veuillez consulter Configuration de la création et du déploiement d'un frontend basé sur Diff.	`true`
AMPLIFY_DIFF_DEPLOY_ROOT	Le chemin à utiliser pour les comparaisons de déploiements frontaux basées sur les différences, par rapport à la racine de votre référentiel.	`dist`

Nom de la variable	Description	Exemple de valeur
AMPLIFY_DIFF_BACKEND	Activez ou désactivez les versions de backend basées sur les différences. Cela s'applique uniquement aux applications Gen 1. Pour de plus amples informations, consultez Configuration de versions de backend basées sur les différences pour une application de première génération.	`true`
AMPLIFY_BACKEND_PULL_ONLY	Amplify gère cette variable d'environnement. Cela s'applique uniquement aux applications Gen 1. Pour de plus amples informations, consultez Modifier un frontend existant pour qu'il pointe vers un autre backend.	`true`
AMPLIFY_BACKEND_APP_ID	Amplify gère cette variable d'environnement. Cela s'applique uniquement aux applications Gen 1. Pour de plus amples informations, consultez Modifier un frontend existant pour qu'il pointe vers un autre backend.	`abcd1234`

Nom de la variable	Description	Exemple de valeur
AMPLIFY_SKIP_BACKEND_BUILD	Si votre spécification de build ne contient aucune section de backend et que vous souhaitez désactiver les builds de backend, définissez cette variable d'environnement sur. `true` Cela s'applique uniquement aux applications Gen 1.	`true`
AMPLIFY_ENABLE_DEBUG_OUTPUT	Définissez cette variable sur `true` pour imprimer une trace de pile dans les journaux. Cela est utile pour corriger les erreurs de compilation du backend.	`true`
AMPLIFY_MONOREPO_APP_ROOT	Le chemin à utiliser pour spécifier la racine d'une application monorepo, par rapport à la racine de votre dépôt.	`apps/react-app`
AMPLIFY_USERPOOL_ID	L'ID du groupe d'utilisateurs Amazon Cognito importé pour l'authentification	`us-west-2_example`

Nom de la variable	Description	Exemple de valeur
AMPLIFY_WEBCLIENT_ID	L'ID du client d'application à utiliser par les applications Web Le client de l'application doit être configuré pour accéder au groupe d'utilisateurs Amazon Cognito spécifié par la variable d'environnement AMPLIFY_USERPOOL_ID.	`123456`
AMPLIFY_NATIVECLIENT_ID	L'ID du client d'application à utiliser par les applications natives Le client de l'application doit être configuré pour accéder au groupe d'utilisateurs Amazon Cognito spécifié par la variable d'environnement AMPLIFY_USERPOOL_ID.	`123456`
AMPLIFY_IDENTITYPOOL_ID	L'ID du pool d'identités Amazon Cognito	`example-identitypool-id`
AMPLIFY_PERMISSIONS_BOUNDARY_ARN	L'ARN que la politique IAM doit utiliser comme limite d'autorisations qui s'applique à tous les rôles IAM créés par Amplify.	`arn:aws:iam::123456789012:policy/example-policy`

Nom de la variable	Description	Exemple de valeur
AMPLIFY_DESTRUCTIVE_UPDATES	Définissez cette variable d'environnement sur true pour permettre à une API GraphQL d'être mise à jour avec des opérations de schéma susceptibles de provoquer une perte de données.	`true`

> **Note**
>
> Les variables d'`AMPLIFY_AMAZON_CLIENT_SECRET`environnement `AMPLIFY_AMAZON_CLIENT_ID` et sont OAuth des jetons, et non une clé d' AWS accès ou une clé secrète.

Variables d'environnement du framework frontal

Si vous développez votre application avec un framework frontal qui prend en charge ses propres variables d'environnement, il est important de comprendre que celles-ci ne sont pas identiques aux variables d'environnement que vous configurez dans la console Amplify. Par exemple, React (préfixé REACT_APP) et Gatsby (préfixé GATSBY) vous permettent de créer des variables d'environnement d'exécution que ces frameworks intègrent automatiquement dans la version de production de votre frontend. Pour comprendre les effets de l'utilisation de ces variables d'environnement pour stocker des valeurs, reportez-vous à la documentation du framework d'interface que vous utilisez.

Le stockage de valeurs sensibles, telles que les clés d'API, dans ces variables d'environnement préfixées par le framework frontal n'est pas une bonne pratique et est fortement déconseillé.

Définition de variables d'environnement

Utilisez les instructions suivantes pour définir les variables d'environnement d'une application dans la console Amplify.

> **Note**
>
> Les variables d'environnement ne sont visibles dans le menu des paramètres de l'application de la console Amplify que lorsqu'une application est configurée pour un déploiement continu et connectée à un référentiel git. Pour obtenir des instructions sur ce type de déploiement, voir Commencer avec le code existant.

Pour définir des variables d'environnement

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Dans la console Amplify, choisissez Hosting, puis choisissez Environment variables.
3. Sur la page Variables d'environnement, sélectionnez Gérer les variables.
4. Pour Variable, entrez votre clé. Dans Valeur, entrez votre valeur. Par défaut, Amplify applique les variables d'environnement à toutes les branches, de sorte que vous n'avez pas à saisir à nouveau les variables lorsque vous connectez une nouvelle branche.
5. (Facultatif) Pour personnaliser une variable d'environnement spécifiquement pour une branche, ajoutez une dérogation de branche comme suit :

   a. Choisissez Actions, puis sélectionnez Ajouter un remplacement de variable.

   b. Vous disposez maintenant d'un ensemble de variables d'environnement spécifique à votre branche.
6. Choisissez Enregistrer.

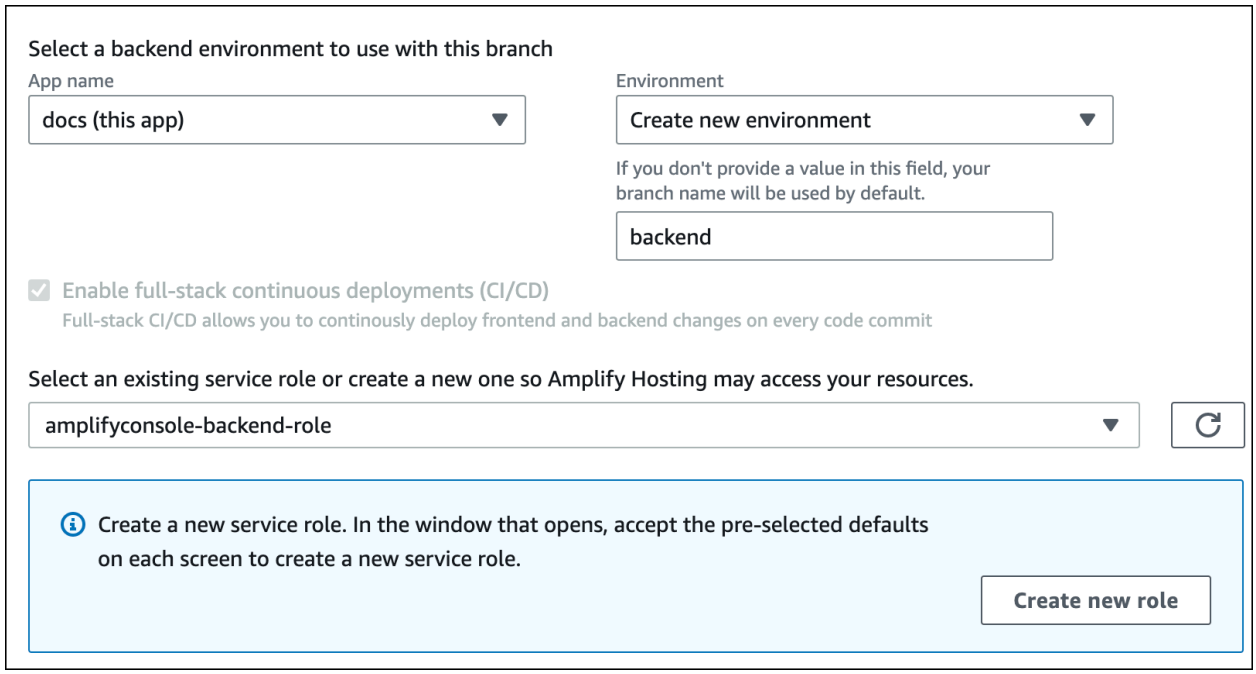
Création d'un nouvel environnement principal avec des paramètres d'authentification pour la connexion sociale

Pour connecter une succursale à une application

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. La procédure de connexion d'une branche à une application varie selon que vous connectez une succursale à une nouvelle application ou à une application existante.
   - Connecter une succursale à une nouvelle application

     a. Sur la page des paramètres de génération, recherchez la section Sélectionnez un environnement principal à utiliser avec cette branche. Pour Environnement, choisissez

Créer un nouvel environnement et entrez le nom de votre environnement principal. La capture d'écran suivante montre la section Sélectionnez un environnement principal à utiliser avec cette branche de la page des paramètres de génération avec le **backend** nom de l'environnement principal saisi.



b. Développez la section Paramètres avancés sur la page des paramètres de création et ajoutez des variables d'environnement pour les clés de connexion aux réseaux sociaux. Par exemple, **AMPLIFY_FACEBOOK_CLIENT_SECRET** est une variable d'environnement valide. Pour la liste des variables d'environnement du système Amplify disponibles par défaut, consultez le tableau dans. Amplifier la référence des variables d'environnement

- Connecter une succursale à une application existante

a. Si vous connectez une nouvelle branche à une application existante, définissez les variables d'environnement de connexion sociale avant de connecter la branche. Dans le volet de navigation, choisissez Paramètres de l'application, Variables d'environnement.

b. Dans la section Variables d'environnement, sélectionnez Gérer les variables.

c. Dans la section Gérer les variables, choisissez Ajouter une variable.

d. Pour Variable (clé), entrez votre identifiant client. Dans Value, entrez le secret de votre client.

e. Choisissez Enregistrer.

Gestion des secrets environnementaux

Avec la sortie d'Amplify Gen 2, le flux de travail relatif aux secrets d'environnement est rationalisé afin de centraliser la gestion des secrets et des variables d'environnement dans la console Amplify. Pour obtenir des instructions sur la configuration et l'accès aux secrets d'une application Amplify Gen 2, voir Secrets et variables d'environnement dans la documentation Amplify.

Les secrets d'environnement d'une application de première génération sont similaires aux variables d'environnement, mais il s'agit de paires clé-valeur du AWS Systems Manager Parameter Store qui peuvent être chiffrées. Certaines valeurs doivent être chiffrées, comme la clé privée de connexion avec Apple pour Amplify.

Utilisation AWS Systems Manager pour définir des secrets d'environnement pour une application Amplify Gen 1

Suivez les instructions suivantes pour définir un secret d'environnement pour une application Amplify de première génération à l'aide de la AWS Systems Manager console.

Pour définir un secret d'environnement

1. Connectez-vous à la AWS Systems Manager console AWS Management Console et ouvrez-la.
2. Dans le volet de navigation, choisissez Application Management, puis Parameter Store.
3. Sur la page AWS Systems Manager Parameter Store, choisissez Create parameter.
4. Sur la page Créer un paramètre, dans la section Détails du paramètre, procédez comme suit :
    a. Pour Nom, entrez un paramètre au format**`/amplify/{your_app_id}/{your_backend_environment_name}/{your_parameter_name}`**.
    b. Dans le champ Type, sélectionnez SecureString.
    c. Pour la source de clé KMS, choisissez Mon compte actuel pour utiliser la clé par défaut pour votre compte.
    d. Dans Valeur, entrez votre valeur secrète à chiffrer.
5. Choisissez Créer un paramètre.

> **Note**
>
> Amplify n'a accès qu'aux clés situées sous le build `/amplify/{your_app_id}/{your_backend_environment_name}` de l'environnement spécifique. Vous devez spécifier la valeur par défaut AWS KMS key pour permettre à Amplify de déchiffrer la valeur.

Accès aux secrets de l'environnement pour une application de première génération

Les secrets d'environnement d'une application Gen 1 sont stockés `process.env.secrets` sous forme de chaîne JSON.

Référence sur les secrets de l'environnement Amplify

Spécifiez un paramètre Systems Manager au format`/amplify/{your_app_id}/{your_backend_environment_name}/AMPLIFY_SIWA_CLIENT_ID`.

Vous pouvez utiliser les secrets d'environnement suivants, accessibles par défaut dans la console Amplify.

Nom de la variable	Description	Exemple de valeur
AMPLIFY_SIWA_CLIENT_ID	La connexion avec l'identifiant client Apple	`com.yourapp.auth`
AMPLIFY_SIWA_TEAM_ID	La connexion à l'aide de l'identifiant d'équipe Apple	`ABCD123`
AMPLIFY_SIWA_KEY_ID	La connexion à l'aide de l'identifiant Apple Key	`ABCD123`
AMPLIFY_SIWA_PRIVATE_KEY	La clé privée de connexion avec Apple	-----COMMENCER LA CLÉ PRIVÉE----- ****..... -----FIN DE LA CLÉ PRIVÉE-----

Configuration d'en-têtes personnalisés pour une application Amplify

Les en-têtes HTTP personnalisés vous permettent de définir des en-têtes pour chaque réponse HTTP. Les en-têtes de réponse peuvent être utilisés à des fins de débogage, de sécurité et d'information. Vous pouvez spécifier des en-têtes dans la console Amplify ou en téléchargeant et en modifiant le fichier d'une application et en `customHttp.yml` l'enregistrant dans le répertoire racine du projet. Pour connaître les procédures détaillées, consultez Configuration d'en-têtes personnalisés.

Auparavant, les en-têtes HTTP personnalisés étaient spécifiés pour une application soit en modifiant la spécification de construction (buildspec) dans la console Amplify, soit en téléchargeant et en mettant à jour le `amplify.yml` fichier et en l'enregistrant dans le répertoire racine du projet. Nous vous recommandons vivement de migrer les en-têtes personnalisés spécifiés de cette manière hors du buildspec et du fichier. `amplify.yml` Pour obtenir des instructions, veuillez consulter Migration des en-têtes personnalisés hors de la spécification de construction et amplify.yml.

Rubriques

- Référence YAML d'en-tête personnalisée
- Configuration d'en-têtes personnalisés
- Migration des en-têtes personnalisés hors de la spécification de construction et amplify.yml
- Exigences relatives aux en-têtes personnalisés de Monorepo

Référence YAML d'en-tête personnalisée

Spécifiez des en-têtes personnalisés en utilisant le format YAML suivant :

```
customHeaders:
  - pattern: '*.json'
    headers:
    - key: 'custom-header-name-1'
      value: 'custom-header-value-1'
    - key: 'custom-header-name-2'
      value: 'custom-header-value-2'
  - pattern: '/path/*'
    headers:
    - key: 'custom-header-name-1'
```

```
      value: 'custom-header-value-2'
```

Pour un monorepo, utilisez le format YAML suivant :

```
applications:
  - appRoot: app1
    customHeaders:
    - pattern: '**/*'
      headers:
      - key: 'custom-header-name-1'
        value: 'custom-header-value-1'
  - appRoot: app2
    customHeaders:
    - pattern: '/path/*.json'
      headers:
      - key: 'custom-header-name-2'
        value: 'custom-header-value-2'
```

Lorsque vous ajoutez des en-têtes personnalisés à votre application, vous devez spécifier vos propres valeurs pour les éléments suivants :

pattern

Les en-têtes personnalisés sont appliqués à tous les chemins de fichiers URL correspondant au modèle.

headers

Définit les en-têtes qui correspondent au modèle de fichier.

clé

Nom de l'en-tête personnalisé.

value

Valeur de l'en-tête personnalisé.

Pour en savoir plus sur les en-têtes HTTP, consultez la liste des en-têtes HTTP de Mozilla.

Configuration d'en-têtes personnalisés

Il existe deux manières de spécifier des en-têtes HTTP personnalisés pour une application Amplify. Vous pouvez spécifier des en-têtes dans la console Amplify ou vous pouvez spécifier des en-têtes en téléchargeant et en modifiant le fichier d'une application et en `customHttp.yml` l'enregistrant dans le répertoire racine de votre projet.

Pour définir des en-têtes personnalisés pour une application et les enregistrer dans la console

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.
3. Dans le volet de navigation, choisissez Hosting, puis Custom headers.
4. Sur la page En-têtes personnalisés, choisissez Modifier.
5. Dans la fenêtre Modifier les en-têtes personnalisés, entrez les informations relatives à vos en-têtes personnalisés en utilisant le format YAML d'en-tête personnalisé.
   a. Pour`pattern`, entrez le modèle correspondant.
   b. Pour`key`, entrez le nom de l'en-tête personnalisé.
   c. Pour`value`, entrez la valeur de l'en-tête personnalisé.
6. Choisissez Enregistrer.
7. Redéployez l'application pour appliquer les nouveaux en-têtes personnalisés.
   - Pour une CI/CD application, accédez à la branche à déployer et choisissez Redéployer cette version. Vous pouvez également effectuer une nouvelle compilation à partir de votre dépôt Git.
   - Pour une application à déploiement manuel, déployez à nouveau l'application dans la console Amplify.

Pour définir des en-têtes personnalisés pour une application et les enregistrer à la racine de votre référentiel

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.
3. Dans le volet de navigation, choisissez Hosting, puis Custom headers.
4. Sur la page des en-têtes personnalisés, choisissez Download YML.

5. Ouvrez le `customHttp.yml` fichier téléchargé dans l'éditeur de code de votre choix et entrez les informations relatives à vos en-têtes personnalisés en utilisant le format YAML d'en-tête personnalisé.

   a. Pour`pattern`, entrez le modèle correspondant.

   b. Pour`key`, entrez le nom de l'en-tête personnalisé.

   c. Pour`value`, entrez la valeur de l'en-tête personnalisé.

6. Enregistrez le `customHttp.yml` fichier modifié dans le répertoire racine de votre projet. Si vous travaillez avec un monorepo, enregistrez le `customHttp.yml` fichier à la racine de votre dépôt.
7. Redéployez l'application pour appliquer les nouveaux en-têtes personnalisés.
   - Pour une CI/CD application, effectuez une nouvelle compilation à partir de votre dépôt Git qui inclut le nouveau `customHttp.yml` fichier.
   - Pour une application à déploiement manuel, déployez à nouveau l'application dans la console Amplify et incluez le nouveau `customHttp.yml` fichier avec les artefacts que vous téléchargez.

> **Note**
>
> Les en-têtes personnalisés définis dans le `customHttp.yml` fichier et déployés dans le répertoire racine de l'application remplacent les en-têtes personnalisés définis dans la section En-têtes personnalisés de la console Amplify.

Exemple d'en-têtes personnalisés de sécurité

Les en-têtes de sécurité personnalisés permettent de renforcer le protocole HTTPS, de prévenir les attaques XSS et de protéger votre navigateur contre le clickjacking. Utilisez la syntaxe YAML suivante pour appliquer des en-têtes de sécurité personnalisés à votre application.

```
customHeaders:
  - pattern: '**'
    headers:
      - key: 'Strict-Transport-Security'
        value: 'max-age=31536000; includeSubDomains'
      - key: 'X-Frame-Options'
        value: 'SAMEORIGIN'
      - key: 'X-XSS-Protection'
```

```
        value: '1; mode=block'
      - key: 'X-Content-Type-Options'
        value: 'nosniff'
      - key: 'Content-Security-Policy'
        value: "default-src 'self'"
```

Configuration des en-têtes personnalisés de Cache-Control

Les applications hébergées avec Amplify respectent `Cache-Control` les en-têtes envoyés par l'origine, sauf si vous les remplacez par des en-têtes personnalisés que vous définissez. Amplify applique uniquement les en-têtes personnalisés Cache-Control pour les réponses réussies avec un code d'état. `200 OK` Cela empêche les réponses aux erreurs d'être mises en cache et de les transmettre aux autres utilisateurs qui font la même demande.

Vous pouvez ajuster manuellement la `s-maxage` directive pour mieux contrôler les performances et la disponibilité du déploiement de votre application. Par exemple, pour augmenter la durée pendant laquelle votre contenu reste en cache à la périphérie, vous pouvez augmenter manuellement le temps de vie (TTL) en le mettant à jour `s-maxage` à une valeur supérieure à la valeur par défaut de 600 secondes (10 minutes).

Pour spécifier une valeur personnalisée pours`-maxage`, utilisez le format YAML suivant. Cet exemple conserve le contenu associé en cache à la périphérie pendant 3 600 secondes (une heure).

```
customHeaders:
  - pattern: '/img/*'
    headers:
      - key: 'Cache-Control'
        value: 's-maxage=3600'
```

Pour plus d'informations sur le contrôle des performances des applications à l'aide d'en-têtes, consultezUtilisation de l'en-tête Cache-Control pour améliorer les performances de l'application.

Migration des en-têtes personnalisés hors de la spécification de construction et amplify.yml

Auparavant, les en-têtes HTTP personnalisés étaient spécifiés pour une application soit en modifiant la spécification de construction dans la console Amplify, soit en téléchargeant et en mettant à jour le fichier et en `amplify.yml` l'enregistrant dans le répertoire racine du projet. Il est fortement

recommandé de migrer vos en-têtes personnalisés hors de la spécification de construction et du `amplify.yml` fichier.

Spécifiez vos en-têtes personnalisés dans la section En-têtes personnalisés de la console Amplify ou en téléchargeant et en modifiant le fichier. `customHttp.yml`

Pour migrer les en-têtes personnalisés stockés dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application sur laquelle effectuer la migration d'en-têtes personnalisés.
3. Dans le volet de navigation, choisissez Hosting, Build settings. Dans la section Spécification de construction de l'application, vous pouvez consulter les spécifications de construction de votre application.
4. Choisissez Télécharger pour enregistrer une copie de vos spécifications de construction actuelles. Vous pourrez faire référence à cette copie ultérieurement si vous avez besoin de récupérer des paramètres.
5. Lorsque le téléchargement est terminé, choisissez Modifier.
6. Prenez note des informations d'en-tête personnalisées contenues dans le fichier, car vous les utiliserez ultérieurement à l'étape 9. Dans la fenêtre d'édition, supprimez les en-têtes personnalisés du fichier et choisissez Enregistrer.
7. Dans le volet de navigation, choisissez Hosting, Custom headers.
8. Sur la page En-têtes personnalisés, choisissez Modifier.
9. Dans la fenêtre Modifier les en-têtes personnalisés, entrez les informations relatives aux en-têtes personnalisés que vous avez supprimés à l'étape 6.
10. Choisissez Enregistrer.
11. Redéployez toutes les branches auxquelles vous souhaitez appliquer les nouveaux en-têtes personnalisés.

Pour migrer des en-têtes personnalisés de amplify.yml vers CustomHttp.yml

1. Accédez au `amplify.yml` fichier actuellement déployé dans le répertoire racine de votre application.
2. Ouvrez `amplify.yml` dans l'éditeur de code de votre choix.

3. Prenez note des informations d'en-tête personnalisées contenues dans le fichier, car vous les utiliserez ultérieurement à l'étape 8. Supprimez les en-têtes personnalisés du fichier. Enregistrez et fermez le fichier .
4. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
5. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.
6. Dans le volet de navigation, choisissez Hosting, Custom headers.
7. Sur la page En-têtes personnalisés, choisissez Télécharger.
8. Ouvrez le `customHttp.yml` fichier téléchargé dans l'éditeur de code de votre choix et entrez les informations relatives aux en-têtes personnalisés que vous avez supprimés `amplify.yml` à l'étape 3.
9. Enregistrez le `customHttp.yml` fichier modifié dans le répertoire racine de votre projet. Si vous travaillez avec un monorepo, enregistrez le fichier à la racine de votre dépôt.
10. Redéployez l'application pour appliquer les nouveaux en-têtes personnalisés.
    - Pour une CI/CD application, effectuez une nouvelle compilation à partir de votre dépôt Git qui inclut le nouveau `customHttp.yml` fichier.
    - Pour une application à déploiement manuel, déployez à nouveau l'application dans la console Amplify et incluez le nouveau `customHttp.yml` fichier avec les artefacts que vous téléchargez.

> **Note**
>
> Les en-têtes personnalisés définis dans le `customHttp.yml` fichier et déployés dans le répertoire racine de l'application remplacent les en-têtes personnalisés définis dans la section En-têtes personnalisés de la console Amplify.

Exigences relatives aux en-têtes personnalisés de Monorepo

Lorsque vous spécifiez des en-têtes personnalisés pour une application dans un monorepo, tenez compte des exigences de configuration suivantes :

- Il existe un format YAML spécifique pour un monorepo. Pour connaître la syntaxe correcte, consultezRéférence YAML d'en-tête personnalisée.

- Vous pouvez spécifier des en-têtes personnalisés pour une application dans un monorepo à l'aide de la section En-têtes personnalisés de la console Amplify. Vous devez redéployer votre application pour appliquer les nouveaux en-têtes personnalisés.
- Au lieu d'utiliser la console, vous pouvez spécifier des en-têtes personnalisés pour une application dans un monorepo dans un fichier. `customHttp.yml` Vous devez enregistrer le `customHttp.yml` fichier à la racine de votre dépôt, puis redéployer l'application pour appliquer les nouveaux en-têtes personnalisés. Les en-têtes personnalisés spécifiés dans le `customHttp.yml` fichier remplacent tous les en-têtes personnalisés spécifiés dans la section En-têtes personnalisés de la console Amplify.

Gestion de la configuration du cache pour une application

Amplify utilise Amazon CloudFront pour gérer la configuration de mise en cache de vos applications hébergées. Une configuration de cache est appliquée à chaque application afin d'optimiser les performances.

Le 13 août 2024, Amplify a publié des améliorations de l'efficacité de la mise en cache pour les applications. Pour plus d'informations, consultez Améliorations de la mise en cache du CDN pour améliorer les performances des applications grâce AWS Amplify à l'hébergement.

Le tableau suivant résume la prise en charge par Amplify de comportements de mise en cache spécifiques avant et après la publication des améliorations apportées à la mise en cache.

Comportement de mise en cache	Support antérieur	Avec des améliorations de mise en cache
Vous pouvez ajouter des en-têtes personnalisés pour une application dans la console Amplify ou dans `customHeaders.yaml` un fichier. L'un des en-têtes que vous pouvez remplacer est. `Cache-Control` Pour de plus amples informations, veuillez consulter Configuration d'en-têtes personnalisés pour une application Amplify.	Oui	Oui
Amplify respecte `Cache-Control` les en-têtes que vous définissez dans un `customHeaders.yaml` fichier et ils ont priorité sur les paramètres de cache par défaut d'Amplify.	Oui	Oui

Comportement de mise en cache	Support antérieur	Avec des améliorations de mise en cache
Amplify respecte `Cache-Control` les en-têtes définis dans le cadre d'une application pour les routes dynamiques (par exemple, les routes SSR Next.js). Si un `Cache-Control` en-tête est défini dans le `customHeaders.yaml` fichier de l'application, celui-ci a priorité sur les paramètres du `next.config.js` fichier.	Oui	Oui
Chaque nouveau déploiement d' CI/CD application efface le cache.	Oui	Oui
Vous pouvez activer le mode performance pour une application.	Oui	Non Le paramètre du mode performance n'est plus disponible dans la console Amplify. Vous pouvez toutefois créer un `Cache-Control` en-tête qui définit la `s-maxage` directive. Pour obtenir des instructions, veuillez consulter Utilisation de l'en-tête Cache-Control pour améliorer les performances de l'application.

Le tableau suivant répertorie les modifications apportées aux valeurs par défaut pour des paramètres de cache spécifiques.

Paramètre du cache	Valeur par défaut précédente	Valeur par défaut avec amélioration de la mise en cache
Durée du cache pour les actifs statiques	Deux secondes	Un an
Durée du cache pour les réponses proxy inversées	Deux secondes	Zéro seconde (pas de mise en cache)
Durée maximale de vie (TTL)	Dix minutes	Un an

Pour plus d'informations sur la façon dont Amplify détermine la configuration de mise en cache à appliquer à une application et pour obtenir des instructions sur la gestion de la configuration des clés de cache, consultez les rubriques suivantes.

Rubriques

- Comment Amplify applique la configuration du cache à une application
- Gestion des cookies clés du cache
- Utilisation de l'en-tête Cache-Control pour améliorer les performances de l'application

Comment Amplify applique la configuration du cache à une application

Pour gérer la mise en cache de votre application, Amplify détermine le type de contenu diffusé en examinant le type de plateforme de l'application et les règles de réécriture. Pour les `Compute` applications, Amplify examine également les règles de routage dans le manifeste de déploiement.

> **Note**
>
> Le type de plateforme de l'application est défini par Amplify Hosting lors du déploiement. Une application SSG (statique) est définie sur le type `WEB` de plateforme. Une application SSR (Next.js 12 ou version ultérieure) est définie sur le type `WEB_COMPUTE` de plate-forme.

Amplify identifie les quatre types de contenu suivants et applique la politique de cache géré spécifiée.

Statique

Le contenu diffusé par les applications associées à la `WEB` plateforme ou les itinéraires statiques d'une `WEB_COMPUTE` application.

Ce contenu utilise la politique de Amplify-StaticContent cache.

Optimisation de l'image

Les images diffusées par les `ImageOptimization` itinéraires dans une `WEB_COMPUTE` application.

Ce contenu utilise la politique de Amplify-ImageOptimization cache.

Calcul

Le contenu diffusé par les `Compute` itinéraires dans une `WEB_COMPUTE` application. Cela inclut tout le contenu rendu côté serveur (SSR).

Ce contenu utilise soit la politique de Amplify-DefaultNoCookies cache, Amplify-Default soit la politique de cache en fonction de la `cacheConfig.type` valeur définie sur votre AmplifyApp.

Proxy inversé

Le contenu diffusé par des chemins correspondant à une règle personnalisée de réécriture par proxy inverse. Pour plus d'informations sur la création de cette règle personnalisée, consultez Réécriture du proxy inversé le chapitre Utilisation des redirections.

Ce contenu utilise soit la politique de Amplify-DefaultNoCookies cache, Amplify-Default soit la politique de cache en fonction de la `cacheConfig.type` valeur définie sur votre AmplifyApp.

Comprendre les politiques de cache géré d'Amplify

Amplify utilise les politiques de cache géré prédéfinies suivantes pour optimiser la configuration de cache par défaut pour vos applications hébergées.

- Amplify-Default
- Amplify-DefaultNoCookies
- Amplify-ImageOptimization
- Amplify-StaticContent

Paramètres de politique de cache géré Amplify-Default

Afficher cette politique dans la CloudFront console

Cette stratégie est conçue pour être utilisée avec une origine qui est une appli web AWS Amplify.

Cette stratégie possède les paramètres suivants :

- Minimum TTL (Durée de vie minimale) : 0 seconde
- TTL maximum : 31536000 secondes (un an)
- Default TTL (Durée de vie par défaut) : 0 seconde
- En-têtes inclus dans la clé de cache :
  - `Authorization`
  - `Accept`
  - `CloudFront-Viewer-Country`
  - `Host`
- Cookies included in cache key (Cookies inclus dans la clé de cache) : tous les cookies sont inclus.
- Query strings included in cache key (Chaînes de requête incluses dans la clé de cache) : toutes les chaînes de requête sont incluses.
- Paramètre des objets compressés en cache : Gzip et Brotli activés.

Paramètres de politique de cache DefaultNoCookies gérés par Amplify

Afficher cette politique dans la CloudFront console

Cette stratégie est conçue pour être utilisée avec une origine qui est une appli web AWS Amplify.

Cette stratégie possède les paramètres suivants :

- Minimum TTL (Durée de vie minimale) : 0 seconde
- TTL maximum : 31536000 secondes (un an)
- Default TTL (Durée de vie par défaut) : 0 seconde
- En-têtes inclus dans la clé de cache :
  - `Authorization`
  - `Accept`

  - `CloudFront-Viewer-Country`
  - `Host`
- Cookies inclus dans la clé de cache : aucun cookie n'est inclus.
- Query strings included in cache key (Chaînes de requête incluses dans la clé de cache) : toutes les chaînes de requête sont incluses.
- Paramètre des objets compressés en cache : Gzip et Brotli activés.

Paramètres de politique de cache ImageOptimization gérés par Amplify

Afficher cette politique dans la CloudFront console

Cette stratégie est conçue pour être utilisée avec une origine qui est une appli web AWS Amplify.

Cette stratégie possède les paramètres suivants :

- Minimum TTL (Durée de vie minimale) : 0 seconde
- TTL maximum : 31536000 secondes (un an)
- Default TTL (Durée de vie par défaut) : 0 seconde
- En-têtes inclus dans la clé de cache :
  - `Authorization`
  - `Accept`
  - `Host`
- Cookies inclus dans la clé de cache : aucun cookie n'est inclus.
- Query strings included in cache key (Chaînes de requête incluses dans la clé de cache) : toutes les chaînes de requête sont incluses.
- Paramètre des objets compressés en cache : Gzip et Brotli activés.

Paramètres de politique de cache StaticContent gérés par Amplify

Afficher cette politique dans la CloudFront console

Cette stratégie est conçue pour être utilisée avec une origine qui est une appli web AWS Amplify.

Cette stratégie possède les paramètres suivants :

- Minimum TTL (Durée de vie minimale) : 0 seconde

- TTL maximum : 31536000 secondes (un an)
- Default TTL (Durée de vie par défaut) : 0 seconde
- En-têtes inclus dans la clé de cache :
  - `Authorization`
  - `Host`
- Cookies inclus dans la clé de cache : aucun cookie n'est inclus.
- Chaînes de requête incluses dans la clé de cache : aucune chaîne de requête n'est incluse.
- Paramètre des objets compressés en cache : Gzip et Brotli activés.

Gestion des cookies clés du cache

Lorsque vous déployez votre application sur Amplify, vous pouvez choisir d'inclure ou d'exclure les cookies dans la clé de cache. Dans la console Amplify, ce paramètre est spécifié sur la page En-têtes personnalisés et cache à l'aide du bouton de réglage de la touche Cache. Pour obtenir des instructions, veuillez consulter Inclure ou exclure des cookies de la clé de cache.

Inclure les cookies dans la clé de cache

Avec ce paramètre, Amplify choisit automatiquement une configuration de cache optimale pour votre application en fonction du type de contenu diffusé. Vous devez choisir explicitement ce type de configuration de cache.

Si vous utilisez le SDKs ou AWS CLI, ce paramètre correspond `cacheConfig.type` au réglage `AMPLIFY_MANAGED` avec le `CreateApp` ou `UpdateApp` APIs.

Exclure les cookies de la clé de cache

Il s'agit de la configuration de cache par défaut. Cette configuration de cache est similaire à la `AMPLIFY_MANAGED` configuration, sauf qu'elle exclut tous les cookies de la clé de cache.

Choisir d'exclure les cookies de la clé de cache peut améliorer les performances du cache. Toutefois, avant de choisir cette configuration de cache, il est important de déterminer si votre application utilise des cookies pour diffuser du contenu dynamique.

Si vous utilisez le SDKs ou AWS CLI, ce paramètre correspond au réglage du `cacheConfig.type` à `AMPLIFY_MANAGED_NO_COOKIES` avec le `CreateApp` ou `UpdateApp` APIs.

Pour plus d'informations sur la clé de cache, consultez Comprendre la clé de cache dans le manuel Amazon CloudFront Developer Guide ;.

Inclure ou exclure des cookies de la clé de cache

Vous pouvez définir la configuration des cookies liés à la clé de cache pour une application dans la console Amplify SDKs, ou dans le. AWS CLI

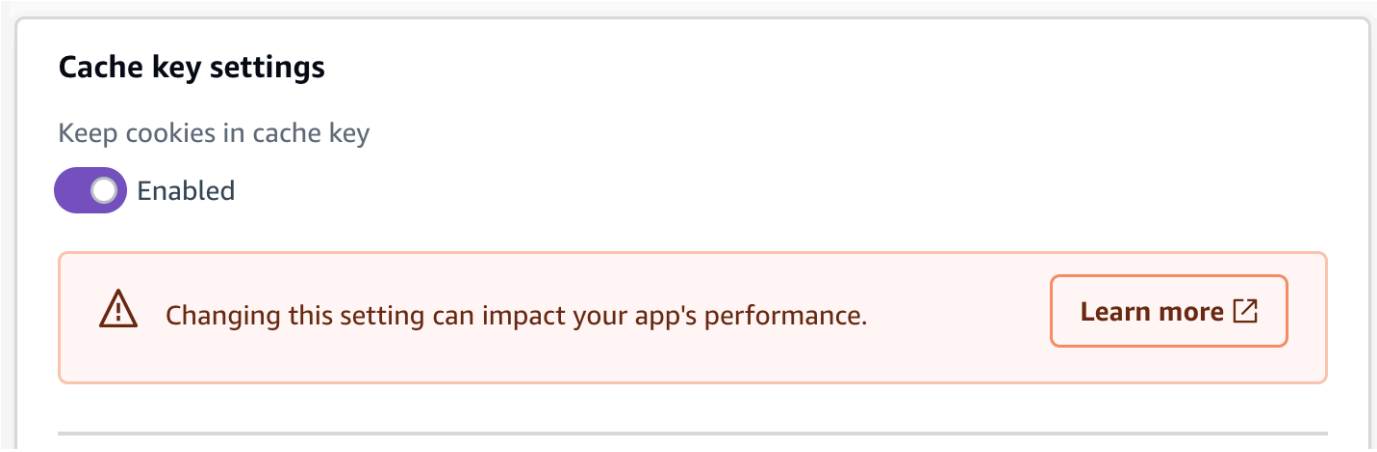
Utilisez la procédure suivante pour spécifier s'il faut inclure ou exclure les cookies de la clé de cache lorsque vous déployez une nouvelle application à l'aide de la console Amplify.

Pour définir la configuration des cookies liés à la clé de cache lors du déploiement d'une application sur Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Créer une nouvelle application.
3. Sur la page Commencer à créer avec Amplify, choisissez votre fournisseur de dépôt Git, puis choisissez Next.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :

   a. Sélectionnez le nom du référentiel à connecter.
   b. Sélectionnez le nom de la branche du référentiel à connecter.
   c. Choisissez Suivant.
5. Si l'application nécessite un rôle de service IAM, vous pouvez soit autoriser Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.
   - Pour permettre à Amplify de créer automatiquement un rôle et de l'associer à votre application :
     - Choisissez Créer et utiliser un nouveau rôle de service.
   - Pour associer un rôle de service que vous avez créé précédemment, procédez comme suit :

     a. Choisissez Utiliser un rôle de service existant.
     b. Sélectionnez le rôle à utiliser dans la liste.
6. Choisissez Paramètres avancés, puis recherchez la section Paramètres des clés du cache.
7. Choisissez Conserver les cookies dans la clé de cache ou Supprimer les cookies de la clé de cache. La capture d'écran suivante montre les paramètres de la touche Cache à bascule dans la console.



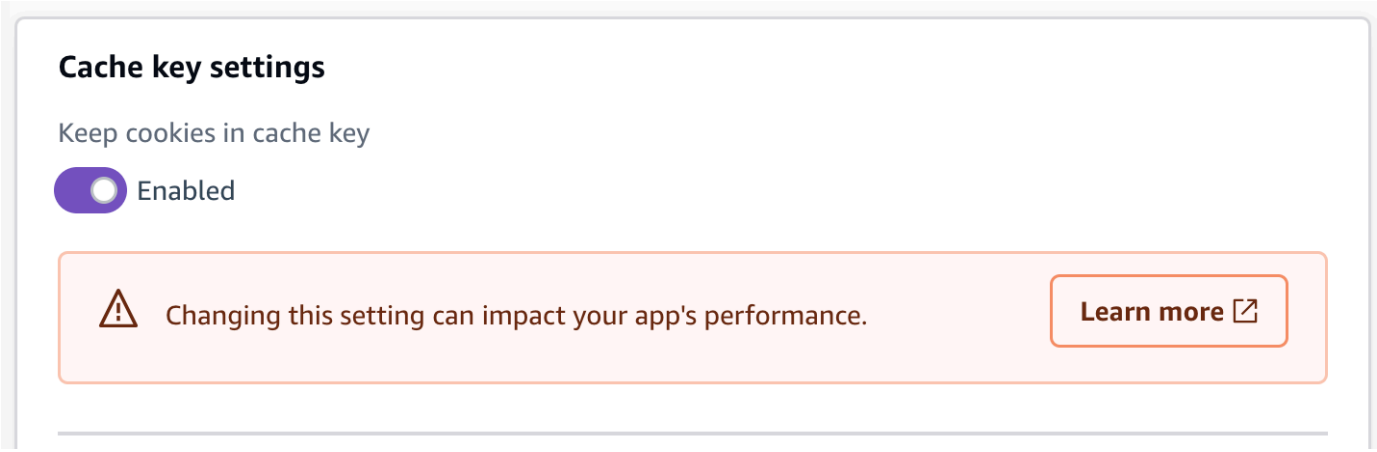
8. Choisissez Suivant.
9. Sur la page Révision, choisissez Enregistrer et déployer.

Modification de la configuration des cookies liés à la clé de cache pour une application

Vous pouvez modifier la configuration des cookies de clé de cache pour une application déjà déployée sur Amplify. Utilisez la procédure suivante pour déterminer s'il faut inclure ou exclure les cookies de la clé de cache d'une application utilisant la console Amplify.

Pour modifier la configuration des cookies liés à la clé de cache pour une application déployée

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez l'application que vous souhaitez mettre à jour.
3. Dans le volet de navigation, choisissez Hosting, puis sélectionnez Custom headers and cache.
4. Sur la page En-têtes personnalisés et cache, recherchez la section Paramètres des clés du cache et choisissez Modifier.
5. Choisissez Conserver les cookies dans la clé de cache ou Supprimer les cookies de la clé de cache. La capture d'écran suivante montre les paramètres de la touche Cache à bascule dans la console.



6. Choisissez Enregistrer.

Utilisation de l'en-tête Cache-Control pour améliorer les performances de l'application

L'architecture d'hébergement par défaut d'Amplify optimise l'équilibre entre les performances d'hébergement et la disponibilité du déploiement. Pour la plupart des clients, nous recommandons d'utiliser l'architecture par défaut.

Si vous souhaitez mieux contrôler les performances d'une application, vous pouvez définir manuellement l'`Cache-Control`en-tête HTTP afin d'optimiser les performances d'hébergement en conservant le contenu en cache à la périphérie du réseau de diffusion de contenu (CDN) pendant un intervalle plus long.

L'`Cache-Control`en-tête `max-age` et les `s-maxage` directives HTTP affectent la durée de mise en cache du contenu de votre application. La `max-age` directive indique au navigateur pendant combien de temps (en secondes) vous souhaitez que le contenu reste dans le cache avant qu'il ne soit actualisé depuis le serveur d'origine. La `s-maxage` directive remplace `max-age` et vous permet de spécifier la durée (en secondes) pendant laquelle vous souhaitez que le contenu reste sur la périphérie du CDN avant qu'il ne soit actualisé depuis le serveur d'origine.

Les applications hébergées avec Amplify respectent `Cache-Control` les en-têtes envoyés par l'origine, sauf si vous les remplacez par des en-têtes personnalisés que vous définissez. Amplify applique uniquement des en-têtes `Cache-Control` personnalisés pour les réponses réussies avec un `200 OK` code d'état. Cela empêche les réponses aux erreurs d'être mises en cache et diffusées aux autres utilisateurs qui font la même demande.

Vous pouvez ajuster manuellement la `s-maxage` directive pour mieux contrôler les performances et la disponibilité du déploiement de votre application. Par exemple, pour modifier la durée pendant laquelle votre contenu reste en cache à la périphérie, vous pouvez définir manuellement la durée de vie (TTL) en le mettant à jour `s-maxage` à une valeur autre que la valeur par défaut 31536000 secondes (un an).

Vous pouvez définir des en-têtes personnalisés pour une application dans la section En-têtes personnalisés de la console Amplify. Pour un exemple de YAML format, voirConfiguration des en-têtes personnalisés de Cache-Control.

Utilisez la procédure suivante pour définir la `s-maxage` directive afin de conserver le contenu en cache à la périphérie du CDN pendant 24 heures.

Pour définir un Cache-Control en-tête personnalisé

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.
3. Dans le volet de navigation, choisissez Hosting, Custom headers.
4. Sur la page En-têtes personnalisés, choisissez Modifier.
5. Dans la fenêtre Modifier les en-têtes personnalisés, entrez les informations relatives à votre en-tête personnalisé comme suit :

   a. Pour`pattern`, entrez **`**/*`** pour tous les chemins.

   b. Pour `key`, saisissez **`Cache-Control`**.

   c. Pour `value`, saisissez **`s-maxage=86400`**.
6. Choisissez Enregistrer.
7. Redéployez l'application pour appliquer le nouvel en-tête personnalisé.

Protection antioblique pour les déploiements d'Amplify

La protection contre le décalage de déploiement est disponible pour les applications Amplify afin d'éliminer les problèmes de distorsion de version entre le client et le serveur dans les applications Web. Lorsque vous appliquez une protection antiasymétrique à une application Amplify, vous pouvez vous assurer que vos clients interagissent toujours avec la bonne version des ressources côté serveur, quel que soit le moment du déploiement.

L'asymétrie des versions est un défi courant pour les développeurs Web. Cela se produit lorsqu'un navigateur Web exécute une version obsolète d'une application et que le serveur en exécute une nouvelle. Cet écart peut entraîner un comportement imprévisible, des erreurs et une expérience dégradée pour l'utilisateur de l'application. La fonctionnalité de protection asymétrique du déploiement d'Amplify associe les clients exécutés sur des navigateurs Web à un déploiement spécifique. Cela garantit qu'Amplify gère toujours les actifs pour ce déploiement particulier, en synchronisant le client et le serveur.

La fonction de protection asymétrique d'Amplify peut réduire les erreurs pour les utilisateurs de votre application lorsque vous lancez de nouveaux déploiements. Cela peut également améliorer l'expérience des développeurs en réduisant le temps passé à gérer les problèmes de compatibilité descendante et descendante.

Détails de la fonction de protection antioblique :

Types d'applications pris en charge

Vous pouvez ajouter une protection antioblique aux applications statiques et SSR créées avec n'importe quel framework pris en charge par Amplify. Les applications peuvent être déployées à partir d'un référentiel Git ou d'un déploiement manuel.

Vous ne pouvez pas ajouter de protection antiasymétrique à une application déployée sur la `WEB_DYNAMIC` plate-forme (Next.js version 11 ou antérieure).

Duration

Pour les applications statiques, Amplify effectue des déploiements d'une semaine. Pour les applications SSR, nous garantissons une protection asymétrique pour un maximum de huit déploiements précédents.

Cost

L'ajout d'une protection antiasymétrique à une application n'entraîne aucun coût supplémentaire.

Considération de performance

Lorsque la protection asymétrique est activée pour une application, Amplify doit mettre à jour ses configurations de cache CDN. Par conséquent, vous devez vous attendre à ce que votre premier déploiement après l'activation de la protection antiasymétrique prenne jusqu'à dix minutes.

Rubriques

Configuration de la protection contre les distorsions de déploiement pour une application Amplify

Vous pouvez ajouter ou supprimer une protection contre les distorsions de déploiement pour une application à l'aide de la console Amplify, AWS Command Line Interface du, ou du. SDKs La fonctionnalité est appliquée au niveau de la branche. Seuls les nouveaux déploiements, effectués après l'activation de la protection asymétrique pour une succursale, seront protégés contre l'asymétrie.

Pour ajouter ou supprimer une protection contre les distorsions de déploiement à l'aide des champs AWS CLI ou SDKs, utilisez les `UpdateBranch.enableSkewProtection` champs `CreateBranch.enableSkewProtection` et. Pour plus d'informations, consultez CreateBranchet consultez la UpdateBranchdocumentation de référence de l'API Amplify.

Si vous souhaitez supprimer un déploiement spécifique afin qu'il ne soit plus desservi, utilisez l'`DeleteJob`API. Pour plus d'informations, consultez la DeleteJobdocumentation de référence de l'API Amplify.

Pour le moment, vous ne pouvez activer la protection asymétrique que sur une application déjà déployée sur Amplify Hosting. Suivez les instructions suivantes pour ajouter une protection antioblique à une branche à l'aide de la console Amplify.

Activer la protection antioblique pour la branche d'une application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/

2. Sur la page Toutes les applications, choisissez le nom de l'application déployée pour activer la protection asymétrique.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Paramètres de la branche.
4. Dans la section Branches, choisissez le nom de la branche à mettre à jour.
5. Dans le menu Actions, choisissez Activer la protection antioblique.
6. Dans la fenêtre de confirmation, choisissez Confirmer. La protection antioblique est désormais activée pour la branche.
7. Redéployez votre branche d'application. Seuls les déploiements effectués après l'activation de la protection antiasymétrique sont protégés contre l'asymétrie.

Suivez les instructions ci-dessous pour supprimer la protection antiasymétrique d'une branche d'une application à l'aide de la console Amplify.

Supprimer la protection antioblique d'une branche d'une application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez le nom de l'application déployée dont vous souhaitez supprimer la protection antiasymétrique.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Paramètres de la branche.
4. Dans la section Branches, choisissez le nom de la branche à mettre à jour.
5. Dans le menu Actions, choisissez Désactiver la protection antioblique. La protection antiasymétrique est désormais désactivée pour la branche et seul le contenu le plus récent sera diffusé.

Comment fonctionne la protection antiasymétrique

Dans la plupart des cas, le comportement par défaut du cookie _dpl répondra à vos besoins de protection asymétrique. Toutefois, dans les scénarios avancés suivants, il est préférable d'activer la protection antioblique à l'aide des paramètres `X-Amplify-Dpl` d'en-tête et de `dpl` requête.

- Chargement de votre site Web dans plusieurs onglets de navigateur en même temps
- Recours à des travailleurs de service

Amplify évalue la demande entrante dans l'ordre suivant lors de la détermination du contenu à diffuser au client :

1. **`X-Amplify-Dpl`**en-tête — Les applications peuvent utiliser cet en-tête pour diriger les demandes vers un déploiement Amplify spécifique. Cet en-tête de demande peut être défini en utilisant la valeur de`process.env.AWS_AMPLIFY_DEPLOYMENT_ID`.
2. **`dpl`**paramètre de requête — Les applications Next.js définiront automatiquement le paramètre de requête _dpl pour les demandes relatives aux actifs comportant des empreintes digitales (fichiers .js et .css).
3. cookie _dpl — Il s'agit du cookie par défaut pour toutes les applications protégées par asymétrie. Pour un navigateur spécifique, le même cookie est envoyé pour chaque onglet ou instance de navigateur qui interagit avec un domaine.

   Sachez que si différentes versions d'un site Web sont chargées dans différents onglets du navigateur, le cookie _dpl est partagé par tous les onglets. Dans ce scénario, il n'est pas possible d'obtenir une protection totale contre l'asymétrie avec le cookie _dpl et vous devriez envisager d'utiliser l'`X-Amplify-Dpl`en-tête pour la protection antiasymétrique.

X-Amplify-Dpl exemple d'en-tête

L'exemple suivant illustre le code d'une page SSR Next.js qui accède à la protection antiasymétrique via l'en-tête. `X-Amplify-Dpl` La page affiche son contenu en fonction de l'une de ses routes API. Le déploiement à desservir à la route de l'API est spécifié à l'aide de l'`X-Amplify-Dpl`en-tête, qui est défini sur la valeur de`process.env.AWS_AMPLIFY_DEPLOYMENT_ID`.

```
import { useEffect, useState } from 'react';

export default function MyPage({deploymentId}) {
    const [data, setData] = useState(null);

    useEffect(() => {
        fetch('/api/hello', {
            headers: {
                'X-Amplify-Dpl': process.env.AWS_AMPLIFY_DEPLOYMENT_ID
            },
        })
        .then(res => res.json())
        .then(data => setData(data))
        .catch(error => console.error("error", error))
```

```
    }, []);

    return <div>
        {data ? JSON.stringify(data) : "Loading ... " }
    </div>
}
```

Surveillance d'une application Amplify

AWS Amplify fournit les fonctionnalités suivantes pour surveiller vos applications hébergées :

- CloudWatch métriques — Amplify émet des métriques via Amazon CloudWatch que vous pouvez utiliser pour surveiller le trafic, les erreurs, le transfert de données et la latence de vos applications.
- Journaux d'accès — Amplify fournit des journaux d'accès contenant des informations détaillées sur les demandes adressées à votre application.
- CloudTrail journalisation — Amplify est intégré et fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amplify. AWS CloudTrail Vous pouvez consulter ces événements dans la CloudTrail console.

Rubriques

Surveillance d'une application Amplify avec Amazon CloudWatch

AWS Amplify est intégré à Amazon CloudWatch, ce qui vous permet de surveiller les métriques de vos applications Amplify en temps quasi réel et de créer des alarmes qui envoient des notifications lorsqu'une métrique dépasse un seuil que vous avez défini. Pour plus d'informations sur le fonctionnement du CloudWatch service, consultez le guide de CloudWatch l'utilisateur Amazon.

CloudWatch Métriques prises en charge

Amplify prend en charge sept CloudWatch métriques dans l'espace de `AWS/AmplifyHosting` noms pour surveiller le trafic, les erreurs, le transfert de données, la latence et les jetons de demande pour vos applications. Ces mesures sont agrégées à intervalles d'une minute. CloudWatch les indicateurs de surveillance sont gratuits et ne sont pas pris en compte dans les quotas CloudWatch de service.

Le tableau suivant décrit chaque métrique prise en charge et répertorie les statistiques les plus pertinentes. Les statistiques disponibles ne sont pas toutes applicables à tous les indicateurs.

Métrique	Description
Requêtes	Le nombre total de demandes de visiteurs reçues par votre application. La statistique la plus pertinente est`Sum`. Utilisez les `Sum` statistiques pour obtenir le nombre total de demandes.
BytesDownloaded	La quantité totale de données transférées depuis votre application (téléchargées) en octets par les utilisateurs pour `GET`, `HEAD`, et les `OPTIONS` demandes. La statistique la plus pertinente est`Sum`.
BytesUploaded	La quantité totale de données transférées dans votre application (téléchargées) en octets pour toute demande, y compris les en-têtes. Amplify ne vous facture pas pour les données téléchargées dans vos applications. La statistique la plus pertinente est`Sum`.
4xxErrors	Nombre de demandes ayant renvoyé une erreur dans la plage de codes d'état HTTP comprise entre 400 et 499. La statistique la plus pertinente est`Sum`. Utilisez les `Sum` statistiques pour obtenir le nombre total d'occurrences de ces erreurs.
5xxErrors	Nombre de demandes ayant renvoyé une erreur dans la plage de codes d'état HTTP comprise entre 500 et 599.

Métrique	Description
	La statistique la plus pertinente est`Sum`. Utilisez les `Sum` statistiques pour obtenir le nombre total d'occurrences de ces erreurs.
Latence	Temps écoulé jusqu'au premier octet, en secondes. Il s'agit du délai total entre le moment où Amplify Hosting reçoit une demande et le moment où il renvoie une réponse au réseau. Cela n'inclut pas la latence réseau rencontrée pour qu'une réponse atteigne l'appareil du spectateur. Les statistiques les plus pertinentes sont `Average` `Maximum``Minimum`,`p10`,`p50`,`p90`,`p95`, et`p100`.  Utilisez les `Average` statistiques pour évaluer les latences attendues.

Métrique	Description
TokensConsumed	Les jetons de demande consommés par votre application. La `Sum` statistique représente la consommation totale de jetons de demande. Vous pouvez comparer cette statistique à votre quota de `Request tokens per second` service actuel pour déterminer si vous devez demander une augmentation du quota afin d'éviter un éventuel ralentissement lors d'un futur événement de trafic élevé. La `Average` statistique représente la consommation de jetons de demande en période normale et en période de pointe. Une consommation de jetons plus élevée entraîne généralement un allongement du délai d'obtention du premier octet (TTFB). Vous pouvez donc utiliser cette statistique pour évaluer la latence de votre application. Si votre latence est faible, vous pouvez améliorer votre système en aval APIs afin de réduire votre consommation de jetons et d'éviter le ralentissement qui peut survenir lorsque la consommation de jetons dépasse le quota de `Request tokens per second` service de votre application. Pour plus d'informations sur le quota `Request tokens per second` de service, consultez Quotas du service Amplify Hosting.

Amplify fournit les dimensions CloudWatch métriques suivantes.

Dimension	Description
Appli	Les données métriques sont fournies par l'application.
Compte AWS	Les données métriques sont fournies dans toutes les applications du Compte AWS.

Accès aux CloudWatch métriques

Vous pouvez accéder aux CloudWatch métriques directement depuis la console Amplify en suivant la procédure suivante.

> **Note**
>
> Vous pouvez également accéder aux CloudWatch métriques dans le AWS Management Console at https://console.aws.amazon.com/cloudwatch/.

Pour accéder aux métriques dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application dont vous souhaitez consulter les statistiques.
3. Dans le volet de navigation, choisissez Monitoring, puis Metrics.

Création d' CloudWatch alarmes

Vous pouvez créer des CloudWatch alarmes dans la console Amplify qui envoient des notifications lorsque des critères spécifiques sont remplis. Une alarme surveille une seule CloudWatch métrique et envoie une notification Amazon Simple Notification Service lorsque la métrique dépasse le seuil pour un certain nombre de périodes d'évaluation.

Vous pouvez créer des alarmes plus avancées qui utilisent des expressions mathématiques métriques dans la CloudWatch console ou à l'aide du CloudWatch APIs. Par exemple, vous pouvez créer une alarme qui vous avertit lorsque le pourcentage 4xxErrors dépasse 15 % pendant trois périodes consécutives. Pour plus d'informations, consultez la section Création CloudWatch d'une

alarme basée sur une expression mathématique métrique dans le guide de CloudWatch l'utilisateur Amazon.

La CloudWatch tarification standard s'applique aux alarmes. Pour plus d'informations, consultez les CloudWatchtarifs Amazon.

Utilisez la procédure suivante pour créer une alarme dans la console Amplify.

Pour créer une CloudWatch alarme pour une métrique Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application sur laquelle vous souhaitez activer une alarme.
3. Dans le volet de navigation, choisissez Surveillance, puis Alarmes.
4. Sur la page Alarmes, choisissez Créer une alarme.
5. Dans la fenêtre Créer une alarme, configurez votre alarme comme suit :

    a. Pour Metric, choisissez le nom de la métrique à surveiller dans la liste.
    b. Dans Nom de l'alarme, entrez un nom significatif pour l'alarme. Par exemple, si vous surveillez des demandes, vous pouvez nommer l'alarme**`HighTraffic`**. Le nom ne doit contenir que des caractères ASCII.
    c. Pour configurer les notifications, effectuez l'une des opérations suivantes :
        - i. Choisissez Nouveau pour configurer une nouvelle rubrique Amazon SNS.
            ii. Dans Adresse e-mail, entrez l'adresse e-mail du destinataire des notifications.
            iii. Choisissez Ajouter une nouvelle adresse e-mail pour ajouter des destinataires supplémentaires.
        - i. Choisissez Existing pour réutiliser une rubrique Amazon SNS.
            ii. Pour le sujet SNS, sélectionnez le nom d'un sujet Amazon SNS existant dans la liste.
    d. Pour Whenever the Statistic of Metric, définissez les conditions de votre alarme comme suit :

        i. Spécifiez si la métrique doit être supérieure, inférieure ou égale à la valeur du seuil.
        ii. Spécifiez la valeur de seuil.
        iii. Spécifiez le nombre de périodes d'évaluation consécutives qui doivent être en état d'alarme pour déclencher l'alarme.
        iv. Spécifiez la durée de la période d'évaluation.
    e. Choisissez Confirmer.

> **Note**
>
> Chaque destinataire Amazon SNS que vous spécifiez reçoit un e-mail de confirmation de la part de AWS Notifications. L'e-mail contient un lien que le destinataire doit suivre pour confirmer son abonnement et recevoir des notifications.

Accès aux CloudWatch journaux pour les applications SSR

Amplify envoie des informations sur votre environnement d'exécution SSR à Amazon CloudWatch Logs dans votre. Compte AWS Lorsque vous déployez une application SSR sur Amplify Hosting Compute, l'application nécessite un rôle de service IAM qu'Amplify assume lorsqu'elle appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.

Si vous choisissez d'autoriser Amplify à créer un rôle IAM pour vous, le rôle sera déjà autorisé à créer des journaux. CloudWatch Si vous créez votre propre rôle IAM, vous devrez ajouter les autorisations suivantes à votre politique pour permettre à Amplify d'accéder à Amazon CloudWatch Logs.

```
logs:CreateLogStream
logs:CreateLogGroup
logs:DescribeLogGroups
logs:PutLogEvents
```

Pour plus d'informations sur l'ajout d'un rôle de service, consultezAjouter un rôle de service avec des autorisations pour déployer des ressources de backend. Pour plus d'informations sur le déploiement d'applications rendues côté serveur, consultez. Déploiement d'applications rendues côté serveur avec Amplify Hosting

Vous pouvez consulter les journaux de calcul d'Amplify Hosting pour une application SSR dans la CloudWatch console ou dans la console Amplify. Suivez les instructions suivantes pour afficher les journaux dans la console Amplify.

Pour afficher CloudWatch les journaux d'une application SSR dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application SSR dont vous souhaitez consulter les CloudWatch journaux.
3. Dans le volet de navigation, choisissez Monitoring, puis Hosting compute logs.

4. Sur la page Hébergement des journaux de calcul, recherchez et sélectionnez un groupe de CloudWatch journaux pour une branche spécifique.

Récupération et analyse des journaux d'accès pour une application Amplify

Amplify stocke les journaux d'accès pour toutes les applications que vous hébergez dans Amplify. Les journaux d'accès contiennent des informations sur les demandes adressées à vos applications hébergées. Amplify conserve tous les journaux d'accès à une application jusqu'à ce que vous supprimiez l'application. Tous les journaux d'accès à une application sont disponibles dans la console Amplify. Cependant, chaque demande individuelle de journaux d'accès est limitée à une période de deux semaines que vous spécifiez.

> **Warning**
>
> N'incluez pas de secrets, d'informations d'identification ou de données sensibles dans URLs les paramètres de chemin ou de requête. Ces valeurs sont consultables en texte brut dans les journaux d'accès de votre application Amplify.

Amplify ne réutilise jamais les CloudFront distributions entre clients. Amplify crée CloudFront des distributions à l'avance afin que vous n'ayez pas à attendre qu'une CloudFront distribution soit créée lorsque vous déployez une nouvelle application. Avant que ces distributions ne soient attribuées à une application Amplify, elles peuvent recevoir du trafic provenant de robots. Toutefois, ils sont configurés pour toujours répondre comme Introuvables avant d'être assignés. Si les journaux d'accès de votre application contiennent des entrées relatives à une période antérieure à la création de votre application, ces entrées sont liées à cette activité.

> **Important**
>
> Il est recommandé d'utiliser les journaux pour comprendre la nature des demandes de votre contenu, et non comme comptabilisation complète de toutes les demandes. Amplify fournit des journaux d'accès dans la mesure du possible. L'entrée du journal pour une demande particulière peut être fournie bien après le traitement réel de la demande et, dans de rares cas, une entrée du journal peut ne pas être fournie du tout. Lorsqu'une entrée de journal est

omise des journaux d'accès, le nombre d'entrées dans les journaux d'accès ne correspond pas à l'utilisation indiquée dans les rapports AWS de facturation et d'utilisation.

Récupération des journaux d'accès d'une application

Utilisez la procédure suivante pour récupérer les journaux d'accès d'une application Amplify.

Pour consulter les journaux d'accès

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez consulter les journaux d'accès.
3. Dans le volet de navigation, choisissez Monitoring, puis Access logs.
4. Choisissez Modifier la plage horaire.
5. Dans la fenêtre Modifier la plage horaire, procédez comme suit.
   a. Pour Date de début, spécifiez le premier jour de l'intervalle de deux semaines pour lequel vous souhaitez récupérer les journaux.
   b. Pour Heure de début, choisissez l'heure du premier jour pour commencer la récupération du journal.
   c. Choisissez Confirmer.
6. La console Amplify affiche les journaux pour la plage de temps spécifiée dans la section Journaux d'accès. Choisissez Télécharger pour enregistrer les journaux au format CSV.

Analyse des journaux d'accès

Pour analyser les journaux d'accès, vous pouvez stocker les fichiers CSV dans un compartiment Amazon S3. L'un des moyens d'analyser vos journaux d'accès consiste à utiliser Athena. Athena est un service de requêtes interactif qui peut vous aider à analyser les données pour AWS les services. Vous pouvez suivre les step-by-step instructions ici pour créer une table. Une fois votre table créée, vous pouvez interroger les données comme suit.

```
SELECT SUM(bytes) AS total_bytes
FROM logs
WHERE "date" BETWEEN DATE '2018-06-09' AND DATE '2018-06-11'
LIMIT 100;
```

Journalisation des appels d'API Amplify à l'aide AWS CloudTrail

AWS Amplify est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amplify. CloudTrail capture tous les appels d'API pour Amplify sous forme d'événements. Les appels capturés incluent des appels provenant de la console Amplify et des appels de code vers les opérations de l'API Amplify. Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrailévénements vers un compartiment Amazon S3, y compris des événements pour Amplify. 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 CloudTrail collectées, vous pouvez déterminer la demande qui a été faite à Amplify, l'adresse IP à partir de laquelle la demande a été faite, qui a fait la demande, quand elle a été faite et des détails supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Amplifiez les informations dans CloudTrail

CloudTrail est activé par défaut sur votre AWS compte. Lorsqu'une activité se produit dans Amplify, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez afficher, rechercher et télécharger les événements récents dans votre compte AWS . Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements dans le Guide de AWS CloudTrail l'utilisateur.

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements pour Amplify, créez un parcours. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal de suivi 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 rubriques suivantes dans le Guide de l'utilisateur AWS CloudTrail :

- Création d'un parcours pour votre AWS compte
- CloudTrail services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les opérations Amplify sont enregistrées CloudTrail et documentées dans la référence d'API de AWS Amplify console, la référence d'API d'interface utilisateur d'administration d'AWS Amplify et la référence d'API d'Amplify UI Builder. Par exemple, les appels aux `CreateApp` `DeleteBackendEnvironment` opérations `DeleteApp` et les opérations 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 les éléments suivants :

- La demande a-t-elle été faite avec les informations d'identification de l'utilisateur root ou Gestion des identités et des accès AWS (IAM) ?
- La demande a-t-elle été faite avec des informations d'identification de sécurité temporaires pour un rôle ou un utilisateur fédéré ?
- La demande a-t-elle été faite par un autre AWS service ?

Pour plus d'informations, consultez l'élément CloudTrail UserIdentity dans le Guide de l'AWS CloudTrail utilisateur.

Comprendre les entrées du fichier journal Amplify

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 qui illustre le `ListApps`fonctionnement de référence de l'API de AWS Amplify console.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::444455556666:user/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
        "userName": "Mary_Major",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-01-12T05:48:10Z"
            }
        }
    },
    "eventTime": "2021-01-12T06:47:29Z",
    "eventSource": "amplify.amazonaws.com",
    "eventName": "ListApps",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.898
 Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.275-b01
 java/1.8.0_275 vendor/Oracle_Corporation",
    "requestParameters": {
        "maxResults": "100"
    },
    "responseElements": null,
    "requestID": "1c026d0b-3397-405a-95aa-aa43aexample",
    "eventID": "c5fca3fb-d148-4fa1-ba22-5fa63example",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "444455556666"
}
```

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'ListBackendJobsopération de référence de l'API AWS Amplify Admin UI.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::444455556666:user/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary_Major",
```

```
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-01-13T00:47:25Z"
            }
        }
    },
    "eventTime": "2021-01-13T01:15:43Z",
    "eventSource": "amplifybackend.amazonaws.com",
    "eventName": "ListBackendJobs",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.898
Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.275-b01
java/1.8.0_275 vendor/Oracle_Corporation",
    "requestParameters": {
        "appId": "d23mv2oexample",
        "backendEnvironmentName": "staging"
    },
    "responseElements": {
        "jobs": [
            {
                "appId": "d23mv2oexample",
                "backendEnvironmentName": "staging",
                "jobId": "ed63e9b2-dd1b-4bf2-895b-3d5dcexample",
                "operation": "CreateBackendAuth",
                "status": "COMPLETED",
                "createTime": "1610499932490",
                "updateTime": "1610500140053"
            },
            {
                "appId": "d23mv2oexample",
                "backendEnvironmentName": "staging",
                "jobId": "06904b10-a795-49c1-92b7-185dfexample",
                "operation": "CreateBackend",
                "status": "COMPLETED",
                "createTime": "1610499657938",
                "updateTime": "1610499704458"
            }
        ],
        "appId": "d23mv2oexample",
        "backendEnvironmentName": "staging"
```

```
    },
    "requestID": "7adfabd6-98d5-4b11-bd39-c7deaexample",
    "eventID": "68769310-c96c-4789-a6bb-68b52example",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "444455556666"
}
```

Utilisation des rôles IAM avec les applications Amplify

Un rôle IAM est une identité IAM dotée d'autorisations spécifiques. Les autorisations du rôle déterminent ce que l'identité peut et ne peut pas faire dans ce rôle AWS. Vous pouvez créer des rôles IAM dans votre Compte AWS et les utiliser pour déléguer des autorisations à Amplify Hosting. Pour en savoir plus sur les rôles, consultez la section Rôles IAM dans le Guide de l'utilisateur IAM.

Vous pouvez utiliser les types de rôles IAM suivants pour accorder à Amplify Hosting les autorisations nécessaires pour effectuer des actions en votre nom ou exécuter du code de calcul qui accède AWS à d'autres ressources.

Rôle de service IAM

Amplify assume ce rôle pour effectuer des actions en votre nom. Ce rôle est requis pour les applications dotées de ressources dorsales.

Rôle de calcul IAM SSR

Permet à une application de rendu côté serveur (SSR) d'accéder en toute sécurité à des ressources spécifiques. AWS

Rôle IAM SSR Logs CloudWatch

Lorsque vous déployez une application SSR, celle-ci nécessite un rôle de service IAM qu'Amplify assume pour permettre à Amplify d'accéder à Amazon Logs. CloudWatch

Rubriques

- Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend
- Ajout d'un rôle SSR Compute pour autoriser l'accès aux AWS ressources
- Ajouter un rôle de service avec des autorisations d'accès aux CloudWatch journaux

Ajouter un rôle de service avec des autorisations pour déployer des ressources de backend

Amplify a besoin d'autorisations pour déployer des ressources de backend avec votre front-end. Pour cela, vous utilisez un rôle de service. Un rôle de service est le rôle Gestion des identités et des accès

AWS (IAM) qui fournit à Amplify Hosting les autorisations nécessaires pour déployer, créer et gérer des backends en votre nom.

Lorsque vous créez une nouvelle application qui nécessite un rôle de service IAM, vous pouvez soit autoriser Amplify Hosting à créer automatiquement un rôle de service pour vous, soit sélectionner un rôle IAM que vous avez déjà créé. Dans cette section, vous apprendrez à créer un rôle de service Amplify doté d'autorisations administratives de compte et autorisant explicitement un accès direct aux ressources dont les applications Amplify ont besoin pour déployer, créer et gérer les backends.

Création d'un rôle de service Amplify dans la console IAM

Pour créer un rôle de service

1. Ouvrez la console IAM et choisissez Rôles dans la barre de navigation de gauche, puis choisissez Créer un rôle.
2. Sur la page Sélectionner une entité de confiance, choisissez Service AWS . Dans le cas d'utilisation, sélectionnez Amplify - Backend Deployment, puis choisissez Next.
3. Sur la page Ajouter des autorisations, sélectionnez Suivant.
4. Sur la page Nom, affichage et création, pour Nom du rôle, entrez un nom significatif, tel que**`AmplifyConsoleServiceRole-AmplifyRole`**.
5. Acceptez toutes les valeurs par défaut et choisissez Create role.
6. Retournez à la console Amplify pour associer le rôle à votre application.
   - Si vous êtes en train de déployer une nouvelle application, procédez comme suit :
     a. Actualisez la liste des rôles de service.
     b. Sélectionnez le rôle que vous venez de créer. Pour cet exemple, cela devrait ressembler à AmplifyConsoleServiceRole- AmplifyRole.
     c. Choisissez Next et suivez les étapes pour terminer le déploiement de votre application.
   - Si vous possédez déjà une application, procédez comme suit :
     a. Dans le volet de navigation, choisissez Paramètres de l'application, puis choisissez Rôles IAM.
     b. Sur la page des rôles IAM, dans la section Rôle de service, sélectionnez Modifier.
     c. Sur la page Rôle de service, sélectionnez le rôle que vous venez de créer dans la liste des rôles de service.
     d. Choisissez Enregistrer.

7. Amplify est désormais autorisé à déployer des ressources de backend pour votre application.

Modifier la politique de confiance d'un rôle de service pour éviter toute confusion chez les adjoints

Le problème de député confus est un problème de sécurité dans lequel une entité qui n'est pas autorisée à effectuer une action peut contraindre une entité plus privilégiée à le faire. Pour de plus amples informations, veuillez consulter Prévention du problème de l'adjoint confus entre services.

Actuellement, la politique de confiance par défaut pour le rôle de `Amplify-Backend Deployment` service applique les clés de condition `aws:SourceArn` contextuelle et `aws:SourceAccount` globale afin d'éviter toute confusion chez les adjoints. Toutefois, si vous avez déjà créé un `Amplify-Backend Deployment` rôle dans votre compte, vous pouvez mettre à jour la politique de confiance du rôle afin d'ajouter ces conditions afin de vous protéger contre la confusion chez les adjoints.

Utilisez l'exemple suivant pour restreindre l'accès aux applications de votre compte. Remplacez la région et l'ID de l'application dans l'exemple par vos propres informations.

```
"Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
```

Pour obtenir des instructions sur la modification de la politique de confiance pour un rôle à l'aide de AWS Management Console, consultez la section Modification d'un rôle (console) dans le guide de l'utilisateur IAM.

Ajout d'un rôle SSR Compute pour autoriser l'accès aux AWS ressources

Cette intégration vous permet d'attribuer un rôle IAM au service Amplify SSR Compute afin de permettre à votre application de rendu côté serveur (SSR) d'accéder en toute sécurité à des ressources AWS spécifiques en fonction des autorisations du rôle. Par exemple, vous pouvez

autoriser les fonctions de calcul SSR de votre application à accéder en toute sécurité à d'autres AWS services ou ressources, tels qu' Amazon Bedrock un bucket Amazon S3, en fonction des autorisations définies dans le rôle IAM attribué.

Le rôle de calcul IAM SSR fournit des informations d'identification temporaires, éliminant ainsi le besoin de coder en dur des informations de sécurité de longue durée dans les variables d'environnement. L'utilisation du rôle IAM SSR Compute est conforme aux meilleures pratiques de AWS sécurité qui consistent à accorder des autorisations de moindre privilège et à utiliser des informations d'identification à court terme lorsque cela est possible.

Les instructions présentées plus loin dans cette section décrivent comment créer une politique avec des autorisations personnalisées et comment associer la politique à un rôle. Lorsque vous créez le rôle, vous devez joindre une politique de confiance personnalisée qui autorise Amplify à assumer le rôle. Si la relation de confiance n'est pas définie correctement, vous recevrez un message d'erreur lorsque vous tenterez d'ajouter le rôle. La politique de confiance personnalisée suivante accorde à Amplify l'autorisation d'assumer le rôle.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "amplify.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Vous pouvez associer un rôle IAM à une application SSR existante Compte AWS à l'aide de la console Amplify ou du. AWS SDKs AWS CLI Le rôle que vous attachez est automatiquement associé au service de calcul Amplify SSR, lui accordant les autorisations que vous spécifiez pour accéder à d'autres ressources. AWS Au fur et à mesure que les besoins de votre application évoluent, vous

pouvez modifier le rôle IAM associé sans redéployer votre application. Cela apporte de la flexibilité et réduit les temps d'arrêt des applications.

> **Important**
>
> Vous êtes responsable de la configuration de votre application pour répondre à vos objectifs de sécurité et de conformité. Cela inclut la gestion de votre rôle SSR Compute, qui doit être configuré de manière à disposer du minimum d'autorisations nécessaires pour prendre en charge votre cas d'utilisation. Pour de plus amples informations, veuillez consulter Gestion de la sécurité des rôles de calcul IAM SSR.

Création d'un rôle de calcul SSR dans la console IAM

Avant de pouvoir associer un rôle de calcul IAM SSR à une application Amplify, le rôle doit déjà exister dans votre. Compte AWS Dans cette section, vous apprendrez à créer une politique IAM et à l'associer à un rôle qu'Amplify peut assumer pour accéder AWS à des ressources spécifiques.

Nous vous recommandons de suivre la AWS meilleure pratique qui consiste à accorder des autorisations de moindre privilège lors de la création d'un rôle IAM. Le rôle IAM SSR Compute est appelé uniquement à partir des fonctions de calcul SSR et ne doit donc accorder que les autorisations requises pour exécuter le code.

Vous pouvez utiliser le AWS Management Console AWS CLI, ou SDKs pour créer des politiques dans IAM. Pour plus d'informations, voir Définir des autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Les instructions suivantes montrent comment utiliser la console IAM pour créer une politique IAM qui définit les autorisations à accorder au service Amplify Compute.

Pour utiliser l'éditeur de stratégie JSON de la console IAM pour créer une stratégie

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Choisissez Create Policy (Créer une politique).
4. Dans la section Éditeur de politiques, choisissez l'option JSON.
5. Composez ou collez un document de politique JSON.

6. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.
7. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé une stratégie, suivez les instructions ci-dessous pour associer la stratégie à un rôle IAM.

Pour créer un rôle qui accorde des autorisations Amplify à des ressources spécifiques AWS

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de la console, choisissez Rôles, puis Créer un rôle.
3. Choisissez le type de rôle Custom trust policy (Politique d'approbation personnalisée).
4. Dans la section Politique de confiance personnalisée, entrez la politique de confiance personnalisée pour le rôle. Une politique de confiance dans les rôles est requise et définit les principaux auxquels vous faites confiance pour assumer le rôle.

   Copiez et collez la politique de confiance suivante pour autoriser le service Amplify à assumer ce rôle.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "amplify.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
```

```
}
```

5. Résolvez tous les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.
6. Sur la page Ajouter des autorisations, recherchez le nom de la politique que vous avez créée lors de la procédure précédente et sélectionnez-la. Ensuite, sélectionnez Suivant.
7. Pour Role name (Nom du rôle), saisissez un nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne sont pas distingués au cas par cas. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Dans la mesure AWS où d'autres ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois celui-ci créé.
8. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
9. (Facultatif) Choisissez Modifier dans les sections Étape 1 : sélection d'entités de confiance ou Étape 2 : Ajouter des autorisations pour modifier la politique et les autorisations personnalisées pour le rôle.
10. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Ajouter un rôle IAM SSR Compute à une application Amplify

Après avoir créé un rôle IAM dans votre Compte AWS, vous pouvez l'associer à une application dans la console Amplify.

Pour ajouter un rôle SSR Compute à une application dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez le nom de l'application à laquelle ajouter un rôle de calcul.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis choisissez Rôles IAM.
4. Dans la section Rôle de calcul, choisissez Modifier.
5. Dans la liste des rôles par défaut, recherchez le nom du rôle que vous souhaitez associer et sélectionnez-le. Pour cet exemple, vous pouvez choisir le nom du rôle que vous avez créé lors de la procédure précédente. Par défaut, le rôle que vous sélectionnez sera associé à toutes les branches de votre application.

Si la relation de confiance du rôle n'est pas définie correctement, un message d'erreur s'affichera et vous ne pourrez pas ajouter le rôle.

6. (facultatif) Si votre application se trouve dans un référentiel public et utilise la création automatique de branches ou si les aperçus Web sont activés pour les pull requests, nous vous déconseillons d'utiliser un rôle au niveau de l'application. Associez plutôt le rôle Compute uniquement aux branches nécessitant un accès à des ressources spécifiques. Pour modifier le comportement par défaut au niveau de l'application et associer un rôle à une branche spécifique, procédez comme suit :

   a. Pour Branche, sélectionnez le nom de la branche à utiliser.

   b. Pour Compute role, sélectionnez le nom du rôle à associer à la branche.

7. Choisissez Enregistrer.

Gestion de la sécurité des rôles de calcul IAM SSR

La sécurité est une responsabilité partagée entre vous AWS et vous. Vous êtes responsable de la configuration de votre application pour répondre à vos objectifs de sécurité et de conformité. Cela inclut la gestion de votre rôle SSR Compute, qui doit être configuré de manière à disposer du minimum d'autorisations nécessaires pour prendre en charge votre cas d'utilisation. Les informations d'identification pour le rôle SSR Compute que vous spécifiez sont immédiatement disponibles lors de l'exécution de votre fonction SSR. Si votre code SSR expose ces informations d'identification, soit intentionnellement, en raison d'un bogue, soit en autorisant l'exécution de code à distance (RCE), un utilisateur non autorisé peut accéder au rôle SSR et à ses autorisations.

Lorsqu'une application d'un référentiel public utilise un rôle SSR Compute et utilise la création automatique de branches ou des aperçus Web pour les pull requests, vous devez gérer avec soin les branches autorisées à accéder à ce rôle. Nous vous recommandons de ne pas utiliser de rôle au niveau de l'application. Vous devez plutôt associer un rôle de calcul au niveau de la branche. Cela vous permet d'accorder des autorisations uniquement aux succursales qui ont besoin d'accéder à des ressources spécifiques.

Si les informations d'identification de votre rôle sont exposées, prenez les mesures suivantes pour supprimer tout accès aux informations d'identification du rôle.

1. Révoquer toutes les sessions

Pour obtenir des instructions sur la révocation immédiate de toutes les autorisations relatives aux informations d'identification du rôle, voir Révoquer les informations d'identification de sécurité temporaires du rôle IAM.

2. Supprimer le rôle de la console Amplify

   Cette action prend effet immédiatement. Vous n'avez pas besoin de redéployer votre application.

Pour supprimer un rôle Compute dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/amplify/
2. Sur la page Toutes les applications, choisissez le nom de l'application dont vous souhaitez supprimer le rôle de calcul.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis choisissez Rôles IAM.
4. Dans la section Rôle de calcul, choisissez Modifier.
5. Pour supprimer le rôle par défaut, choisissez le X à droite du nom du rôle.
6. Choisissez Enregistrer.

Ajouter un rôle de service avec des autorisations d'accès aux CloudWatch journaux

Amplify envoie des informations sur votre environnement d'exécution SSR à Amazon CloudWatch Logs dans votre. Compte AWS Lorsque vous déployez une application SSR, celle-ci nécessite un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.

Si vous choisissez d'autoriser Amplify à créer un rôle IAM pour vous, le rôle aura déjà les autorisations nécessaires pour créer des journaux. CloudWatch Si vous créez votre propre rôle IAM, vous devrez ajouter les autorisations suivantes à votre politique pour permettre à Amplify d'accéder à Amazon CloudWatch Logs.

```
logs:CreateLogStream
logs:CreateLogGroup
logs:DescribeLogGroups
```

```
logs:PutLogEvents
```

Webhooks unifiés pour les référentiels Git

Amplify Hosting utilise des webhooks pour lancer automatiquement une compilation après un nouveau commit dans votre dépôt Git. La fonctionnalité de webhooks unifiés améliore les intégrations d'Amplify avec les fournisseurs Git et vous permet de connecter davantage d'applications Amplify à un seul référentiel. Grâce aux webhooks unifiés, Amplify utilise désormais un seul webhook par région pour toutes les applications associées dans votre référentiel. Par exemple, si votre référentiel est connecté à des applications situées à la fois dans les régions USA Est (Virginie du Nord) et USA Ouest (Oregon), vous disposerez de deux webhooks unifiés.

Avant cette version, Amplify créait un nouveau webhook pour chaque application associée à un référentiel. Si vous disposiez de plusieurs applications dans un seul référentiel, vous pourriez atteindre les limites de webhook imposées par les différents fournisseurs Git et vous pourriez être empêché d'ajouter d'autres applications. Cela s'est avéré particulièrement difficile pour les équipes travaillant dans des monorepos, où plusieurs projets existent dans un seul référentiel.

Les webhooks unifiés offrent les avantages suivants :

- Surmontez les limites du webhook du fournisseur Git : vous pouvez connecter autant d'applications Amplify que nécessaire à un seul référentiel.
- Support monorepo amélioré : vous bénéficiez de plus de flexibilité et d'efficacité lorsque vous travaillez avec monorepos, où plusieurs projets partagent un seul référentiel.
- Gestion simplifiée : la gestion de plusieurs applications Amplify à l'aide d'un seul webhook de référentiel réduit la complexité et les points de défaillance potentiels.
- Intégration améliorée des flux de travail : vous pouvez utiliser les webhooks alloués par votre fournisseur Git pour d'autres flux de travail essentiels de votre processus de développement.

Commencer à utiliser les webhooks unifiés

Création d'une nouvelle application

Lorsque vous déployez une nouvelle application sur Amplify Hosting à partir d'un référentiel Git, la fonctionnalité de webhooks unifiés est automatiquement implémentée pour votre référentiel. Pour obtenir des instructions sur la création d'une nouvelle application, consultezCommencer à déployer une application sur Amplify Hosting.

Mettre à jour une application existante

Pour les applications Amplify existantes, vous devez reconnecter votre référentiel Git à votre application pour remplacer les webhooks existants par un webhook unifié. Si vous avez déjà atteint le nombre maximum de webhooks autorisé par votre fournisseur Git, la migration vers le webhook unifié risque d'échouer. Dans ce cas, supprimez manuellement au moins un webhook existant avant de vous reconnecter.

Un référentiel peut contenir plusieurs applications déployées dans différentes AWS régions. Les opérations Amplify étant basées sur les régions, la migration vers un webhook unifié ne s'effectue que pour les webhooks de la région dans laquelle vous avez reconnecté votre application Amplify. Par conséquent, vous pouvez voir à la fois des webhooks basés sur un identifiant d'application et des webhooks unifiés basés sur des régions dans votre référentiel.

Suivez les instructions suivantes pour migrer une application Amplify existante vers un webhook unifié.

Pour migrer une application Amplify existante vers un webhook unifié

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application que vous souhaitez migrer vers un webhook unifié.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Paramètres de la branche.
4. Sur la page des paramètres de la branche, choisissez Reconnecter le référentiel.
5. Pour vérifier la réussite de la migration vers le webhook unifié, accédez aux paramètres du webhook dans votre référentiel Git. Vous ne devriez voir apparaître qu'une seule URL de webhook au format`https://amplify-webhooks.`*`Region`*`.amazonaws.com/`*`git-provider`*.

Sécurité dans Amplify

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez de centres de données et d'architectures 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 qui s'appliquent à AWS Amplify, voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amplify. Les rubriques suivantes vous montrent comment configurer Amplify pour atteindre vos objectifs de sécurité et de conformité. Vous apprenez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources Amplify.

Rubriques

Identity and Access Management pour Amplify

Gestion des identités et des accès AWS (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 Amplify. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Rubriques

Public ciblé

La façon dont vous utilisez Gestion des identités et des accès AWS (IAM) varie en fonction de votre rôle :

- Utilisateur du service : demandez des autorisations à votre administrateur si vous ne pouvez pas accéder aux fonctionnalités (voir Résolution des problèmes Amplify identity and access)
- Administrateur du service : déterminez l'accès des utilisateurs et soumettez les demandes d'autorisation (voir Comment Amplify fonctionne avec IAM)
- Administrateur IAM : rédigez des politiques pour gérer l'accès (voir Exemples de politiques basées sur l'identité pour Amplify)

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 en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en tant qu'identité fédérée à l'aide d'informations d'identification provenant d'une source d'identité telle que AWS IAM Identity Center (IAM Identity Center), d'une authentification unique ou d'informations d'identification. Google/Facebook Pour plus d'informations

sur la connexion, consultez Connexion à votre Compte AWS dans le Guide de l'utilisateur Connexion à AWS .

Pour l'accès par programmation, AWS fournit un SDK et une CLI pour signer les demandes de manière cryptographique. Pour plus d'informations, consultez Signature AWS Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une seule identité de connexion appelée utilisateur Compte AWS root qui dispose d'un accès complet à toutes Services AWS les ressources. Il est vivement déconseillé d'utiliser l'utilisateur racine pour vos tâches quotidiennes. Pour les tâches qui requièrent des informations d'identification de l'utilisateur racine, consultez Tâches qui requièrent les informations d'identification de l'utilisateur racine dans le Guide de l'utilisateur IAM.

Identité fédérée

Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à Services AWS l'aide d'informations d'identification temporaires.

Une identité fédérée est un utilisateur provenant de l'annuaire de votre entreprise, de votre fournisseur d'identité Web ou Directory Service qui y accède à Services AWS l'aide d'informations d'identification provenant d'une source d'identité. Les identités fédérées 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. Pour plus d'informations, 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 qui dispose d'autorisations spécifiques pour une seule personne ou application. Nous vous recommandons d'utiliser ces informations d'identification temporaires au lieu des utilisateurs IAM avec des informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le guide de l'utilisateur IAM.

Les groupes IAM spécifient une collection d'utilisateurs IAM et permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. 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 dotée d'autorisations spécifiques qui fournit des informations d'identification temporaires. Vous pouvez assumer un rôle en passant d'un rôle utilisateur à un rôle IAM (console) ou en appelant une opération AWS CLI ou AWS API. Pour plus d'informations, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM sont utiles pour l'accès des utilisateurs fédérés, les autorisations temporaires des utilisateurs IAM, les accès intercompte, les accès entre services et les applications exécutées sur Amazon EC2. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Gestion de l'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 définit les autorisations lorsqu'elles sont associées à une identité ou à une ressource. AWS évalue ces politiques lorsqu'un directeur fait une demande. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations les documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

À l'aide de politiques, les administrateurs précisent qui a accès à quoi en définissant 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. Un administrateur IAM crée des politiques IAM et les ajoute aux rôles, que les utilisateurs peuvent ensuite assumer. Les politiques IAM définissent les autorisations quelle que soit la méthode que vous utilisez pour exécuter l'opération.

Politiques basées sur l'identité

Les stratégies basées sur l'identité sont des documents de stratégie d'autorisations JSON que vous attachez à une identité (utilisateur, groupe ou rôle). Ces politiques contrôlent les actions que peuvent exécuter ces identités, 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 des politiques intégrées (intégrées directement dans une seule identité) ou des politiques gérées (politiques autonomes associées à plusieurs identités).

Pour découvrir comment choisir entre des politiques gérées et 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. Les exemples incluent les politiques de confiance de rôle IAM et les stratégies de compartiment Amazon S3. 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. Vous devez spécifier un principal dans une politique basée sur les ressources.

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.

Autres types de politique

AWS prend en charge des types de politiques supplémentaires qui peuvent définir les autorisations maximales accordées par les types de politiques les plus courants :

- Limites d'autorisations : une limite des autorisations définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCPs) — Spécifiez les autorisations maximales pour une organisation ou une unité organisationnelle dans AWS Organizations. Pour plus d'informations, consultez Politiques de contrôle de service dans le Guide de l'utilisateur AWS Organizations .
- Politiques de contrôle des ressources (RCPs) : définissez le maximum d'autorisations disponibles pour les ressources de vos comptes. Pour plus d'informations, voir Politiques de contrôle des ressources (RCPs) dans le guide de AWS Organizations l'utilisateur.
- Politiques de session : politiques avancées que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. 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 Amplify fonctionne avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amplify, découvrez quelles fonctionnalités IAM peuvent être utilisées avec Amplify.

Fonctionnalités IAM que vous pouvez utiliser avec Amplify

Fonctionnalité IAM	Amplifier le support
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 (identifications dans les politiques)	Partielle
Informations d'identification temporaires	Oui
Transmission des sessions d'accès (FAS)	Oui
Rôles de service	Oui
Rôles liés à un service	Non

Pour obtenir une vue d'ensemble de la façon dont Amplify 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 Amplify

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. 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 Amplify

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Politiques basées sur les ressources dans Amplify

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. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions politiques pour Amplify

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. Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour une liste des actions Amplify, voir Actions définies par AWS Amplify dans la référence d'autorisation de service.

Les actions politiques dans Amplify utilisent le préfixe suivant avant l'action :

```
amplify
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "amplify:action1",
      "amplify:action2"
         ]
```

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Ressources relatives aux politiques pour Amplify

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. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Pour les

actions qui ne sont pas compatibles avec les autorisations de niveau ressource, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour une liste des types de ressources Amplify et de leurs caractéristiques ARNs, voir Types de ressources définis par AWS Amplify dans la référence d'autorisation de service. Pour savoir grâce à quelles actions vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par AWS Amplify.

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Clés de conditions de politique pour Amplify

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` indique à quel moment les instructions s'exécutent en fonction de critères définis. 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. 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 une liste des clés de condition Amplify, voir Clés de condition pour AWS Amplify dans la référence d'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez la section Actions définies par AWS Amplify.

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Listes de contrôle d'accès (ACLs) dans Amplify

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 Amplify

Prend en charge ABAC (identifications dans les politiques) : partiellement

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction des attributs nommés balise. Vous pouvez associer des balises aux entités et aux AWS ressources IAM, puis concevoir des politiques ABAC pour autoriser les opérations lorsque la balise du principal correspond à la balise de la ressource.

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 avec Amplify

Prend en charge les informations d'identification temporaires : oui

Les informations d'identification temporaires fournissent un accès à court terme aux AWS ressources et sont automatiquement créées lorsque vous utilisez la fédération ou que vous changez de rôle. 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 et Services AWS compatibles avec IAM dans le Guide de l'utilisateur IAM.

Transférer les sessions d'accès pour Amplify

Prend en charge les sessions d'accès direct (FAS) : oui

Les sessions d'accès direct (FAS) utilisent les autorisations du principal appelant et Service AWS, combinées Service AWS à la demande d'envoi de demandes aux services en aval. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez la section Sessions de transmission d'accès.

Rôles de service pour Amplify

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.

> **Warning**
>
> La modification des autorisations pour un rôle de service peut interrompre les fonctionnalités d'Amplify. Modifiez les rôles de service uniquement lorsque Amplify fournit des instructions à cet effet.

Rôles liés à un service pour Amplify

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 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 à un service, consultez la section AWS Services compatibles avec IAM dans le Guide de l'utilisateur d'IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôle lié à un service. Cliquez sur le lien Oui pour consulter la documentation relative aux rôles liés à un service pour ce service.

Exemples de politiques basées sur l'identité pour Amplify

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier des ressources Amplify. 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.

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 Amplify, y compris le format de ARNs pour chacun des types de ressources, voir Actions, ressources et clés de condition AWS Amplify dans la référence d'autorisation de service.

Rubriques

- Bonnes pratiques en matière de politiques
- Utilisation de la console Amplify
- 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 Amplify 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 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 Amplify

Pour accéder à la AWS Amplify console, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et d'afficher les détails des ressources Amplify de 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.

Avec la sortie d'Amplify Studio, la suppression d'une application ou d'un backend nécessite à la fois `amplify` des autorisations et des autorisations. `amplifybackend` Si une politique IAM fournit uniquement des `amplify` autorisations, un utilisateur reçoit une erreur d'autorisation lorsqu'il tente de supprimer une application. Si vous êtes un administrateur qui rédige des politiques, déterminez les autorisations appropriées à accorder aux utilisateurs qui doivent effectuer des actions de suppression.

Pour garantir que les utilisateurs et les rôles peuvent toujours utiliser la console Amplify, associez également la politique Amplify `ConsoleAccess` ou `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": "*"
        }
    ]
```

```
}
```

AWS politiques gérées pour AWS Amplify

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

Politique gérée par AWS : AdministratorAccess -Amplify

Vous pouvez associer la politique `AdministratorAccess-Amplify` à vos identités IAM. Amplify associe également cette politique à un rôle de service qui permet à Amplify d'effectuer des actions en votre nom.

Lorsque vous déployez un backend dans la console Amplify, vous devez créer `Amplify-Backend Deployment` un rôle de service qu'Amplify utilise pour créer et gérer des ressources. AWS IAM associe la politique `AdministratorAccess-Amplify` gérée au rôle de `Amplify-Backend Deployment` service.

Cette politique accorde des autorisations administratives aux comptes tout en autorisant explicitement l'accès direct aux ressources dont les applications Amplify ont besoin pour créer et gérer les backends.

Détails de l'autorisation

Cette politique donne accès à plusieurs AWS services, y compris aux actions IAM. Ces actions permettent aux identités soumises à cette politique d'utiliser Gestion des identités et des accès AWS pour créer d'autres identités avec n'importe quelle autorisation. Cela permet une escalade des autorisations et cette politique doit être considérée comme aussi puissante que la `AdministratorAccess` politique.

Cette politique accorde l'autorisation `iam:PassRole` d'action pour toutes les ressources. Cela est nécessaire pour prendre en charge la configuration des groupes d'utilisateurs Amazon Cognito.

Pour consulter les autorisations associées à cette politique, voir AdministratorAccess-Amplify dans le manuel AWS Managed Policy Reference.

AWS politique gérée : AmplifyBackendDeployFullAccess

Vous pouvez associer la politique `AmplifyBackendDeployFullAccess` à vos identités IAM.

Cette politique accorde à Amplify des autorisations d'accès complètes pour déployer les ressources du backend Amplify à l'aide du. AWS Cloud Development Kit (AWS CDK) Les autorisations sont reportées aux AWS CDK rôles dotés des autorisations `AdministratorAccess` politiques nécessaires.

Détails de l'autorisation

Cette politique inclut les autorisations permettant d'effectuer les opérations suivantes.

- `Amplify`— Récupérez les métadonnées relatives aux applications déployées.
- `CloudFormation`— Créez, mettez à jour et supprimez les piles gérées par Amplify.
- `SSM`— Créez, mettez à jour et supprimez le magasin `String` et les paramètres SSM gérés par Amplify. `SecureString`
- `AWS AppSync`— Mettez à jour et récupérez les ressources AWS AppSync du schéma, du résolveur et des fonctions. L'objectif est de prendre en charge la fonctionnalité de hotswapping du sandbox de 2e génération.
- `Lambda`— Mettez à jour et récupérez la configuration des fonctions gérées par Amplify. L'objectif est de prendre en charge la fonctionnalité de hotswapping du sandbox de 2e génération.

  Récupérez les balises d'une fonction Lambda. L'objectif est de prendre en charge les fonctions Lambda définies par les clients.
- `Amazon S3`— Récupérez les ressources de déploiement d'Amplify.

- `AWS Security Token Service`— Permet à la AWS Cloud Development Kit (AWS CDK) CLI d'assumer le rôle de déploiement.
- `Amazon RDS`— Lisez les métadonnées des instances de base de données, des clusters et des proxys.
- `Amazon EC2`— Lisez les informations de zone de disponibilité d'un sous-réseau.
- `CloudWatch Logs`— Récupère les journaux de la fonction Lambda d'un client. L'objectif est de permettre à un environnement de sandbox de développement dans le cloud Amplify de diffuser les journaux d'une fonction Lambda sur le terminal d'un client.

Pour voir les autorisations de cette stratégie, consultez AmplifyBackendDeployFullAccess dans le AWS Guide de référence des stratégies gérées par.

Amplifier les mises à jour des politiques gérées AWS

Consultez les détails des mises à jour des politiques AWS gérées pour Amplify depuis que ce service a commencé à suivre ces modifications. Pour obtenir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS de la page Historique du document pour AWS Amplify.

Modifier	Description	Date
AmplifyBackendDeployFullAccess – Mise à jour d'une stratégie existante	Ajoutez un accès en lecture à la `logs:FilterLogEvents` ressource pour permettre à Amplify de diffuser les journaux à partir des fonctions dans lesquelles un groupe de journaux personnalisé a été créé. Il s'agit d'une extension de la capacité existante de diffuser les journaux d'une fonction Lambda.	14 novembre 2024
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez un accès en lecture aux `logs:FilterLogEvents` ressources `lambda:Li`	18 juillet 2024

Modifier	Description	Date
	`stTags` et pour prendre en charge les fonctions Lambda définies par les clients. Ces autorisations permettent à un environnement de sandbox de développement dans le cloud Amplify de diffuser les journaux d'une fonction Lambda sur le terminal d'un client.	
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez un accès en lecture à la `arn:aws:ssm:*:*:parameter/cdk-bootstrap/*` ressource pour permettre à Amplify de détecter la version bootstrap du CDK dans le compte d'un client.	31 mai 2024

Modifier	Description	Date
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez une nouvelle déclaration `AmplifyDiscoverRDSVpcConfig` de politique avec des autorisations en lecture seule Amazon RDS et Amazon EC2 définies par les conditions des ressources et du compte. Ces autorisations prennent en charge la `npx amplify generate schema-from-database` commande Amplify Gen 2 qui permet aux clients de générer un schéma de données Typescript à partir d'une base de données SQL existante. Ajoutez les `ec2:DescribeSubnets` autorisations `rds:DescribeDBProxies` `rds:DescribeDBInstances` `rds:DescribeDBClusters` ,`rds:DescribeDBSubnetGroups` ,, et. La `npx amplify generate schema-from-database` commande nécessite ces autorisations pour vérifier si un hôte de base de données spécifié est hébergé sur Amazon RDS et pour générer automatiquement la	17 avril 2024

Modifier	Description	Date
	configuration Amazon VPC requise pour fournir les autres ressources requises pour configurer une AWS AppSync API basée sur une base de données SQL.	
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez l'action `cloudformation:DeleteStack` politique pour prendre en charge la suppression de la pile lorsque l'`DeleteBranch` API est appelée. Ajoutez l'action `lambda:GetFunction` politique pour prendre en charge les fonctions de hotswapping. Ajoutez l'action de `lambda:UpdateFunctionConfiguration` politique pour prendre en charge les mises à jour de la fonction Lambda.	5 avril 2024
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez les `cloudformation:UnTagResource` autorisations `cloudformation:TagResource` et pour prendre en charge les appels à CloudFormation APIs.	4 avril 2024

Modifier	Description	Date
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez l'action `lambda:InvokeFunction` politique pour prendre en charge le AWS Cloud Development Kit (AWS CDK) hotswapping. Il AWS CDK fait des appels directs à une fonction Lambda pour effectuer le hotswapping des actifs Amazon S3. Ajoutez l'action `lambda:UpdateFunctionCode` politique pour prendre en charge les fonctions de hotswapping.	2 janvier 2024
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez des actions politiques pour soutenir l'`UpdateApiKey` opération. Cela est nécessaire pour permettre le déploiement réussi de l'application après avoir quitté et redémarré le sandbox sans supprimer de ressources.	17 novembre 2023
AmplifyBackendDeployFullAccess : mise à jour d'une politique existante	Ajoutez l'`amplify:GetBackendEnvironment` autorisation de prendre en charge le déploiement de l'application Amplify.	6 novembre 2023

Modifier	Description	Date
AmplifyBackendDeployFullAccess : nouvelle politique	Amplify a ajouté une nouvelle politique avec les autorisations minimales requises pour déployer les ressources du backend Amplify.	8 octobre 2023
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez l'`ecr:DescribeRepositories` autorisation requise par l'interface de ligne de commande (CLI) Amplify.	1er juin 2023

Modifier	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez une action politique pour soutenir la suppressi on des balises d'une AWS AppSync ressource. Ajoutez une action politique pour soutenir la ressource Amazon Polly. Ajoutez une action politique pour prendre en charge la mise à jour de la configuration du OpenSearch domaine. Ajoutez une action politique pour soutenir la suppression des balises d'un Gestion des identités et des accès AWS rôle. Ajoutez une action politique pour soutenir la suppression de balises d'une ressource Amazon DynamoDB. Ajoutez les `cloudfron t:GetCloudFrontOri ginAccessIdentityC onfig` autorisations `cloudfront:GetClou dFrontOriginAccess Identity` et au bloc d'instructions pour prendre `CLISDKCalls` en charge les flux de travail de publication et d'hébergement d'Amplify.	24 février 2023

Modifier	Description	Date
	Ajoutez l'`s3:PutBucketPublicAccessBlock` autorisation au `CLIManageviaCFNPolicy` bloc d'instructions pour permettre à Amazon S3 de AWS CLI respecter les meilleures pratiques de sécurité d'Amazon S3, qui consistent à activer la fonctionnalité Amazon S3 Block Public Access sur les compartiments internes. Ajoutez l'`cloudformation:DescribeStacks` autorisation au bloc d'instructions pour `CLISDKCalls` permettre de récupérer les CloudFormation piles des clients lors de nouvelles tentatives dans le processeur principal Amplify afin d'éviter de dupliquer les exécutions en cas de mise à jour d'une pile. Ajoutez l'`cloudformation:ListStacks` autorisation au bloc de `CLICloudformationPolicy` déclarations. Cette autorisation est requise pour prendre pleinement en charge	

Modifier	Description	Date
	l' CloudFormation DescribeStacks action.	
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions politiques pour permettre à la fonctionnalité de rendu côté serveur Amplify de transférer les métriques de l'application vers celles du client CloudWatch . Compte AWS	30 août 2022
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions politiques pour bloquer l'accès public au compartiment Amazon S3 du déploiement d'Amplify.	27 avril 2022

Modifier	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez une action pour permettre aux clients de supprimer leurs applications de rendu côté serveur (SSR). Cela permet également de supprimer correctement la CloudFront distribution correspondante. Ajoutez une action pour permettre aux clients de spécifier une fonction Lambda différente pour gérer les événements provenant d'une source d'événements existante à l'aide de la CLI Amplify. Avec ces modifications, AWS Lambda vous serez en mesure d'effectuer l'UpdateEventSourceMappingaction.	17 avril 2022
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez une action de politique pour activer les actions Amplify UI Builder sur toutes les ressources.	2 décembre 2021

Modifier	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions politique s pour soutenir la fonctionn alité d'authentification Amazon Cognito qui utilise des fournisseurs d'identité sociale. Ajoutez une action de politique pour prendre en charge les couches Lambda. Ajoutez une action politique pour soutenir la catégorie Amplify Storage.	8 novembre 2021

Modifier	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions Amazon Lex pour soutenir la catégorie Amplify Interactions. Ajoutez des actions Amazon Rekognition pour soutenir la catégorie Amplify Predictions. Ajoutez une action Amazon Cognito pour prendre en charge la configuration MFA sur les groupes d'utilisateurs Amazon Cognito. Ajoutez CloudFormation des actions au support CloudFormation StackSets. Ajoutez des actions Amazon Location Service pour soutenir la catégorie Amplify Geo. Ajoutez une action Lambda pour prendre en charge les couches Lambda dans Amplify. Ajoutez des actions de CloudWatch journalisation pour prendre en charge CloudWatch les événements. Ajoutez des actions Amazon S3 pour soutenir la catégorie Amplify Storage.	27 septembre 2021

Modifier	Description	Date
	Ajoutez des actions politiques pour prendre en charge les applications de rendu côté serveur (SSR).	

Modifier	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Consolidez toutes les actions Amplify en une seule `amplify:*` action. Ajoutez une action Amazon S3 pour prendre en charge le chiffrement des compartiments Amazon S3 des clients. Ajoutez des actions de limite d'autorisation IAM pour prendre en charge les applications Amplify dont les limites d'autorisation sont activées. Ajoutez des actions Amazon SNS pour faciliter l'affichage des numéros de téléphone d'origine et l'affichage, la création, la vérification et la suppression des numéros de téléphone de destination. Amplify Studio : ajoutez Amazon Cognito AWS Lambda, IAM et des actions de politique pour permettre la gestion des backends dans la console Amplify CloudFormation et Amplify Studio. Ajoutez une déclaration de politique AWS Systems Manager (SSM) pour gérer les secrets de l'environnement Amplify.	28 juillet 2021

Modifier	Description	Date
	Ajoutez une CloudFormation `ListResources` action pour prendre en charge les couches Lambda pour les applications Amplify.	
Amplify a commencé à suivre les modifications	Amplify a commencé à suivre les modifications apportées à ses politiques AWS gérées.	28 juillet 2021

Résolution des problèmes Amplify identity and access

Utilisez les informations suivantes pour vous aider à diagnostiquer et à résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation d'Amplify et d'IAM.

Rubriques

- Je ne suis pas autorisé à effectuer une action dans Amplify
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite autoriser des personnes extérieures à mon AWS compte à accéder à mes ressources Amplify

Je ne suis pas autorisé à effectuer une action dans Amplify

Si vous recevez une erreur qui indique que vous n'êtes pas autorisé à effectuer une action, vos politiques doivent être mises à jour afin de vous permettre d'effectuer l'action.

L'exemple d'erreur suivant se produit quand l'utilisateur IAM `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `amplify:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 amplify:GetWidget on resource: my-example-widget
```

Dans ce cas, la politique qui s'applique à l'utilisateur `mateojackson` doit être mise à jour pour autoriser l'accès à la ressource *`my-example-widget`* à l'aide de l'action `amplify:`*`GetWidget`*.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Avec la sortie d'Amplify Studio, la suppression d'une application ou d'un backend nécessite à la fois `amplify` des autorisations et des autorisations. `amplifybackend` Si un administrateur a rédigé une politique IAM qui fournit uniquement des `amplify` autorisations, vous recevrez une erreur d'autorisation lorsque vous tenterez de supprimer une application.

L'exemple d'erreur suivant se produit lorsque l'utilisateur `mateojackson` IAM essaie d'utiliser la console pour supprimer une *`example-amplify-app`* ressource fictive mais ne dispose pas des `amplifybackend:`*`RemoveAllBackends`* autorisations nécessaires.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 amplifybackend;:RemoveAllBackends on resource: example-amplify-app
```

Dans ce cas, Mateo demande à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder à la ressource *`example-amplify-app`* à l'aide de l'action `amplifybackend:`*`RemoveAllBackends`*.

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 transmettre un rôle à Amplify.

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, vous devez 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 Amplify. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary n'est pas autorisée à transmettre 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 autoriser des personnes extérieures à mon AWS compte à accéder à mes ressources Amplify

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 Amplify prend en charge ces fonctionnalités, consultez. Comment Amplify fonctionne 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.

Protection des données dans Amplify

AWS Amplify est conforme au modèle de responsabilité AWS partagée modèle de de , qui inclut des réglementations et des directives pour la protection des données. AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS services. AWS conserve le contrôle des données hébergées sur cette infrastructure, y compris les contrôles de configuration de sécurité pour le traitement du contenu client et des données personnelles. AWS les clients et les partenaires APN,

agissant en tant que contrôleurs ou sous-traitants de données, sont responsables de toutes les données personnelles qu'ils placent dans le AWS Cloud.

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 Gestion des identités et des accès AWS (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.
- 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 au sein AWS des services.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données personnelles stockées dans Amazon S3.

Nous vous recommandons vivement de ne jamais placer d'informations identifiables sensibles, telles que les numéros de compte de vos clients, dans des champs de formulaire comme Nom. Cela inclut lorsque vous travaillez avec Amplify ou d'autres AWS services à l'aide de la console, de l'API ou. AWS CLI AWS SDKs Toutes les données que vous entrez dans Amplify ou dans d'autres services peuvent être récupérées pour être incluses dans les journaux de diagnostic. Lorsque vous fournissez une URL à un serveur externe, n'incluez pas les informations d'identification non chiffrées dans l'URL pour valider votre demande adressée au serveur.

Pour en savoir plus sur la protection des données, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD sur le Blog sur la sécurité d'AWS .

Chiffrement au repos

Le chiffrement au repos consiste à protéger vos données contre tout accès non autorisé en chiffrant les données stockées. Amplify chiffre les artefacts de construction d'une application par défaut en utilisant pour Amazon AWS KMS keys S3 qui sont gérés par le. AWS Key Management Service

Amplify utilise Amazon CloudFront pour proposer votre application à vos clients. CloudFront utilisations SSDs chiffrées pour les points de présence de localisation périphériques (POPs) et volumes EBS chiffrés pour les caches périphériques régionaux ()RECs. Le code de fonction et la configuration dans CloudFront Functions sont toujours stockés dans un format crypté SSDs sur

l'emplacement POPs crypté en périphérie et dans d'autres emplacements de stockage utilisés par CloudFront.

Chiffrement en transit

Le chiffrement en transit consiste à protéger vos données contre l'interception pendant qu'elles se déplacent entre les points de terminaison de communication. Amplify Hosting fournit le cryptage des données en transit par défaut. Toutes les communications entre les clients et Amplify et entre Amplify et ses dépendances en aval sont protégées par des connexions TLS signées à l'aide du processus de signature Signature Version 4. Tous les points de terminaison Amplify Hosting utilisent des certificats SHA-256 gérés par. AWS Autorité de certification privée Pour plus d'informations, consultez les sections Processus de signature de Signature version 4 et Qu'est-ce que c'est AWS Autorité de certification privée.

Gestion des clés de chiffrement

AWS Key Management Service (KMS) est un service géré permettant de créer et de contrôler AWS KMS keys les clés de chiffrement utilisées pour chiffrer les données des clients. AWS Amplify génère et gère des clés cryptographiques pour chiffrer les données pour le compte des clients. Il n'y a aucune clé de chiffrement à gérer.

Validation de conformité pour AWS Amplify

Des auditeurs tiers évaluent la sécurité et AWS Amplify la conformité de plusieurs programmes de AWS conformité. Il s'agit notamment du SOC, du PCI, de l'ISO, de l'HIPAA, du MTCS, du C5, du K-ISMS, de l'ENS High, de l'OSPAR, du HITRUST CSF et de la FINMA.

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

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 .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. Pour plus d'informations sur votre responsabilité en matière de conformité lors de l'utilisation Services AWS, consultez AWS la documentation de sécurité.

Sécurité de l'infrastructure dans AWS Amplify

En tant que service géré, AWS Amplify 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 à Amplify 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.

Enregistrement et surveillance des événements de sécurité dans Amplify

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amplify et de vos autres AWS solutions. AWS fournit les outils de surveillance suivants pour surveiller Amplify, signaler un problème et prendre des mesures automatiques le cas échéant :

- Amazon CloudWatch surveille en temps réel vos AWS ressources et les applications que vous utilisez AWS. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés et définir des alarmes qui vous avertissent ou prennent des mesures lorsqu'une certaine métrique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation du processeur ou d'autres indicateurs de vos instances Amazon Elastic Compute Cloud (Amazon EC2) et lancer automatiquement de nouvelles instances en cas de besoin. Pour plus d'informations sur l'utilisation CloudWatch des métriques et des alarmes avec Amplify, consultez. Surveillance d'une application Amplify
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d'instances Amazon EC2 et d'autres sources. AWS 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 AWS compte de votre compte et transmet les fichiers journaux à un compartiment Amazon Simple Storage Service (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 de plus amples informations, veuillez consulter Journalisation des appels d'API Amplify à l'aide AWS CloudTrail.
- Amazon EventBridge est un service de bus d'événements sans serveur qui permet de connecter facilement vos applications à des données provenant de diverses sources. EventBridge fournit un flux de données en temps réel à partir de vos propres applications, applications Software-as-a-Service (SaaS) et AWS services, et achemine ces données vers des cibles telles que AWS Lambda. Cela vous permet de surveiller les événements qui se produisent dans les services et de créer des architectures axées sur les événements. Pour plus d'informations, consultez le guide de EventBridge l'utilisateur Amazon.

Prévention du problème de l'adjoint confus entre services

Le problème de député confus est un problème de sécurité dans lequel une entité qui n'est pas autorisée à effectuer une action peut contraindre une entité plus privilégiée à le faire. En AWS, l'usurpation d'identité interservices peut entraîner la confusion des adjoints. L'usurpation d'identité entre services peut se produire lorsqu'un service (le service appelant) appelle un autre service (le service appelé). Le service appelant peut être manipulé et ses autorisations utilisées pour agir sur les ressources d'un autre client auxquelles on ne serait pas autorisé à accéder autrement. Pour éviter cela, AWS fournit des outils qui vous aident à protéger vos données pour tous les services avec des principaux de service qui ont eu accès aux ressources de votre compte.

Nous recommandons d'utiliser les clés de contexte de condition `aws:SourceAccount`globale `aws:SourceArn`et les clés contextuelles dans les politiques de ressources afin de limiter les autorisations qui AWS Amplify accordent un autre service à la ressource. Si vous utilisez les deux clés de contexte de condition globale, la valeur `aws:SourceAccount` et le compte de la valeur `aws:SourceArn` doit utiliser le même ID de compte lorsqu'il est utilisé dans la même déclaration de stratégie.

La valeur de `aws:SourceArn` doit être l'ARN de branche de l'application Amplify. Spécifiez cette valeur dans le format`arn:`*`Partition`*`:amplify:`*`Region`*`:`*`Account`*`:apps/`*`AppId`*`/branches/`*`BranchName`*.

Le moyen le plus efficace de se protéger contre le problème de député confus consiste à utiliser la clé de contexte de condition globale `aws:SourceArn` avec l'ARN complet de la ressource. Si vous ne connaissez pas l'ARN complet de la ressource ou si vous spécifiez plusieurs ressources, utilisez la clé de contexte de condition globale `aws:SourceArn` avec des caractères génériques (*) pour les parties inconnues de l'ARN. Par exemple, `arn:aws:`*`servicename`*`::`*`123456789012`*`:*`.

L'exemple suivant montre une politique de confiance dans les rôles que vous pouvez appliquer pour limiter l'accès à n'importe quelle application Amplify de votre compte et éviter le problème de confusion des adjoints. Pour utiliser cette politique, remplacez le texte en italique rouge dans l'exemple de politique par vos propres informations.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": [
          "amplify.me-south-1.amazonaws.com",
          "amplify.eu-south-1.amazonaws.com",
          "amplify.ap-east-1.amazonaws.com",
          "amplifybackend.amazonaws.com",
          "amplify.amazonaws.com"
        ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

L'exemple suivant montre une politique de confiance dans les rôles que vous pouvez appliquer pour limiter l'accès à une application Amplify spécifiée dans votre compte et éviter le problème de confusion des adjoints. Pour utiliser cette politique, remplacez le texte en italique rouge dans l'exemple de politique par vos propres informations.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": [
          "amplify.me-south-1.amazonaws.com",
          "amplify.eu-south-1.amazonaws.com",
          "amplify.ap-east-1.amazonaws.com",
          "amplifybackend.amazonaws.com",
          "amplify.amazonaws.com"
        ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/d123456789/
branches/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

Bonnes pratiques de sécurité pour Amplify

Amplify fournit un certain nombre de fonctionnalités de sécurité à prendre en compte lorsque vous développez et mettez en œuvre vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou

suffisantes pour votre environnement, considérez-les comme des recommandations utiles plutôt que comme des prescriptions.

Utilisation de cookies avec le domaine par défaut Amplify

Lorsque vous utilisez Amplify pour déployer une application Web, Amplify l'héberge pour vous sur le domaine par défaut. `amplifyapp.com` Vous pouvez afficher votre application sur une URL au format. `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com`

Pour renforcer la sécurité de vos applications Amplify, le domaine amplifyapp.com est enregistré dans la liste des suffixes publics (PSL). Pour plus de sécurité, nous vous recommandons d'utiliser des cookies avec un `__Host-` préfixe si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Amplify. Cette pratique vous aidera à protéger votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Quotas du service Amplify Hosting

Les quotas de service pour l' AWS Amplify hébergement sont les suivants. Les quotas de service (précédemment appelés limites) sont le nombre maximum de ressources de service ou d'opérations pour votre Compte AWS.

Comptes AWS Les nouveautés ont réduit les quotas d'applications et de tâches simultanées. AWS augmente automatiquement ces quotas en fonction de votre utilisation. Vous pouvez également demander une augmentation de quota.

La console Service Quotas fournit des informations sur les quotas de votre compte. Vous pouvez utiliser la console Service Quotas pour afficher les quotas par défaut et demander des augmentations de quota pour les quotas ajustables. Pour de plus amples informations, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Name	Par défaut	Ajusta	Description
Applications	Chaque région prise en charge : 25	Oui	Le nombre maximum d'applications que vous pouvez créer dans AWS Amplify Console dans ce compte dans la région actuelle.
Branches par application	Chaque région prise en charge : 50	Non	Nombre maximal de branches par application que vous pouvez créer dans ce compte dans la région actuelle.
Taille de l'artefact de génération	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) d'un artefact de génération d'applica tion. Un artefact de build est déployé par AWS Amplify Console après une compilation.

Name	Par défaut	Ajusta	Description
Taille de l'artefact du cache	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) d'un artefact de cache.
Tâches simultanées	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches simultanées que vous pouvez créer dans ce compte dans la région actuelle.
Domaines par application	Chaque Région prise en charge : 5	Oui	Nombre maximal de domaines par application que vous pouvez créer dans ce compte dans la région actuelle.
Taille de l'artefact du cache d'environnement	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) de l'artefact du cache d'environnement.
Taille du fichier ZIP de déploiement manuel	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) d'un fichier ZIP de déploiement manuel.
Nombre maximum de créations d'applications par heure	Chaque région prise en charge : 25	Non	Le nombre maximum d'applications que vous pouvez créer dans AWS Amplify Console par heure sur ce compte dans la région actuelle.

Name	Par défaut	Ajusta	Description
Demander des jetons par seconde	Chaque Région prise en charge : 20 000	Oui	Le nombre maximum de jetons de demande par seconde pour une application. Amplify Hosting alloue des jetons aux demandes en fonction de la quantité de ressources (temps de traitement et transfert de données) qu'elles consomment.
Sous-domaines par domaine	Chaque région prise en charge : 50	Non	Nombre maximal de sous-domaines par domaine que vous pouvez créer dans ce compte dans la région actuelle.
Webhooks par application	Chaque Région prise en charge : 50	Oui	Nombre maximal de webhooks par applicati on que vous pouvez créer dans ce compte dans la région actuelle.

Pour plus d'informations sur les quotas de service Amplify, consultez la section AWS Amplify Points de terminaison et quotas dans le. Références générales AWS

Résolution des problèmes liés à Amplify Hosting

Si vous rencontrez des erreurs ou des problèmes de déploiement lorsque vous travaillez avec Amplify Hosting, consultez les rubriques de cette section.

Rubriques

Résolution des problèmes généraux liés à Amplify

Les informations suivantes peuvent vous aider à résoudre les problèmes généraux liés à Amplify Hosting.

Rubriques

Code d'état HTTP 429 (trop de requêtes)

Amplify contrôle le nombre de requêtes par seconde (RPS) adressées à votre site Web en fonction du temps de traitement et du transfert de données consommés par les demandes entrantes. Si

votre application renvoie un code d'état HTTP 429, les demandes entrantes dépassent le temps de traitement et de transfert de données alloué à votre application. Cette limite d'applications est gérée par le quota de `REQUEST_TOKENS_PER_SECOND` service d'Amplify. Pour plus d'informations sur les quotas, consultez Quotas du service Amplify Hosting.

Pour résoudre ce problème, nous vous recommandons d'optimiser votre application afin de réduire la durée des demandes et le transfert de données afin d'augmenter le RPS de l'application. Par exemple, avec les mêmes 20 000 jetons, une page SSR hautement optimisée qui répond dans les 100 millisecondes peut supporter un RPS plus élevé qu'une page avec une latence supérieure à 200 millisecondes.

De même, une application qui renvoie une taille de réponse de 1 Mo consommera plus de jetons qu'une application qui renvoie une taille de réponse de 250 Ko.

Nous vous recommandons également de tirer parti du CloudFront cache Amazon en configurant Cache-Control des en-têtes qui maximisent le temps pendant lequel une réponse donnée est conservée dans le cache. Les demandes traitées depuis le CloudFront cache ne sont pas prises en compte dans le calcul de la limite de débit. Chaque CloudFront distribution peut traiter jusqu'à 250 000 requêtes par seconde, ce qui vous permet de faire évoluer votre application de manière très élevée en utilisant le cache. Pour plus d'informations sur le CloudFront cache, consultez Optimisation de la mise en cache et de la disponibilité dans le manuel Amazon CloudFront Developer Guide.

La console Amplify n'affiche pas l'état de construction et l'heure de la dernière mise à jour de mon application

Lorsque vous accédez à la page Toutes les applications de la console Amplify, une vignette s'affiche pour chacune de vos applications dans la région actuelle. Si l'état de la version, tel que Déployé, et l'heure de la dernière mise à jour ne s'affichent pas pour une application, aucune branche d'`Production`étape n'est associée à l'application.

Pour répertorier les applications de la console, Amplify utilise l'`ListApps`API. Amplify utilise l'`ProductionBranch.status`attribut pour afficher l'état de la construction et l'`ProductionBranch.lastDeployTime`attribut pour afficher l'heure de la dernière mise à jour. Pour plus d'informations sur cette API, consultez la documentation ProductionBranchde l'API d'hébergement Amplify.

Suivez les instructions ci-dessous pour associer une `Production` étape à la branche de votre application.

1. Connectez-vous à la console Amplify.
2. Sur la page Toutes les applications, choisissez l'application que vous souhaitez mettre à jour.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Paramètres de la branche.
4. Dans la section Paramètres de la branche, choisissez Modifier.
5. Pour Branche de production, choisissez le nom de branche que vous souhaitez utiliser.
6. Choisissez Enregistrer.
7. Retournez à la page Toutes les applications. L'état de construction et l'heure de la dernière mise à jour devraient maintenant être affichés pour votre application.

Les aperçus Web ne sont pas créés pour les nouvelles pull requests

La fonctionnalité d'aperçu Web vous permet de prévisualiser les modifications apportées par les pull requests avant de les fusionner dans une branche d'intégration. Un aperçu Web déploie chaque pull request envoyée à votre référentiel vers une URL d'aperçu unique, différente de l'URL utilisée par votre site principal.

Si vous avez activé les aperçus Web pour votre application, mais qu'ils ne sont pas créés pour une nouvelle application PRs, déterminez si l'une des causes suivantes est à l'origine de votre problème.

1. Vérifiez si votre application a atteint le quota de `Branches per app` service maximal. Pour plus d'informations sur les quotas, consultez Quotas du service Amplify Hosting.

   Pour respecter le quota par défaut de 50 succursales par application, pensez à activer la suppression automatique des branches dans votre application. Cela vous évitera d'accumuler dans votre compte des branches qui n'existent plus dans votre référentiel.

2. Si vous utilisez un GitHub référentiel public et que votre application Amplify est associée à un rôle de service IAM, Amplify ne crée pas d'aperçu pour des raisons de sécurité. Par exemple, les applications dotées de backends et les applications déployées sur la plate-forme `WEB_COMPUTE` d'hébergement nécessitent un rôle de service IAM. Par conséquent, vous ne pouvez pas activer les aperçus Web pour ces types d'applications si leur référentiel est public.

   Pour que les aperçus Web fonctionnent pour votre application, vous pouvez soit dissocier le rôle de service (si l'application ne possède pas de backend ou n'en est pas une`WEB_COMPUTE`), soit rendre le GitHub référentiel privé.

Mon déploiement manuel est bloqué avec un statut en attente dans la console Amplify

Les déploiements manuels vous permettent de publier votre application Web avec Amplify Hosting sans connecter un fournisseur Git. Vous pouvez utiliser l'une des quatre options de déploiement suivantes.

1. Faites glisser et déposez le dossier de votre application dans la console Amplify.
2. Glissez et déposez un fichier .zip (contenant les artefacts de construction de votre site) dans la console Amplify.
3. Téléchargez un fichier .zip (qui contient les artefacts de construction de votre site) dans un compartiment Amazon S3 et connectez le compartiment à une application dans la console Amplify.
4. Utilisez une URL publique qui pointe vers un fichier .zip (contenant les artefacts de construction de votre site) dans la console Amplify.

Nous sommes conscients des problèmes liés à la fonctionnalité glisser-déposer lors de l'utilisation d'un dossier d'application pour un déploiement manuel dans la console Amplify. Ces déploiements peuvent échouer pour les raisons suivantes.

- Des problèmes réseau transitoires se produisent.
- Une modification locale est apportée aux fichiers lors du téléchargement.
- La session du navigateur tente de télécharger simultanément un grand nombre de ressources statiques.

Alors que nous nous efforçons d'améliorer la fiabilité de nos téléchargements par glisser-déposer, nous vous recommandons d'utiliser un fichier .zip au lieu de glisser-déposer les dossiers de l'application.

Nous vous recommandons vivement de télécharger un fichier .zip dans un compartiment Amazon S3, car cela évite le téléchargement de fichiers depuis la console Amplify et améliore la fiabilité des déploiements manuels. L'intégration d'Amplify à Amazon S3 simplifie ce processus. Pour de plus amples informations, veuillez consulter Déploiement d'un site Web statique vers Amplify à partir d'un compartiment Amazon S3.

Je dois mettre à jour la version Node.js de mon application

Amplify prend fin le 15 septembre 2025 pour les applications utilisant les versions 14, 16 et 18 de Node.js. Le comportement après cette date dépend de votre type d'application :

- Applications SSR : des échecs de compilation se produiront lors de l'utilisation de versions obsolètes de Node.js. Vous ne pourrez pas déployer de mises à jour tant que vous n'aurez pas effectué la mise à niveau vers Node.js 20 ou version ultérieure.
- Applications non SSR : vous pouvez continuer à utiliser les versions obsolètes de Node.js si vous les installez manuellement par le biais de buildspec ou de mises à jour de packages en direct.

Les applications déjà déployées continueront de fonctionner quelle que soit la version de Node.js.

Si vous utilisez l'image de build Amazon Linux 2023, la version 20 de Node.js est prise en charge par défaut. À compter du 15 septembre 2025, l' AL2023 image prendra automatiquement en charge Node.js 22 et changera sa version par défaut de Node.js de 18 à 22.

Amazon Linux 2 (AL2) ne prend pas automatiquement en charge la version 20 ou ultérieure de Node.js. Si vous utilisez actuellement AL2, nous vous recommandons de passer à AL2023. Vous pouvez modifier l'image de construction dans la console Amplify. Vous pouvez également utiliser une image de construction personnalisée qui prend en charge la version de Node.js que vous spécifiez.

Avant de procéder à la mise à niveau, nous vous recommandons de tester votre application sur une nouvelle branche pour vérifier qu'elle fonctionne correctement.

Options de mise à niveau

Console Amplify

Vous pouvez utiliser la fonctionnalité de mise à jour des packages en direct de la console Amplify pour spécifier la version de Node.js à utiliser. Pour obtenir des instructions, veuillez consulter Utilisation de versions de package et de dépendance spécifiques dans l'image de construction.

Image de construction personnalisée

Si vous utilisez une image de construction personnalisée et que NVM est installé sur votre image, vous pouvez l'ajouter `nvm install 20` à votre Dockerfile. Pour en savoir plus sur les exigences et les instructions de configuration relatives à une image de construction personnalisée, consultezPersonnalisation de l'image de construction.

Paramètres de construction

Vous pouvez spécifier la version de Node.js à utiliser dans les paramètres de `amplify.yml` compilation de votre application, en ajoutant la `nvm use` commande dans la section des commandes de pré-construction. Pour obtenir des instructions sur la mise à jour des paramètres de compilation d'une application, consultezConfiguration des paramètres de compilation pour une application Amplify.

L'exemple suivant montre comment personnaliser les paramètres de compilation pour définir la version par défaut de Node.js sur Node.js 18 et effectuer une mise à niveau vers Node.js version 20 sur une branche de test nomméenode-20.

```
frontend:
  phases:
    preBuild:
      commands:
        - nvm use 18
        - if [ "${AWS_BRANCH}" = "node-20" ]; then nvm use 20; fi
```

> **Warning**
>
> Sachez que `preBuild` les commandes s'exécutent après les mises à jour des packages en direct. La version de Node.js spécifiée par la `nvm use` commande remplacera la version de Node.js définie par les mises à jour des packages en direct.

Résolution des problèmes liés à l'image de build d'Amazon Linux 2023

Les informations suivantes peuvent vous aider à résoudre les problèmes liés à l'image de build Amazon Linux 2023 (AL2023).

Rubriques

- Je souhaite exécuter les fonctions Amplify avec le runtime Python
- Je souhaite exécuter des commandes qui nécessitent des privilèges de superutilisateur ou de root

Je souhaite exécuter les fonctions Amplify avec le runtime Python

Amplify Hosting utilise désormais l'image de build Amazon Linux 2023 par défaut lorsque vous déployez une nouvelle application. AL2023 est préinstallé avec les versions 3.8, 3.9, 3.10 et 3.11 de Python.

Pour des raisons de rétrocompatibilité avec l'image Amazon Linux 2, l'image de compilation contient AL2023 des liens symboliques préinstallés vers les anciennes versions de Python.

Par défaut, la version 3.10 de Python est utilisée dans le monde entier. Pour créer vos fonctions à l'aide d'une version spécifique de Python, exécutez les commandes suivantes dans le fichier de spécification de construction de votre application.

```
version: 1
backend:
  phases:
    build:
      commands:
        # use a python version globally
        - pyenv global 3.11
        # verify python version
        - python --version
        # install pipenv
        - pip install --user pipenv
        # add to path
        - export PATH=$PATH:/root/.local/bin
        # verify pipenv version
        - pipenv --version
        - amplifyPush --simple
```

Je souhaite exécuter des commandes qui nécessitent des privilèges de superutilisateur ou de root

Si vous utilisez l'image de build Amazon Linux 2023 et que vous recevez une erreur lors de l'exécution de commandes système nécessitant des privilèges de superutilisateur ou de superutilisateur, vous devez exécuter ces commandes à l'aide de la `sudo` commande Linux. Par exemple, si une erreur s'affiche lors de l'exécution`yum install -y gcc`, utilisez`sudo yum install -y gcc`.

L'image de compilation d'Amazon Linux 2 utilisait l'utilisateur root, mais l' AL2023 image d'Amplify exécute votre code avec un `amplify` utilisateur personnalisé. Amplify accorde à cet utilisateur

les privilèges nécessaires pour exécuter des commandes à l'aide de la commande Linux. `sudo` Il s'agit d'une bonne pratique à utiliser `sudo` pour les commandes qui nécessitent des privilèges de superutilisateur.

Résolution des problèmes de compilation

Si vous rencontrez des problèmes lors de la création ou du développement d'une application Amplify, consultez les rubriques de cette section pour obtenir de l'aide.

Rubriques

Les nouveaux commits dans mon dépôt ne déclenchent pas les builds d'Amplify

Si les nouveaux commits dans votre dépôt Git ne déclenchent pas les builds d'Amplify, vérifiez que votre webhook est toujours présent dans votre dépôt. S'il est présent, vérifiez l'historique des demandes de webhook pour voir s'il y a eu des échecs. Amplify a une limite de charge utile de 256 Ko pour les webhooks entrants. Si vous envoyez un commit à votre dépôt contenant un grand nombre de fichiers modifiés, vous risquez de dépasser cette limite et de ne pas déclencher les builds.

Le nom de mon référentiel n'est pas répertorié dans la console Amplify lors de la création d'une nouvelle application

Lorsque vous créez une nouvelle application dans la console Amplify, vous pouvez choisir parmi les référentiels disponibles de votre organisation sur la page Ajouter un référentiel et une branche. Il est possible que votre référentiel cible ne soit pas affiché dans la liste s'il n'a pas été récemment mis à jour. Cela peut se produire si votre organisation possède un grand nombre de référentiels. Pour résoudre ce problème, envoyez un commit au dépôt, puis actualisez la liste des référentiels dans la console. Cela devrait entraîner l'affichage du référentiel.

Ma compilation échoue avec l'**`Cannot find module aws-exports`**erreur (applications Gen 1 uniquement)

Si votre application ne trouve pas le `aws-exports.js` fichier lors d'une compilation, l'erreur suivante est renvoyée.

```
TS2307: Cannot find module 'aws-exports'
```

L'interface de ligne de commande (CLI) Amplify génère le `aws-exports.js` fichier lors de la construction de votre backend. Pour résoudre cette erreur, vous devez créer un `aws-exports.js` fichier à utiliser dans le build. Ajoutez le code suivant à votre spécification de construction pour créer le fichier :

```
backend:
  phases:
    build:
      commands:
        - "# Execute Amplify CLI with the helper script"
        - amplifyPush --simple
```

Pour un exemple complet des paramètres de spécification de construction d'une application Amplify, voir. Référence de syntaxe YAML pour les spécifications de construction

Je souhaite annuler un délai de construction

Le délai de compilation par défaut est de 30 minutes. Vous pouvez annuler le délai de construction par défaut à l'aide de la variable d'`_BUILD_TIMEOUT`environnement. Le délai de construction minimum est de 5 minutes. Le délai de construction maximal est de 120 minutes.

Pour obtenir des instructions sur la définition d'une variable d'environnement pour une application dans la console Amplify, consultez. Définition de variables d'environnement

Résolution des problèmes liés aux domaines personnalisés

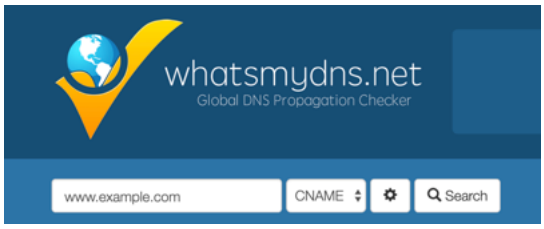
Si vous rencontrez des problèmes lors de la connexion d'un domaine personnalisé à votre application Amplify, consultez les rubriques de cette section pour obtenir de l'aide.

Si vous ne trouvez pas de solution à votre problème ici, contactez Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .

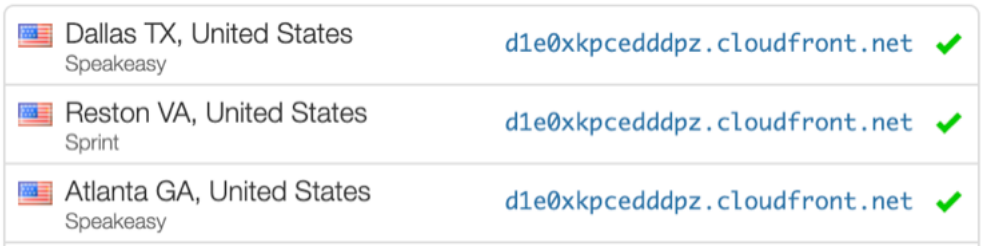
Rubriques

Je dois vérifier que mon CNAME est résolu

1. Après avoir mis à jour vos enregistrements DNS auprès de votre fournisseur de domaine tiers, vous pouvez utiliser un outil tel que dig ou un site Web gratuit tel que https://www.whatsmydns.net/ pour vérifier que votre enregistrement CNAME est correctement résolu. La capture d'écran suivante montre comment utiliser whatsmydns.net pour vérifier votre enregistrement CNAME pour le domaine www.example.com.



2. Choisissez Rechercher, et whatsmydns.net affiche les résultats de votre CNAME. La capture d'écran suivante est un exemple de liste de résultats qui vérifient que le CNAME correspond correctement à une URL cloudfront.net.



Mon domaine hébergé chez un tiers est bloqué dans l'état En attente de vérification

1. Si votre domaine personnalisé est bloqué dans l'état En attente de vérification, vérifiez que vos CNAME enregistrements sont en cours de résolution. Consultez la rubrique de résolution des problèmes précédente, Comment puis-je vérifier que mes problèmes sont résolusCNAME, pour obtenir des instructions sur l'exécution de cette tâche.
2. Si vos CNAME enregistrements ne sont pas résolus, vérifiez que l'CNAMEentrée existe dans vos paramètres DNS auprès de votre fournisseur de domaine.

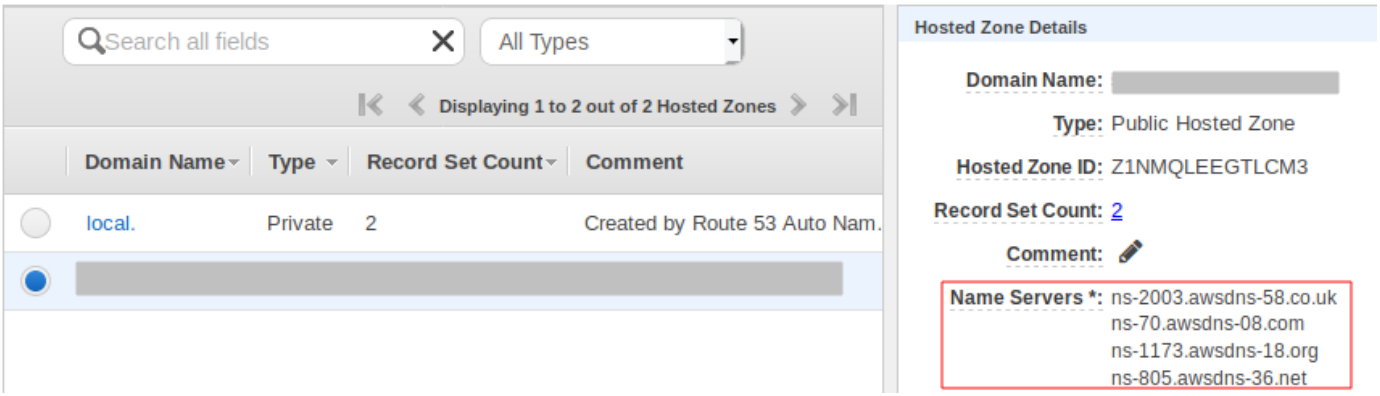
   > **Important**
   >
   > Il est important de mettre à jour vos CNAME enregistrements dès que vous créez votre domaine personnalisé. Une fois votre application créée dans la console Amplify, votre CNAME dossier est vérifié toutes les quelques minutes pour déterminer s'il est résolu. Si le problème persiste au bout d'une heure, la vérification est effectuée toutes les quelques heures, ce qui peut retarder la mise en service de votre domaine. Si vous avez ajouté ou mis à jour vos CNAME enregistrements quelques heures après avoir créé votre application, il est fort probable que votre application reste bloquée dans l'état En attente de vérification.

3. Si vous avez vérifié que l'CNAMEenregistrement existe, il se peut qu'il y ait un problème avec votre fournisseur DNS. Vous pouvez soit contacter le fournisseur DNS pour savoir pourquoi la vérification DNS ne résout pas le problème, CNAME soit migrer votre DNS vers Route 53. Pour plus d'informations, consultez Faire d'Amazon Route 53 le service DNS d'un domaine existant.

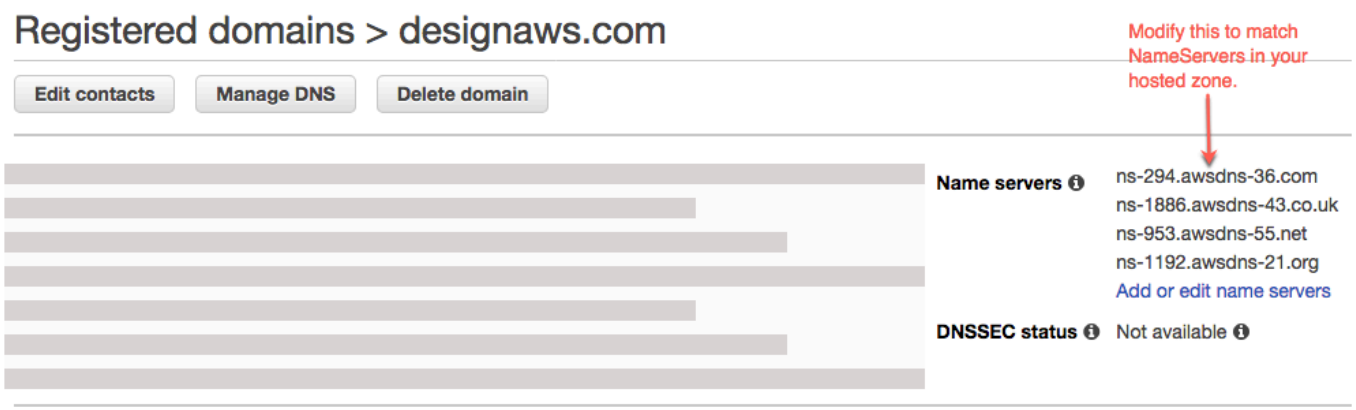
Mon domaine hébergé par Amazon Route 53 est bloqué dans l'état En attente de vérification

Si vous avez transféré votre domaine vers Amazon Route 53, il est possible que votre domaine possède des serveurs de noms différents de ceux émis par Amplify lors de la création de votre application. Procédez comme suit pour diagnostiquer la cause de l'erreur.

1. Connectez-vous à la console Amazon Route 53
2. Dans le volet de navigation, choisissez Hosted Zones, puis choisissez le nom du domaine que vous connectez.
3. Enregistrez les valeurs du serveur de noms dans la section Détails de la zone hébergée. Vous avez besoin de ces valeurs pour passer à l'étape suivante. La capture d'écran suivante de la console Route 53 montre l'emplacement des valeurs du serveur de noms dans le coin inférieur droit.



4. Dans le panneau de navigation, choisissez Registered domains (Domaines membres). Vérifiez que les serveurs de noms affichés dans la section Domaines enregistrés correspondent aux valeurs des serveurs de noms que vous avez enregistrées à l'étape précédente dans la section Détails de la zone hébergée. S'ils ne correspondent pas, modifiez les valeurs du serveur de noms pour qu'elles correspondent aux valeurs de votre zone hébergée. La capture d'écran suivante de la console Route 53 montre l'emplacement des valeurs du serveur de noms sur le côté droit.



5. Si cela ne résout pas le problème, contactez Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .

Mon application avec des sous-domaines à plusieurs niveaux est bloquée dans l'état En attente de vérification

Si une application comportant des sous-domaines à plusieurs niveaux est bloquée dans l'état En attente de vérification lors de la connexion à un fournisseur DNS tiers, il se peut que le format de vos enregistrements DNS présente un problème. Certains fournisseurs DNS ajoutent automatiquement les suffixes de domaine de deuxième niveau (SLD) et de domaine de premier niveau (TLD) à vos enregistrements. Si vous spécifiez également le domaine dans un format qui inclut le SLD et le TLD, cela peut entraîner un problème de vérification du domaine.

Lorsque vous connectez un domaine, essayez d'abord de spécifier le nom de domaine en utilisant le format complet fourni par Amplify, par exemple. `_hash.docs.backend.example.com` Si la configuration SSL reste bloquée dans l'état En attente de vérification, essayez de supprimer le TLD et le SLD des enregistrements. Par exemple, si le format complet est`_hash.docs.backend.example.com`, spécifiez`_hash.docs.backend`. Attendez 15 à 30 minutes pour permettre aux enregistrements de se propager. Utilisez ensuite un outil tel que MX Toolbox pour vérifier si le processus de vérification fonctionne.

Mon fournisseur DNS ne prend pas en charge les enregistrements A avec des noms de domaine complets

Certains fournisseurs DNS ne prennent pas en charge les enregistrements A dotés d'un nom de domaine complet (FQDN), tels que`example.cloudfront.net`. Par exemple, Cloudflare ne A

records peut écrire que des IPv4 adresses et ne les prend pas en chargeFQDNs. Pour contourner cette limitation, nous vous recommandons d'utiliser CNAME des enregistrements plutôt que A records dans votre DNS configuration.

À titre d'exemple, la DNS configuration suivante utilise unA record.

```
A     | @ | ***.cloudfront.net
CNAME | www | ***.cloudfront.net
```

Modifiez-le selon la DNS configuration suivante pour utiliser uniquement CNAME les enregistrements.

```
CNAME | @ | ***.cloudfront.net
CNAME | www | ***.cloudfront.net
```

Cette solution vous permet de pointer correctement votre domaine apex (@ record) vers des services tels que CloudFront, tout en évitant la IPv4 seule limitation du système A records Cloudflare.

Je reçois un CNAMEAlready ExistsException message d'erreur

Si une CNAMEAlreadyExistsExceptionerreur s'affiche, cela signifie que l'un des noms d'hôte que vous avez essayé de connecter (un sous-domaine ou le domaine apex) est déjà déployé sur une autre CloudFront distribution Amazon. La source de votre erreur dépend de vos fournisseurs d'hébergement et de DNS actuels.

Un CNAME alias, tel que `example.com` ou ne `sub.example.com` peut être associé qu'à une seule CloudFront distribution à la fois. Cela CNAMEAlreadyExistsExceptionindique que votre domaine est déjà associé à une autre CloudFront distribution, soit au sein du même compte Compte AWS, soit potentiellement dans un autre compte. Le domaine doit être dissocié de la CloudFront distribution précédente pour que la nouvelle distribution créée par Amplify Hosting fonctionne. Vous devrez peut-être vérifier plusieurs comptes si vous ou votre organisation en possédez plusieurs Compte AWS.

Procédez comme suit pour diagnostiquer la cause de l'CNAMEAlreadyExistsExceptionerreur.

1. Connectez-vous à la CloudFront console Amazon et vérifiez que ce domaine n'est pas déployé sur une autre distribution. Un seul CNAME enregistrement peut être joint à une CloudFront distribution à la fois.
2. Si vous avez déjà déployé le domaine sur une CloudFront distribution, vous devez le supprimer.

a. Choisissez Distributions dans le menu de navigation de gauche.

b. Sélectionnez le nom de la distribution à modifier.

c. Choisissez l'onglet Général. Dans la section Settings (Paramètres), choisissez Edit (Modifier).

d. Supprimez le nom de domaine du nom de domaine alternatif (CNAME). Choisissez ensuite Enregistrer les modifications.

3. Vérifiez qu'il n'existe aucune autre CloudFront distribution utilisant ce domaine dans le domaine actuel Compte AWS ou dans un autre Comptes AWS. Si cela ne perturbe aucun service en cours d'exécution, essayez de supprimer et de recréer la zone hébergée.
4. Vérifiez si ce domaine est connecté à une autre application Amplify que vous possédez. Si tel est le cas, assurez-vous que vous n'essayez pas de réutiliser l'un des noms d'hôte. Si vous utilisez `www.example.com` pour une autre application, vous ne pouvez pas l'utiliser `www.example.com` avec l'application à laquelle vous êtes actuellement connecté. Vous pouvez utiliser d'autres sous-domaines, tels `blog.example.com` que.
5. Si ce domaine a été connecté avec succès à une autre application puis supprimé au cours de la dernière heure, réessayez au bout d'une heure au moins. Si cette exception persiste après 6 heures, contactez Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .
6. Si vous gérez votre domaine via Route 53, veillez à nettoyer toute zone hébergée CNAME ou tout ALIAS enregistrement pointant vers l'ancienne CloudFront distribution.
7. Après avoir effectué les étapes précédentes, supprimez le domaine personnalisé d'Amplify Hosting et recommencez le flux de travail pour connecter un domaine personnalisé dans la console Amplify.

Je reçois un message d'erreur « Vérification supplémentaire requise »

Si le message d'erreur « Vérification supplémentaire requise » s'affiche, cela signifie que AWS Certificate Manager (ACM) a besoin d'informations supplémentaires pour traiter cette demande de certificat. Cela peut se produire en tant que mesure de protection contre la fraude, par exemple lorsque le domaine se classe dans les 1 000 meilleurs sites Web d'Alexa. Pour fournir ces informations, utilisez le Centre de support pour contacter Support. Si vous n'avez pas de plan de support, publiez un nouveau fil de discussion dans le forum de discussion ACM.

> **Note**
>
> Vous ne pouvez pas demander de certificat pour des noms de domaine qui sont la propriété d'Amazon, par exemple ceux qui se terminent par amazonaws.com, cloudfront.net ou elasticbeanstalk.com.

Je reçois une erreur 404 sur l' CloudFront URL

Pour gérer le trafic, Amplify Hosting pointe vers une CloudFront URL via un enregistrement CNAME. Lors du processus de connexion d'une application à un domaine personnalisé, la console Amplify affiche l' CloudFront URL de l'application. Toutefois, vous ne pouvez pas accéder directement à votre application à l'aide de cette CloudFront URL. Elle renvoie une erreur 404. Votre application est résolue uniquement à l'aide de l'URL de l'application Amplify (par exemple) ou de votre domaine personnalisé (par exemple`www.example.com`). `https://main.d5udybEXAMPLE.amplifyapp.com`

Amplify doit acheminer les demandes vers la branche déployée appropriée et utilise le nom d'hôte pour ce faire. Par exemple, vous pouvez configurer le domaine `www.example.com` qui pointe vers la branche principale d'une application, mais également configurer `dev.example.com` qui pointe vers la branche de développement de la même application. Par conséquent, vous devez visiter votre application en fonction de ses sous-domaines configurés afin qu'Amplify puisse acheminer les demandes en conséquence.

Je reçois des erreurs de certificat SSL ou HTTPS lorsque je visite mon domaine

Si des enregistrements DNS d'autorisation d'autorité de certification (CAA) sont configurés auprès de votre fournisseur DNS tiers, AWS Certificate Manager (ACM) risque de ne pas être en mesure de mettre à jour ou de réémettre les certificats intermédiaires pour votre certificat SSL de domaine personnalisé. Pour résoudre ce problème, vous devez ajouter un enregistrement CAA pour approuver au moins un des domaines d'autorité de certification d'Amazon. La procédure suivante décrit les étapes que vous devez effectuer.

Pour ajouter un enregistrement CAA afin de faire confiance à une autorité de certification Amazon

1. Configurez un enregistrement CAA auprès de votre fournisseur de domaine pour faire confiance à au moins un des domaines d'autorité de certification d'Amazon. Pour plus d'informations sur la

configuration de l'enregistrement CAA, consultez la section Problèmes d'autorisation de l'autorité de certification (CAA) dans le guide de AWS Certificate Manager l'utilisateur.

2. Utilisez l'une des méthodes suivantes pour mettre à jour votre certificat SSL :

   - Effectuez une mise à jour manuelle à l'aide de la console Amplify.

     > **Note**
     >
     > Cette méthode entraînera une interruption de service de votre domaine personnalisé.

     a. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.

     b. Choisissez l'application à laquelle vous souhaitez ajouter un enregistrement CAA.

     c. Dans le volet de navigation, choisissez Paramètres de l'application, Gestion des domaines.

     d. Sur la page Gestion du domaine, supprimez le domaine personnalisé.

     e. Connectez à nouveau votre application au domaine personnalisé. Ce processus émet un nouveau certificat SSL et ses certificats intermédiaires peuvent désormais être gérés par ACM.

        Pour reconnecter votre application à votre domaine personnalisé, appliquez l'une des procédures suivantes correspondant au fournisseur de domaine que vous utilisez.

        - Ajouter un domaine personnalisé géré par Amazon Route 53.
        - Ajouter un domaine personnalisé géré par un fournisseur DNS tiers.
        - Mise à jour des enregistrements DNS pour un domaine géré par GoDaddy.

   - Contactez-nous Support pour que votre certificat SSL soit réémis.

Les composants de chemin ne sont pas pris en charge dans les redirections de domaine

Les redirections de domaine correspondent uniquement à la partie du nom d'hôte. Les composants de chemin dans les règles source basées sur le domaine (par exemple,`"https://domain.com/path"`) ne sont pas pris en charge et entraîneront l'ignorance de la règle sans erreur. Pour de plus amples informations, veuillez consulter Exemple de référence pour les redirections et les réécritures.

Je reçois une erreur 400 pour une association de domaines entre comptes

Lorsque vous DomainAssociation lancez une demande pour une application Amplify avec un domaine qui est déjà ou a été précédemment associé à différentes applications Amplify dans d'autres comptes AWS de la même région, cela est considéré comme une association de domaines entre comptes. Si cette erreur s'affiche, cela signifie que vous essayez d'associer des domaines entre comptes, ce qui nécessite une vérification manuelle. Si vous souhaitez créer une association de domaines entre comptes, contactez le support AWS pour obtenir de l'aide.

Résolution des problèmes liés aux applications rendues côté serveur

Si vous rencontrez des problèmes inattendus lors du déploiement d'une application SSR avec le système de calcul Amplify Hosting, consultez les rubriques de résolution des problèmes suivantes. Si vous ne trouvez pas de solution à votre problème ici, consultez le guide de résolution des problèmes de calcul Web SSR dans le référentiel Amplify GitHub Hosting Issues.

Rubriques

- J'ai besoin d'aide pour utiliser un adaptateur de framework
- Les routes de l'API Edge font échouer ma compilation de Next.js
- La régénération statique incrémentielle à la demande ne fonctionne pas pour mon application
- La sortie de compilation de mon application dépasse la taille maximale autorisée
- Ma compilation échoue en raison d'une erreur de mémoire insuffisante
- La taille de la réponse HTTP de mon application est trop grande
- Comment puis-je mesurer le temps de démarrage de mon application informatique localement ?
- Ma compilation échoue avec une erreur de version obsolète de Node.js

J'ai besoin d'aide pour utiliser un adaptateur de framework

Si vous rencontrez des problèmes pour déployer une application SSR qui utilise un adaptateur de framework, consultezUtilisation d'adaptateurs open source pour n'importe quel framework SSR.

Les routes de l'API Edge font échouer ma compilation de Next.js

Actuellement, Amplify ne prend pas en charge les routes d'API Next.js Edge. Vous devez utiliser des logiciels non périphériques APIs et des intergiciels lorsque vous hébergez votre application avec Amplify.

La régénération statique incrémentielle à la demande ne fonctionne pas pour mon application

À partir de la version 12.2.0, Next.js prend en charge la régénération statique incrémentielle (ISR) pour purger manuellement le cache Next.js pour une page spécifique. Cependant, Amplify ne prend actuellement pas en charge l'ISR à la demande. Si votre application utilise la revalidation à la demande de Next.js, cette fonctionnalité ne fonctionnera pas lorsque vous déployez votre application sur Amplify.

La sortie de compilation de mon application dépasse la taille maximale autorisée

Actuellement, la taille de sortie maximale prise en charge par Amplify pour les applications SSR est de 220 Mo. Si vous recevez un message d'erreur indiquant que la taille de la sortie de build de votre application dépasse la taille maximale autorisée, vous devez prendre des mesures pour la réduire.

Pour réduire la taille de la sortie de build d'une application, vous pouvez inspecter les artefacts de build de l'application et identifier les dépendances importantes à mettre à jour ou à supprimer. Tout d'abord, téléchargez les artefacts de construction sur votre ordinateur local. Vérifiez ensuite la taille des répertoires. Par exemple, le `node_modules` répertoire peut contenir des fichiers binaires tels que `@swc` et `@esbuild` qui sont référencés par les fichiers d'exécution du serveur Next.js. Comme ces fichiers binaires ne sont pas nécessaires au moment de l'exécution, vous pouvez les supprimer après la compilation.

Utilisez les instructions suivantes pour télécharger le résultat de compilation d'une application et inspecter la taille des répertoires à l'aide de la AWS Command Line Interface (CLI).

Pour télécharger et inspecter le résultat de compilation d'une application Next.js

1. Ouvrez une fenêtre de terminal et exécutez la commande suivante. Modifiez l'identifiant de l'application, le nom de la branche et l'identifiant de la tâche selon vos propres informations. Pour l'identifiant de la tâche, utilisez le numéro de version de la version échouée que vous étudiez.

```
aws amplify get-job --app-id abcd1234 --branch-name main --job-id 2
```

2. Dans la sortie du terminal, recherchez l'URL des artefacts présignés dans la `stepName: "BUILD"` section `jobsteps`,. L'URL est surlignée en rouge dans l'exemple de sortie suivant.

```
"job": {
    "summary": {
        "jobArn": "arn:aws:amplify:us-west-2:111122223333:apps/abcd1234/main/
jobs/0000000002",
        "jobId": "2",
        "commitId": "HEAD",
        "commitTime": "2024-02-08T21:54:42.398000+00:00",
        "startTime": "2024-02-08T21:54:42.674000+00:00",
        "status": "SUCCEED",
        "endTime": "2024-02-08T22:03:58.071000+00:00"
    },
    "steps": [
        {
            "stepName": "BUILD",
            "startTime": "2024-02-08T21:54:42.693000+00:00",
            "status": "SUCCEED",
            "endTime": "2024-02-08T22:03:30.897000+00:00",
            "logUrl": "https://aws-amplify-prod-us-west-2-artifacts.s3.us-
west-2.amazonaws.com/abcd1234/main/0000000002/BUILD/log.txt?X-Amz-Security-
Token=IQoJb3JpZ2luX2V...Example
```

3. Copiez et collez l'URL dans une fenêtre de navigateur. Un `artifacts.zip` fichier est téléchargé sur votre ordinateur local. Il s'agit du résultat de votre build.
4. Exécutez la commande d'utilisation du du disque pour vérifier la taille des répertoires. L'exemple de commande suivant renvoie la taille des `static` répertoires `compute` et.

```
du -csh compute static
```

Voici un exemple de sortie avec des informations de taille pour les `static` répertoires `compute` et.

```
 29M    compute
3.8M    static
 33M    total
```

5. Ouvrez le `compute` répertoire et `node_modules` localisez-le. Vérifiez vos dépendances pour les fichiers que vous pouvez mettre à jour ou supprimer afin de réduire la taille du dossier.
6. Si votre application inclut des fichiers binaires qui ne sont pas nécessaires à l'exécution, supprimez-les après la compilation en ajoutant les commandes suivantes à la section build du `amplify.yml` fichier de votre application.

```
- rm -f node_modules/@swc/core-linux-x64-gnu/swc.linux-x64-gnu.node
- rm -f node_modules/@swc/core-linux-x64-musl/swc.linux-x64-musl.node
```

Voici un exemple de la section des commandes de génération d'un `amplify.yml` fichier dans laquelle ces commandes ont été ajoutées après l'exécution d'une version de production.

```
frontend:
  phases:
    build:
      commands:
         -npm run build

         // After running a production build, delete the files
         - rm -f node_modules/@swc/core-linux-x64-gnu/swc.linux-x64-gnu.node
         - rm -f node_modules/@swc/core-linux-x64-musl/swc.linux-x64-musl.node
```

Ma compilation échoue en raison d'une erreur de mémoire insuffisante

Next.js vous permet de mettre en cache des artefacts de build afin d'améliorer les performances lors des builds suivants. En outre, le AWS CodeBuild conteneur d'Amplify compresse et télécharge ce cache sur Amazon S3, en votre nom, afin d'améliorer les performances de compilation ultérieures. Cela pourrait entraîner l'échec de votre compilation en raison d'une erreur de mémoire insuffisante.

Effectuez les actions suivantes pour empêcher votre application de dépasser la limite de mémoire pendant la phase de création. Tout d'abord, `.next/cache/**/*` supprimez-le de la section cache.paths de vos paramètres de compilation. Supprimez ensuite la variable d'`NODE_OPTIONS`environnement de votre fichier de paramètres de compilation. Définissez plutôt la variable d'`NODE_OPTIONS`environnement dans la console Amplify pour définir la limite de mémoire maximale du nœud. Pour plus d'informations sur la définition des variables d'environnement à l'aide de la console Amplify, consultez. Définition de variables d'environnement

Après avoir apporté ces modifications, réessayez de créer. En cas de succès, ajoutez-le à `.next/cache/**/*` nouveau à la section cache.paths de votre fichier de paramètres de compilation.

Pour plus d'informations sur la configuration du cache Next.js afin d'améliorer les performances de compilation, consultez AWS CodeBuild sur le site Web Next.js.

La taille de la réponse HTTP de mon application est trop grande

Actuellement, la taille de réponse maximale prise en charge par Amplify pour les applications Next.js 12 et ultérieures utilisant la plate-forme Web Compute est de 5,72 Mo. Les réponses dépassant cette limite renvoient 504 erreurs sans aucun contenu aux clients.

Comment puis-je mesurer le temps de démarrage de mon application informatique localement ?

Suivez les instructions ci-dessous pour déterminer le temps de initialization/start disponibilité local de votre application Compute Next.js 12 ou version ultérieure. Vous pouvez comparer les performances de votre application localement à celles d'Amplify Hosting et utiliser les résultats pour améliorer les performances de votre application.

Pour mesurer le temps d'initialisation d'une application Next.js Compute localement

1. Ouvrez le `next.config.js` fichier de l'application et définissez l'`output`option `standalone` comme suit.

```
** @type {import('next').NextConfig} */
const nextConfig = {
  // Other options
  output: "standalone",
};

module.exports = nextConfig;
```

2. Ouvrez une fenêtre de terminal et exécutez la commande suivante pour créer l'application.

```
next build
```

3. Exécutez la commande suivante pour copier le `.next/static` dossier dans`.next/standalone/.next/static`.

```
cp -r .next/static .next/standalone/.next/static
```

4. Exécutez la commande suivante pour copier le `public` dossier dans`.next/standalone/public`.

```
cp -r public .next/standalone/public
```

5. Exécutez la commande suivante pour démarrer le serveur Next.js.

```
node .next/standalone/server.js
```

6. Notez le temps qui s'écoule entre l'exécution de la commande à l'étape 5 et le démarrage du serveur. Lorsque le serveur écoute sur un port, il doit imprimer le message suivant.

```
Listening on port 3000
```

7. Notez le temps nécessaire au chargement des autres modules après le démarrage du serveur à l'étape 6. Par exemple, le chargement de bibliothèques `bugsnag` prend 10 à 12 secondes. Une fois chargé, le message de confirmation s'affichera`[bugsnag] loaded`.
8. Additionnez les durées des étapes 6 et 7 ensemble. Ce résultat correspond au temps de disponibilité local initialization/start de votre application Compute.

Ma compilation échoue avec une erreur de version obsolète de Node.js

Problème : La compilation de votre application SSR échoue avec une erreur indiquant que la version de Node.js n'est pas prise en charge.

```
# NODE.JS VERSION NOT SUPPORTED
================================================================================
Your application uses Node.js v18.x.x, which is no longer supported.
AWS Amplify Console has ended support for Node.js 14, Node.js 16 and Node.js 18.

To deploy your application, please upgrade to Node.js 20 or later.

For detailed migration guidelines, visit: https://docs.aws.amazon.com/amplify/latest/
userguide/troubleshooting-general.html#update-node-version
================================================================================
```

Cause : Votre application SSR a été créée à l'aide d'une version obsolète de Node.js (14.x, 16.x ou 18.x). À compter du 15 septembre 2025, Amplify bloque le déploiement des applications SSR qui utilisent ces versions obsolètes pendant le processus de génération.

Mettez à jour votre environnement de génération pour utiliser Node.js 20 ou version ultérieure. Pour obtenir des instructions complètes, consultez Je dois mettre à jour la version Node.js de mon application.

Résolution des problèmes de redirections et de réécritures

Si vous rencontrez des problèmes lors de la configuration des redirections et des réécritures pour une application Amplify, consultez les rubriques de cette section pour obtenir de l'aide.

Rubriques

- L'accès est refusé pour certains itinéraires, même avec la règle de redirection SPA.
- Je souhaite configurer un proxy inverse pour une API

L'accès est refusé pour certains itinéraires, même avec la règle de redirection SPA.

Si vous recevez un message d'erreur de refus d'accès pour certains itinéraires dotés d'une règle de redirection SPA, il `baseDirectory` se peut que celle-ci ne soit pas définie correctement dans les paramètres de compilation de l'application. Par exemple, si le frontend de votre application est intégré au `build` répertoire, vos paramètres de génération doivent également pointer vers le `build` répertoire. L'exemple de spécification de construction suivant illustre ce paramètre.

```
frontend:
  artifacts:
    baseDirectory: build
    files:
        - "**/*"
```

Pour un exemple complet des paramètres de spécification de construction d'une application Amplify, voir Référence de syntaxe YAML pour les spécifications de construction

Je souhaite configurer un proxy inverse pour une API

Vous pouvez utiliser le code JSON suivant pour configurer un proxy inverse pour un point de terminaison dynamique.

```
[
  {
    "source": "/documents/<*>",
    "target": "https://otherdomain/resource/<*>",
    "status": "200",
    "condition": null
  }
]
```

Pour un exemple de base de création d'un proxy inverse pour votre application Amplify vers une API tierce, voir. Réécriture du proxy inversé

Résolution des problèmes de mise en cache

Si vous rencontrez des problèmes de mise en cache pour une application Amplify, consultez les rubriques de cette section pour obtenir de l'aide.

Rubriques

- Je souhaite réduire la taille du cache d'une application
- Je souhaite désactiver la lecture depuis le cache d'une application

Je souhaite réduire la taille du cache d'une application

Si vous utilisez le cache, vous mettez peut-être en cache des fichiers intermédiaires qui ne sont pas nettoyés entre les versions. La mise en cache de ces fichiers peu utilisés augmentera la taille de votre cache. Pour éviter cela, vous pouvez exclure des dossiers spécifiques de la mise en cache à l'aide de la `!` directive figurant dans la `cache` section des spécifications de compilation de votre application.

L'exemple de paramètres de construction suivant montre comment utiliser la `!` directive pour spécifier un dossier que vous ne souhaitez pas mettre en cache.

```
cache:
```

```
  paths:
    - node_modules/**/*
    - "!node_modules/path/not/to/cache"
```

Lorsque vous mettez en cache le `node_modules` dossier, `node_modules/.cache` il est omis par défaut.

Pour un exemple complet des paramètres de spécification de construction d'une application Amplify, voir Référence de syntaxe YAML pour les spécifications de construction

Je souhaite désactiver la lecture depuis le cache d'une application

Si vous souhaitez désactiver la lecture depuis le cache d'une application, supprimez la section du cache de la spécification de construction de votre application.

Configuration de l'accès d'Amplify aux référentiels GitHub

Amplify utilise désormais la fonctionnalité GitHub Apps pour autoriser Amplify à accéder en lecture seule aux référentiels. GitHub Avec l' GitHub application Amplify, les autorisations sont mieux affinées, ce qui vous permet d'accorder à Amplify l'accès uniquement aux référentiels que vous spécifiez. Pour en savoir plus sur les GitHub applications, consultez la section À propos GitHub des applications sur le GitHub site Web.

Lorsque vous connectez une nouvelle application stockée dans un GitHub dépôt, Amplify utilise par défaut GitHub l'application pour accéder au dépôt. Cependant, les applications Amplify existantes que vous avez précédemment connectées depuis des GitHub dépôts sont utilisées OAuth pour y accéder. CI/CD continuera de fonctionner pour ces applications, mais nous vous recommandons vivement de les migrer pour utiliser la nouvelle application Amplify GitHub .

Lorsque vous déployez une nouvelle application ou que vous migrez une application existante à l'aide de la console Amplify, vous êtes automatiquement dirigé vers l'emplacement d'installation de l'application Amplify. GitHub Pour accéder manuellement à la page d'accueil d'installation de l'application, ouvrez un navigateur Web et accédez à l'application par région. Utilisez le format en `https://github.com/apps/aws-amplify-`*`REGION`* le *`REGION`* remplaçant par la région dans laquelle vous allez déployer votre application Amplify. Par exemple, pour installer l' GitHub application Amplify dans la région de l'ouest des États-Unis (Oregon), accédez à https://github.com/apps/ aws-amplify-us-west -2.

Rubriques

Installation et autorisation de l'application GitHub Amplify pour un nouveau déploiement

Lorsque vous déployez une nouvelle application sur Amplify à partir du code existant dans un GitHub dépôt, suivez les instructions suivantes pour installer et autoriser l'application. GitHub

Pour installer et autoriser l'application Amplify GitHub

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web.
3. Sur la page Commencer avec Amplify Hosting, choisissez GitHub, puis choisissez Continuer.
4. Si c'est la première fois que vous connectez un GitHub dépôt, une nouvelle page s'ouvre dans votre navigateur sur GitHub .com, demandant l'autorisation d'autoriser l' AWS Amplify accès à votre GitHub compte. Choisissez Authorize (Autoriser).
5. Ensuite, vous devez installer l' GitHub application Amplify sur votre GitHub compte. Une page s'ouvre sur GitHub.com demandant l'autorisation d'installation et d'autorisation AWS Amplify sur votre GitHub compte.
6. Sélectionnez le GitHub compte sur lequel vous souhaitez installer l'application Amplify GitHub .
7. Effectuez l'une des actions suivantes :
   - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
   - Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le dépôt de l'application que vous migrez dans les dépôts que vous sélectionnez.
8. Choisissez Installer et autoriser.
9. Vous êtes redirigé vers la page Ajouter une branche de référentiel pour votre application dans la console Amplify.
10. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.

11. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.
12. Choisissez Suivant.
13. Sur la page Configurer les paramètres de build, choisissez Next.
14. Sur la page Révision, choisissez Enregistrer et déployer.

Migration d'une OAuth application existante vers l'application Amplify GitHub

Les applications Amplify existantes que vous avez précédemment connectées à partir de GitHub référentiels utilisent OAuth pour accéder aux référentiels. Nous vous recommandons vivement de migrer ces applications pour utiliser l'application Amplify GitHub.

Suivez les instructions ci-dessous pour migrer une application et supprimer le OAuth webhook correspondant dans votre GitHub compte. Notez que la procédure de migration varie selon que l'application GitHub Amplify est déjà installée ou non. Après avoir migré votre première application, installé et autorisé l' GitHub application, il vous suffit de mettre à jour les autorisations du référentiel pour les migrations d'applications suivantes.

Pour migrer une application depuis l'application OAuth vers l' GitHub application

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application que vous souhaitez migrer.
3. Sur la page d'informations de l'application, repérez le message bleu Migrer vers notre GitHub application et choisissez Démarrer la migration.
4. Sur la page Installer et autoriser GitHub l'application, choisissez Configurer GitHub l'application.
5. Une nouvelle page s'ouvre dans votre navigateur sur GitHub .com, demandant l'autorisation d'autoriser l' AWS Amplify accès à votre GitHub compte. Choisissez Authorize (Autoriser).
6. Sélectionnez le GitHub compte sur lequel vous souhaitez installer l'application Amplify GitHub .
7. Effectuez l'une des actions suivantes :
   - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
   - Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le dépôt de l'application que vous migrez dans les référentiels que vous sélectionnez.
8. Choisissez Installer et autoriser.

9. Vous êtes redirigé vers la page Installer et autoriser GitHub l'application pour votre application dans la console Amplify. Si GitHub l'autorisation est réussie, vous verrez un message de réussite. Choisissez Next.
10. Sur la page Installation complète, choisissez Terminer l'installation. Cette étape supprime votre webhook existant, en crée un nouveau et termine la migration.

Configuration de l' GitHub application Amplify pour les déploiements, CloudFormation CLI et SDK

Les applications Amplify existantes que vous avez précédemment connectées à partir de GitHub référentiels utilisent OAuth pour accéder aux référentiels. Cela peut inclure les applications que vous avez déployées à l'aide de l'interface de ligne de commande (CLI) Amplify CloudFormation, ou du. SDKs Nous vous recommandons vivement de migrer ces applications pour utiliser la nouvelle application Amplify GitHub . La migration doit être effectuée dans la console Amplify dans le. AWS Management Console Pour obtenir des instructions, veuillez consulter Migration d'une OAuth application existante vers l'application Amplify GitHub.

Vous pouvez utiliser CloudFormation la CLI Amplify et le SDKs pour déployer une nouvelle application Amplify qui utilise l'application pour accéder au dépôt GitHub . Ce processus nécessite que vous installiez d'abord l' GitHub application Amplify sur votre GitHub compte. Ensuite, vous devrez générer un jeton d'accès personnel dans votre GitHub compte. Enfin, déployez l'application et spécifiez le jeton d'accès personnel.

Installez l' GitHub application Amplify sur votre compte

1. Ouvrez un navigateur Web et accédez à l'emplacement d'installation de l' GitHubapplication Amplify dans la AWS région où vous allez déployer votre application.

   Utilisez le format `https://github.com/apps/aws-amplify-REGION/installations/new` en le *`REGION`* remplaçant par votre propre entrée. Par exemple, si vous installez votre application dans la région de l'ouest des États-Unis (Oregon), spécifiez`https://github.com/apps/aws-amplify-us-west-2/installations/new`.
2. Sélectionnez le GitHub compte sur lequel vous souhaitez installer l'application Amplify GitHub .
3. Effectuez l'une des actions suivantes :
   - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.

- Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le dépôt de l'application que vous migrez dans les dépôts que vous sélectionnez.

4. Choisissez Installer.

Générez un jeton d'accès personnel dans votre GitHub compte

1. Connectez-vous à votre GitHub compte.
2. Dans le coin supérieur droit, localisez votre photo de profil et choisissez Paramètres dans le menu.
3. Dans le menu de navigation de gauche, choisissez Paramètres du développeur.
4. Sur la page GitHub Applications, dans le menu de navigation de gauche, choisissez Jetons d'accès personnels.
5. Sur la page des jetons d'accès personnels, choisissez Générer un nouveau jeton.
6. Sur la page Nouveau jeton d'accès personnel, dans Note, entrez un nom descriptif pour le jeton.
7. Dans la section Select scopes, sélectionnez admin:repo_hook.
8. Choisissez Generate token (Générer le jeton).
9. Copiez et enregistrez le jeton d'accès personnel. Vous devrez le fournir lorsque vous déployez une application Amplify avec la CLI CloudFormation, ou le. SDKs

Une fois que l' GitHub application Amplify est installée sur votre GitHub compte et que vous avez généré un jeton d'accès personnel, vous pouvez déployer une nouvelle application avec la CLI Amplify, CloudFormation ou le. SDKs Utilisez le `accessToken` champ pour spécifier le jeton d'accès personnel que vous avez créé lors de la procédure précédente. Pour plus d'informations, consultez CreateAppla référence de l'API Amplify et le guide AWS::Amplify::Appde l'AWS CloudFormation utilisateur.

La commande CLI suivante déploie une nouvelle application Amplify qui utilise l'application pour GitHub accéder au référentiel. Remplacez *myapp-using-githubapphttps://github.com/Myaccount/react-app*, et *MY_TOKEN* par vos propres informations.

```
aws amplify create-app --name myapp-using-githubapp --repository https://github.com/
Myaccount/react-app --access-token MY_TOKEN
```

Configuration des aperçus Web avec l'application Amplify GitHub

Un aperçu Web déploie chaque pull request (PR) envoyée à votre GitHub référentiel vers une URL de prévisualisation unique. Les aperçus utilisent désormais l'application GitHub Amplify pour accéder à GitHub votre dépôt. Pour obtenir des instructions sur l'installation et l'autorisation de l' GitHub application pour les aperçus Web, consultez. Activer les aperçus Web pour les pull requests

AWS Amplify Référence d'hébergement

Utilisez les rubriques de cette section pour trouver des documents de référence détaillés pour AWS Amplify.

Rubriques

AWS CloudFormation soutien

Utilisez des AWS CloudFormation modèles pour provisionner les ressources Amplify, permettant ainsi des déploiements d'applications Web répétables et fiables. AWS CloudFormation fournit un langage commun qui vous permet de décrire et de provisionner toutes les ressources d'infrastructure de votre environnement cloud et simplifie le déploiement sur plusieurs and/or régions de AWS comptes en quelques clics.

Pour Amplify Hosting, consultez la documentation Amplify. CloudFormation Pour Amplify Studio, consultez la documentation Amplify UI Builder. CloudFormation

AWS Command Line Interface soutien

Utilisez le AWS Command Line Interface pour créer des applications Amplify par programmation à partir de la ligne de commande. Pour plus d'informations, consultez la AWS CLI documentation.

Support pour le balisage des ressources

Vous pouvez utiliser le AWS Command Line Interface pour étiqueter les ressources Amplify. Pour plus d'informations, consultez la documentation AWS CLI tag-resource.

API d'hébergement Amplify

Cette référence fournit des descriptions des actions et des types de données pour l'API d'hébergement Amplify. Pour plus d'informations, consultez la documentation de référence de l'API Amplify.

Historique du document pour AWS Amplify

Le tableau suivant décrit les modifications importantes apportées à la documentation depuis la dernière version de AWS Amplify.

- Dernière mise à jour de la documentation : 8 septembre 2025

Modifier	Description	Date
Mises à jour des versions prises en charge de Node.js	Informations mises à jour sur les versions prises en charge par Node.js dans Déploiement d'applications rendues côté serveur avec Amplify Hosting.	8 septembre 2025
Chapitre sur les paramètres de build et la configuration mis à jour	Le Gestion de la configuration de compilation pour une application Amplify chapitre a été mis à jour pour décrire la nouvelle fonctionnalité de type d'instance de construction configurable qui vous permet de choisir un type d'instance fournissant à votre application les ressources de processeur, de mémoire et d'espace disque dont elle a besoin.	28 mai 2025
Chapitre sur le pare-feu mis à jour	Le Support de pare-feu pour les sites hébergés par Amplify chapitre a été mis à jour pour décrire la disponibilité générale (GA) de l'intégration d'Amplify AWS WAF, y compris les fonctionnalités GA et la structure tarifaire.	26 mars 2025

Modifier	Description	Date
Nouveau chapitre sur la protection Skew	Ajout du Protection antiobliq ue pour les déploiements d'Amplify chapitre décrivant la fonctionnalité de protectio n asymétrique qui élimine les problèmes de distorsion de version entre le client et les serveurs dans les applications Web Amplify.	10 mars 2025
Chapitre sur les webhooks mis à jour	Ajout d'une Webhooks unifiés pour les référentiels Git rubrique décrivant la fonctionn alité de webhooks unifiés qui utilise un webhook complet pour toutes les applications Amplify associées à un seul référentiel Git.	10 mars 2025
Nouveau : ajout d'un rôle de calcul SSR pour autoriser l'accès aux AWS ressources (rubrique)	Ajout d'une Ajout d'un rôle SSR Compute pour autoriser l'accès aux AWS ressources rubrique décrivant comment créer et associer un rôle Amplify SSR Compute à une application pour permettre au service Amplify Compute d'accéder aux ressources. AWS	17 février 2025

Modifier	Description	Date
Nouveau chapitre « Utiliser AWS WAF pour protéger vos applications Amplify »	Ajout du Support de pare-feu pour les sites hébergés par Amplify chapitre décrivant l'intégration d'Amplify AWS WAF (en version préliminaire) qui vous permet de protéger vos applications Web à l'aide d'une liste de contrôle d'accès Web (ACL Web).	18 décembre 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	14 novembre 2024
Support d'Amplify mis à jour pour la rubrique Next.js	Mise à jour de la Amplify le support pour Next.js rubrique pour décrire le support d'Amplify pour Next.js version 15.	6 novembre 2024
Nouveau chapitre Déploiement d'un site Web statique pour Amplify à partir d'Amazon S3	Ajout d'un Déploiement d'un site Web statique vers Amplify à partir d'un compartiment Amazon S3 chapitre décrivant la nouvelle intégration d'Amplify à Amazon S3, qui vous permet d'héberger du contenu de site Web statique stocké en quelques clics. S3	16 octobre 2024

Modifier	Description	Date
Nouveau chapitre sur la gestion de la configuration du cache	Ajout du Gestion de la configuration du cache pour une application chapitre décrivant le comportement de mise en cache par défaut d'Amplify et la manière dont il applique les politiques de cache géré au contenu.	13 août 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	18 juillet 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	31 mai 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	17 avril 2024

Modifier	Description	Date
Chapitre de démarrage mis à jour	Le Commencer à déployer une application sur Amplify Hosting chapitre a été mis à jour pour utiliser un exemple d'application Next.js dans le didacticiel.	12 avril 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	5 avril 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	4 avril 2024
Nouveau chapitre sur le dépannage	Ajout du Résolution des problèmes liés à Amplify Hosting chapitre décrivant comment résoudre les problèmes que vous rencontre z avec les applications déployées sur Amplify Hosting.	2 avril 2024

Modifier	Description	Date
Nouveau support pour les SSL/TLS certificats personnalisés	Ajout d'une Utilisation de SSL/TLS certificats rubrique au Connexion d'un domaine personnalisé chapitre pour décrire le support d'Amplify pour les SSL/TLS certificats personnalisés lors de la connexion d'une application à un domaine personnalisé.	20 février 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	2 janvier 2024
Nouveau support pour les frameworks SSR	Mise à jour de la Déploiement d'applications rendues côté serveur avec Amplify Hosting rubrique pour décrire le support d'Amplify pour tout framework SSR basé sur JavaScript avec un adaptateur open source.	19 novembre 2023
Nouveau support pour le lancement de la fonctionnalité d'optimisation des images	Ajout d'une Optimisation de l'image pour les applications SSR rubrique décrivant la prise en charge intégrée de l'optimisation des images pour les applications rendues côté serveur.	19 novembre 2023

Modifier	Description	Date
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	17 novembre 2023
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	6 novembre 2023
Nouvelle rubrique sur les sous-domaines génériques	Ajout d'une Configuration de sous-domaines génériques rubrique décrivant la prise en charge des sous-domaines génériques sur les domaines personnalisés.	6 novembre 2023
Nouvelle politique gérée par	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire la nouvelle politique AmplifyBackendDeployFullAccess AWS gérée pour Amplify.	8 octobre 2023

Modifier	Description	Date
Nouveau support pour les frameworks monorepo : lancement des fonctionnalités	Mise à jour de la Configuration des paramètres de compilation de monorepo rubrique pour décrire la prise en charge du déploiement d'applications dans des monorepos créés à l'aide de npm workspace, pnpm workspace, Yarn workspace, Nx et Turborepo.	19 juin 2023
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	1er juin 2023
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	24 février 2023
Chapitre sur le rendu côté serveur mis à jour	Le Déploiement d'applications rendues côté serveur avec Amplify Hosting chapitre a été mis à jour pour décrire les modifications récentes apportées à la prise en charge par Amplify des versions 12 et 13 de Next.js.	17 novembre 2022

Modifier	Description	Date
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	30 août 2022
Rubrique sur les politiques gérées mise à jour	Mise à jour de la Création d'un backend pour une application rubrique pour décrire comment déployer un backend à l'aide d'Amplify Studio.	23 août 2022
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	27 avril 2022
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	17 avril 2022
Lancement d'une nouvelle fonctionnalité de GitHub l'application	Ajout d'une Configuration de l'accès d'Amplify aux référentiels GitHub rubrique décrivant la nouvelle GitHub application permettant d'autoriser Amplify à accéder à GitHub votre référentiel.	5 avril 2022

Modifier	Description	Date
Lancement de la nouvelle fonctionnalité Amplify Studio	Mise à jour de la Bienvenue chez AWS Amplify Hosting rubrique pour décrire les mises à jour d'Amplify Studio qui fournissent un concepteur visuel pour créer des composants d'interface utilisateur que vous pouvez connecter à vos données principales.	2 décembre 2021
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify afin de prendre en charge Amplify Studio.	2 décembre 2021
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	8 novembre 2021
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	27 septembre 2021

Modifier	Description	Date
Nouvelle rubrique sur les politiques gérées	Ajout d'une AWS politiques gérées pour AWS Amplify rubrique décrivant les politiques AWS gérées pour Amplify et les modifications récentes apportées à ces politiques.	28 juillet 2021
Chapitre sur le rendu côté serveur mis à jour	Le Déploiement d'applications rendues côté serveur avec Amplify Hosting chapitre a été mis à jour pour décrire le nouveau support de Next.js version 10. x. x et Next.js version 11.	22 juillet 2021
Chapitre sur la configuration des paramètres de construction mis à jour	Ajout d'une Configuration des paramètres de compilation de monorepo rubrique décrivant comment configurer les paramètres de compilation et la nouvelle variable d'`AMPLIFY_MONOREPO_APP_ROOT` environnement lors du déploiement d'une application monorepo avec Amplify.	20 juillet 2021

Modifier	Description	Date
Chapitre actualisé sur les déploiements dans les branches de fonctionnalités	Ajout d'une Génération automatique de la configuration Amplify au moment de la construction (applications Gen 1 uniquement) rubrique décrivant comment générer automatiquement le `aws-exports.js` fichier au moment de la création. Ajout d'une Constructions de backend conditionnelles (applications Gen 1 uniquement) rubrique décrivant comment activer les builds de backend conditionnels. Ajout d'une Utiliser les backends Amplify dans toutes les applications (applications de première génération uniquement) rubrique décrivant comment réutiliser les backends existants lorsque vous créez une nouvelle application, connectez une nouvelle branche à une application existante ou mettez à jour un frontend existant pour qu'il pointe vers un environnement de backend différent.	30 Juin 2021

Modifier	Description	Date
Chapitre sur la sécurité mis à jour	Ajout d'une Protection des données dans Amplify rubrique décrivant comment appliquer le modèle de responsabilité partagée et comment Amplify utilise le chiffrement pour protéger vos données au repos et en transit.	3 juin 2021
Nouveau support pour le lancement de la fonctionnalité SSR	Ajout du Déploiement d'applications rendues côté serveur avec Amplify Hosting chapitre décrivant le support d'Amplify pour les applications Web qui utilisent le rendu côté serveur (SSR) et sont créées avec Next.js.	18 mai 2021
Nouveau chapitre sur la sécurité	Ajout du Sécurité dans Amplify chapitre décrivant comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amplify et comment configurer Amplify pour atteindre vos objectifs de sécurité et de conformité.	26 mars 2021

Modifier	Description	Date
Rubrique sur les builds personnalisés mise à jour	Mise à jour de la rubrique Images de construction personnalisées et mises à jour des packages en direct pour décrire comment configure r une image de construction personnalisée hébergée dans Amazon Elastic Container Registry Public.	12 mars 2021
Rubrique de surveillance mise à jour	Mise à jour de la rubrique Surveillance pour décrire comment accéder aux données CloudWatch des métriques Amazon et définir des alarmes.	2 février 2021
Nouveau sujet de CloudTrail journalisation	Ajout de la AWS CloudTrail rubrique Logging Amplify API using pour décrire comment AWS CloudTrail capture et enregistre toutes les actions d'API pour la référence d'API de AWS Amplify console et la référence d'API AWS Amplify d'interface utilisateur d'adminis tration.	2 février 2021

Modifier	Description	Date
Lancement de nouvelles fonctionnalités de l'interface utilisateur d'administration	Mise à jour de la Bienvenue chez AWS Amplify Hosting rubrique pour décrire la nouvelle interface utilisateur d'administration qui fournit une interface visuelle permettant aux développeurs Web et mobiles de créer et de gérer des backends d'applications en dehors de. AWS Management Console	1er décembre 2020
Lancement d'une nouvelle fonctionnalité du mode performance	Mise à jour de la rubrique Gestion des performances des applications pour décrire comment activer le mode performance afin d'optimiser les performances d'hébergement plus rapides.	4 novembre 2020
Mise à jour de la rubrique sur les en-têtes personnalisés	Mise à jour de la rubrique En-têtes personnalisés pour décrire comment définir des en-têtes personnalisés pour une application Amplify à l'aide de la console ou en modifiant un fichier YML.	28 octobre 2020

Modifier	Description	Date
Lancement de la nouvelle fonctionnalité de sous-domaines automatiques	Ajout de la rubrique Configurer des sous-domaines automatiques pour un domaine personnalisé Route 53 afin de décrire comment utiliser les déploiements de branches de fonctionnalités basés sur des modèles pour une application connectée à un domaine personnalisé Amazon Route 53. Ajout de la rubrique Accès à l'aperçu Web avec les sous-domaines pour décrire comment configurer les aperçus Web à partir de pull requests afin qu'ils soient accessibles avec les sous-domaines.	20 juin 2020
Nouveau sujet de notifications	Ajout de la rubrique Notifications pour décrire comment configurer des notifications par e-mail pour une application Amplify afin d'avertir les parties prenantes ou les membres de l'équipe en cas de réussite ou d'échec d'un build.	20 juin 2020

Modifier	Description	Date
Mise à jour de la rubrique sur les domaines personnalisés	Mise à jour de la Connexion d'un domaine personnalisé rubrique afin d'améliorer les procédures d'ajout de domaines personnalisés dans Amazon Route 53 et Google Domains. GoDaddy Cette mise à jour inclut également de nouvelles informations de dépannage pour la configuration de domaines personnalisés.	12 mai 2020
AWS Amplify libération	Cette version présente Amplify.	26 novembre 2018

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.