AWS クラウドでアプリケーションをモダナイズするための段階的なアプローチ

AWS 規範ガイダンス

AWS 規範ガイダンス: AWS クラウドでアプリケーションをモダナイズするための段階的なアプローチ

Table of Contents

AWS クラウドでアプリケーションをモダナイズするための段階的なアプローチ

ビジェイ・トゥマとアシッシュ・アメタ、Amazon Web Services (AWS)

2023 年 5 月 (ドキュメント履歴)

モダナイゼーションには、新しいテクノロジーを採用・利用し、ポートフォリオ、アプリケーション、インフラストラクチャーの価値をより迅速に提供し、最適な価格でスケーリングできるようにするための多角的なアプローチが必要です。アプリケーションを最適化し、維持し、モダナイズされたモデルで混乱なく運用することが必要であり、ビジネス・オペレーション、アーキテクチャ、エンジニアリング全般を簡素化する必要があります。

モダナイゼーションはアプリケーションだけの問題ではなく、安全で柔軟な運用フレームワークを提供する最新のインフラストラクチャーを必要とします。ビジネス・プロセスの品質、可用性、敏捷性に関して、アプリケーションとインフラは切っても切れない関係にあります。インフラストラクチャを考慮せずにアプリケーションをモダナイズすると、全体的なコストが高くなり、パフォーマンスと品質に悪影響を及ぼします。最新のアプリケーションは、新しいアーキテクチャパターン、運用モデル、ソフトウェア配信プロセスを組み合わせて構築されています。ゼロから数百万のユーザーまでスケールアップ、スケールダウンし、 (ペタバイトとは言わないまでも) テラバイト単位のデータを管理し、グローバルに利用可能で、ミリ秒単位で応答します。Amazon Web Services (AWS) クラウドで管理するワークロードのポートフォリオをモダナイズする場合、コンテナ、サーバーレス技術、専用データストア、ソフトウェア自動化を使用して、これらのワークロードをリプラットフォーム、リファクタリング、リプレースし、 AWSが提供する俊敏性と総コスト最適化 (TCO) のメリットを最大限に活用します。

このガイドは、アプリケーションオーナー、ビジネスオーナー、アーキテクト、テクニカルリード、プロジェクトマネージャーを対象としています。このガイドでは、モダナイゼーション評価フェーズで選択されたアプリケーションの基礎的な機能を開発する方法や、段階的アプローチを用いてモダナイゼーションの取り組みを加速させる方法について説明します。

このガイドは、 が推奨するアプリケーションのモダナイゼーションアプローチをカバーするコンテンツシリーズの一部です AWS。このシリーズには以下のものも含まれます:

- AWS クラウドでアプリケーションをモダナイズするための戦略
- AWS クラウド内のアプリケーションのモダナイゼーション準備状況の評価
- マイクロサービスへのモノリスの分解

- AWS サーバーレスサービスを使用したマイクロサービスの統合

ターゲットを絞ったビジネス成果

アプリケーションのモダナイゼーションに対する段階的アプローチによって、以下のような成果が期待できる：

- ビルド・アンド・プルーブ・アプローチやマイクロサービスなどのクラウドネイティブ・アーキテクチャを活用することで、より迅速にイノベーションを起こすための組織的な能力とキャパシティ。
- トレーニングとツールの改善を通じて組織を構築する変更管理および運用モデル。
- チーム・アプローチにより、最短 12 週間で最初の成果を出し、体験的な学習を提供し、独立した持続的なカスタマーの成功を可能にします。
- マイクロサービス、API、再利用可能なコンポーネント、コンテナ化に基づくコンポーザブル・アプリケーション・アーキテクチャ。
- 厳選された戦略的アプリケーションのための拡張可能なモダナイゼーションロードマップであり、スプリット・アンド・シードモデルで作業するための規定ガイダンスを含みます。このモデルでは、モダナイゼーションの機能とサービスが複数のエンジニアリングチームに分散され、ビジネスの成果に焦点を当てます。新しい最小実行可能製品 (MVPs) が定義されると、最初のチームメンバーは新しい製品チームを作るために分裂します。

モダナイゼーションプロセス

モダナイゼーションへの段階的アプローチの目標は、モダナイゼーションの側面を利用して価値を段階的に提供し、それをビジネスの差別化要因となる中核的なアプリケーションのサブセットに適用し、最新テクノロジーの採用を加速することです。段階的アプローチは次の 3 つのステップで構成されます。

1. モダナイゼーション診断プレイブックを使用して、アプリケーションの成熟度を評価します。採用に対して包括的なアプローチをとり、意図したビジネス成果に沿ったプロセスを開発します。
2. 早期に段階的にビジネス成果を上げることで、勢いを得るために、小さく始めて MVP を提供する準備をします。このフェーズには通常 12 ~ 16 週間かかります。
3. スプリットアンドシードモデルで機能するスケーラブルなモダナイゼーションロードマップを作成します。

以下のセクションでは、これらの手順について詳しく説明します。

トピック

ステップ 1. アプリケーションの評価

このフェーズの目標は以下のとおりです。

- アプリケーションの状況を十分に理解し、最新のデータプラットフォーム向けにアプリケーションを準備することで、ビジネスに影響を与えずに価値実現までの時間を短縮し、モダナイズ、最適化、スケーリングを行うことができます。
- アプリケーションランドスケープのプロファイルを作成して、変更に伴うメリット、リスク、コストを特定します。
- 戦略と計画から、導入、移行、アプリケーションのモダナイゼーション、継続的なサポートまで、エンドツーエンドのサービスを提供します。
- 継続的なビジネス価値を提供するために、再利用可能なプラクティスとツールを提供するポリシー、推奨、コントロールを構築します。

評価段階では、アプリケーションオーナーとアーキテクトはモダナイゼーション診断プレイブックを使用して、モダナイゼーションの目標と優先事項を検証します。

モダナイゼーション診断プレイブックの使用

モダナイゼーション診断プレイブックは、企業にとって現在の状態から将来の状態へ移行する価値を決定するためのプロセスを提供します。これには、モダナイゼーションに伴うテクノロジーの変化も含まれます。

診断プレイブックを使用して、クラウドのモダナイゼーションにおけるアプリケーションまたはアプリケーションスイートの優先順位を決定し、モダナイゼーションの際に対処する必要のあるコンポーネントを特定します。

診断サイズ

モダナイゼーション診断プレイブックは、アプリケーションまたはアプリケーショングループの現在とターゲット (移行後) の状態について、以下の側面を理解するのに役立ちます：

- アプリケーションのグループ化 — モダナイゼーションの対象となるアプリケーションを (たとえば、テクノロジーや運用モデルごとに) グループ化する理由はありますか?
- シーケンシング — 依存関係に基づいてアプリケーションをモダナイズすべき順序はありますか?
- テクノロジー — テクノロジーのカテゴリーは何ですか (例えば、ミドルウェア、データベース、メッセージング) ？
- 依存関係 — アプリケーションは他のシステムやミドルウェアと重要な依存関係を持っていますか?
- 環境 — 開発環境、テスト環境、本番環境はいくつ使用されていますか？
- ストレージ - ストレージ要件は何ですか (例えば、テストデータのコピー数)。
- 運用モデル - アプリケーションのすべてのコンポーネントが、継続的インテグレーションと継続的デリバリー (CI/CD) パイプラインを採用できますか。
  - もしそうなら、どのようなインフラ責任をアプリケーション· チームに分配し、誰に分配すべきですか。
  - そうでない場合、どのようなインフラ責任 (例えば、パッチ適用など) を運用チームに残すべきですか。
- デリバリーモデル：
  - アプリケーションやアプリケーション群に基づいて、リプラットフォーム、リファクタリング、リライト、リプレイスのどれを行うべきですか。

    - モダナイゼーションのどの部分にクラウドネイティブサービスを使用すべきですか。
- スキルセット — どのような専門知識が必要ですか? 例：
    - コンテナやサーバーレスの技術を一から使い、モジュラーアーキテクチャでアプリケーションを構築するクラウドアプリケーションのバックグラウンドです。
    - オープンソースと AWS ツールおよびサービスを使用して、CI/CD プロセス、Infrastructure as Code、自動化またはアプリケーションのオブザーバビリティの分野でソリューションを開発する DevOps の専門知識。
- モダナイゼーションアプローチ — アプリケーションの現状、クラウドテクノロジーの選択肢、現在の技術的負債、CI/CD、監視、スキル、運用モデルを考慮すると、どのような技術的移行作業を行う必要があるのでしょうか。
- モダナイゼーションのタイミング — モダナイゼーションのタイミングに影響を与える可能性のある、事業ポートフォリオのタイミングやその他の計画的な作業上の考慮事項は何ですか？
- インフラストラクチャのユニットコストと総コスト – 経済分析に基づくと AWS、オンプレミスとのワークロードを維持するための年間コストはいくらですか?

これらの側面に照らしてアプリケーションを評価することで、クラウドへのモダナイゼーションを推進するにあたり、ビジネス、テクノロジー、経済に定着し続けることができます。

ビルディングブロック

アプリケーションをモダナイズする場合、観察結果をビジネスアジリティ、組織のアジリティ、エンジニアリングの有効性という 3 つの構成要素に分類できます。

- ビジネスアジリティ ー ビジネスニーズを要件に変換するための、ビジネス内の効果に関係するプラクティス。デリバリー組織がビジネスの要求にどれだけ迅速に対応できるか、また、本番環境への機能リリースをビジネスがどれだけコントロールできるかのです。
- 組織のアジリティ — デリバリープロセスを定義するプラクティス。例としては、アジャイル方法論や DevOps セレモニーのほか、役割の割り当てと明確化、組織全体にわたるコラボレーション、コミュニケーション、支援などがあります。
- エンジニアリングの有効性 — 品質保証、テスト、CI/CD、構成管理、アプリケーション設計、ソースコード管理に関連する開発プラクティス。

測定基準の特定

顧客にとって重要なことを提供しているかどうかを知るには、改善を促進し、提供を早めるための対策を講じる必要があります。目標、質問、測定基準 (GQM) は、測定基準がこれらの基準を満たしていることを確認するための効果的な枠組みを提供します。このフレームワークを使い、以下のステップで目標から逆算する：

1. 取り組んでいる目標や成果を特定します。
2. 目標が達成されているかどうかを判断するために答えなければならない質問を導き出します。
3. 質問に適切に答えるために何を測定すべきか、何を測定できるかを決めます。対策には 2 つのカテゴリがあります。
   - 製品メトリクスは、顧客にとって重要なものを提供していることを確認するものです。
   - ソフトウェアデリバリライフサイクルを改善していることを確認するための運用指標。

製品指標

製品メトリクスはビジネス成果に焦点を当て、新しい業務範囲の投資収益率 (ROI) が決定された時点で確立する必要があります。製品指標を確立するうえで役立つ手法は、その新しい作業範囲が導入されたときに、ビジネスで何が変わるかを確認することです。この考え方を定式化して、モダナイゼーション機能が提供されたらどうなるかに焦点を当てたテスト形式にすると便利です。

たとえば、トランザクションをレガシーシステムから移行することで、クライアントをオンボーディングする新しい機会が開かれると考えているなら、その改善策とは何ですか？ 次のクライアントをオンボーディングするには、どのくらいのキャパシティを作成する必要がありますか？ その結果を検証するためのテストはどのように構築されるのでしょうか。このシナリオの場合、製品の評価基準には次のようなものが考えられる：

- ビジネス価値のテストや仮説を特定します (例えば、トランザクション容量の x%を解放することで、y%の新規ビジネスを取り込むことができる)。
- ベースラインを確立します (例えば、x 件の現在の取引能力が y 件の顧客をサポートしている)。
- 結果を検証します (たとえば、キャパシティが x パーセント向上したので、今では y パーセントの新規ビジネスに対応できるか？)

運用メトリクス

ソフトウェアデリバリーのライフサイクルを改善し、モダナイゼーションを加速させているかどうかを判断するには、ソフトウェア配信のリードタイムと実装時間を把握する必要があります。つまり、ビジネスニーズをどれだけ早く本番環境の機能に変換できるかということです。

運用上の有用な指標には以下が含まれます。

- リードタイム — ある作業範囲が要求から製造に移行するまでにどれくらいの時間がかかりますか？
- サイクルタイム — ある範囲の作業を、開始から終了まで実施するのにどれくらいの時間がかかりますか？
- デプロイ頻度 — どのくらいの頻度で変更を本番環境にデプロイしますか?
- サービスの復旧に要する時間 — 障害からの回復にはどのくらいの時間がかかりますか (平均修復時間またはMTTRとして測定)。
- 変更障害率 — 平均故障間隔 (MTBF) は？

ステップ 2。小規模から始めて、勢いをつけましょう。

このステップのゴールは、勢いをつけるために、最初の実行可能な最小限の製品 (MVP) を提供することです。このアプローチにより、早い段階から段階的に業績を上げることができます。

優先度ドライバーの検証

アプリケーションチームとモダナイゼーション作業を開始する前に、先に決定した優先度を検証することをお勧めします。以下の手順に従ってください。

1. 診断プレイブックから必要な情報をコンパイルします。
   - 優先アプリケーションのリストから、優先ドライバーと実現可能性評価を集めます。
   - アプリケーションの移行状態と目標状態の決定事項を収集します。
   - クラウドのモダナイゼーション計画におけるアプリケーション所有者、アーキテクト、利害関係者を特定します。
   - 依存関係やアプリケーションスイートの順序がわかっている場合は、その情報を求めます。
   - インベントリエントリが依存関係やアプリケーションスイートのグループとどのように関連しているかを判断します。アプリケーションには、他のコンポーネントと緊密に結合したり、他

のコンポーネントに依存したりする個別のコンポーネントがあり、それらのコンポーネントをまとめてモダナイズしたい場合があります。

2. ステップ 1 の担当者との 1 時間または 2 時間のミーティングをスケジュールして、優先度の高いドライバーを検証します。
   - ソリューションエンジニアやアーキテクトごとに複数 (最大 3~4 個) のアプリケーションをグループ化し、アプリケーションの依存関係やアプリケーションスイートの情報に基づいて、1 回の会議で話し合うようにします。
   - 今度のミーティングにおける各チームメンバーの役割と期待を決めます。
3. ミーティングを実施します。

詳細の確定

前のセクションで説明したプロセスに従って優先順位を検証したら、詳細を収集してモダナイゼーションのアプローチとタイミングを決定できます。

このフェーズでは、コアチームはアプリケーションチームと協力して短い 2 日間のスプリントを行い、 AWS クラウド上のアプリケーションの将来の状態を設計します。アクティビティには、製品定義、製品発見、ストーリー作成、バリューストリームマッピング、CI/CD プロセスの設計などがあります。いくつかのアイデアを紹介しよう：

- アプリケーションの各コンポーネント (例えば、ネットワーク構成、ストレージ構成、データベース、サーバー、アプリケーションのサーバーへの配置方法) をモデル化します。
- コンテナやサーバーレステクノロジーなどのツールを使用して、そのモデルをさまざまな構成要素や構成に分解します。
- アプリケーションの機能を、基盤となるインフラストラクチャーへの依存関係とは切り離します。アプリケーションの機能を、ソースコードを変更せずに移動できるコンポーネントに抽象化します。
- CI/CD ツールとメカニズムを使用して DevOps と緊密に統合します。

基盤となるプラットフォーム・サービスを構築し、アプリケーションをモダナイズする

この 12 週間のフェーズでは、コアチームはフルスタックチームによってサポートされ、優先度の高いビジネスユースケースを実現します。この作業は複数のツーピザチームによって行われます。た

とえば、基盤となるプラットフォームサービスを開発するためにプラットフォームエンジニアリングチームが結成され、新しいビジネス成果をもたらす製品チームが結成されます。

- プラットフォーム・エンジニアリング・チームは、クラウド基盤、開発者ワークフロー、データ分析機能をサポートする AWS サービスの設定、統合、カスタマイズします。大規模で複雑な企業では、これらの機能をそれぞれサポートする複数のチームがあるかもしれません。
- プロダクトチームは、事業開始段階で優先順位付けされたビジネス成果を実現するために、新しいサービスとエクスペリエンスを開発します。プロダクトチームは新しいサービスを開発すると同時に、中核となるビジネス機能のモダナイズも行います。

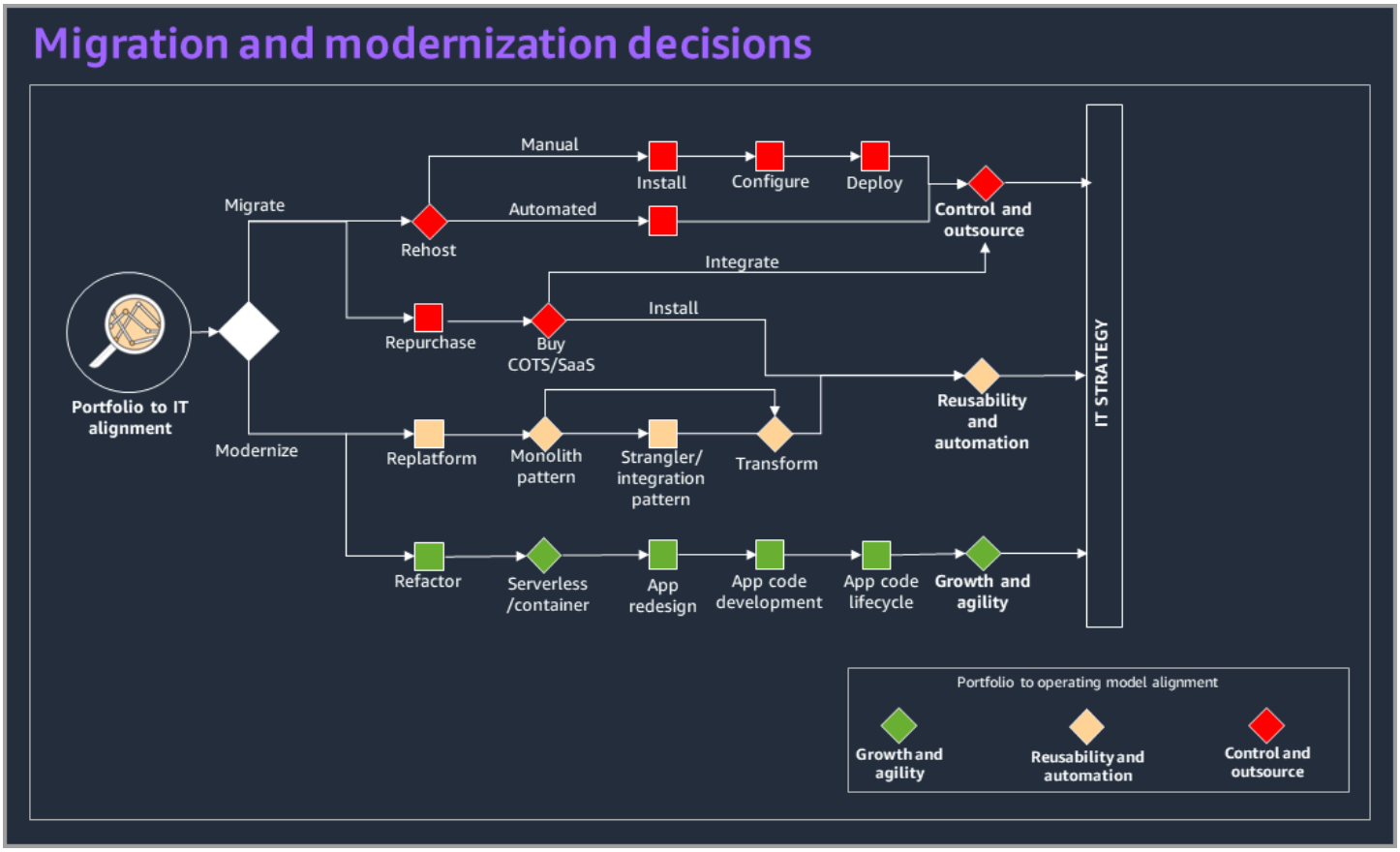
プラットフォーム・エンジニアリング・チームと製品チームは、あなたが評価できる最小限の実行可能製品 (MVP) を提供します。最初の MVP が成功したら、スプリットアンドシード方式を使用してモダナイゼーションプログラムを拡大できます。この方法では、新しいアプリケーションを特定し、最初のチームメンバーを分割して新しい製品チームを作成します。

ステップ 3。スケーラブルなモダナイゼーションロードマップを作成する

優先順位付けされた成果とアプリケーションの初回 MVP リリース後に、モダナイゼーションの取り組みをスケーリングして加速させ、開発者の生産性を向上させ、イノベーションを迅速に進めるためのロードマップを作成することをお勧めします。コアチームは、ビジネスの成果に重点を置く複数のエンジニアリングチームに組織の能力とサービスを拡大するために、チームを分割して新しいチームを編成します。スプリットアンドシードアプローチを徐々に取り入れることで、組織はより多くの開発を引き受け、モダナイゼーションのスピードを加速させることができます。

モダナイゼーションロードマップは、イベント駆動型、ストラングラー型、ドメイン駆動型設計、分解、最新のデータベース・オプションなど、明確に定義されたパターンを用いて、アプリケーションのモダナイゼーションへの実用的かつ継続的なアプローチを概説する必要があります。

ロードマップには、以下の図に示すように、アプリケーションのコンポーネントを特定し、ビジネスロジックに変更を加えることなく、マネージドサービス (データベースサービスなど) に移行する、あるいは、パフォーマンス、スケーラビリティ、管理性、信頼性、リソース使用量を改善するためにコードレベルの変更を加えるための、デシジョンツリーマトリックスを含める必要があります。


Migration and modernization decisions
Portfolio to IT alignment
Migrate
Modernize
Rehost
Manual
Automated
Install
Configure
Deploy
Control and outsource
Repurchase
Buy COTS/SaaS
Integrate
Install
Reusability and automation
Replatform
Monolith pattern
Strangler/ integration pattern
Transform
Refactor
Serverless /container
App redesign
App code development
App code lifecycle
Growth and agility
IT STRATEGY
Portfolio to operating model alignment
Growth and agility
Reusability and automation
Control and outsource

モダナイゼーション準備要因

アプリケーションをモダナイズするときは、以下の標準とベストプラクティスを遵守する。

トピック

コード

- ソフトウェアの機能を説明するコードコメントを入力し、それを使用してドキュメントを生成します。
- 頻繁なコードチェックインと機能リクエストへのトレーサビリティをサポートするコード管理とデプロイプロセスに従います。
- ユニットテスト、機能テスト、パフォーマンステスト、クリティカルパステストを含むテストスイートを 100% コードカバレッジで構築します。
- コードの再利用を奨励し、コードベースで同一または類似の機能を提供します。
- 完全なコード開発に投資する前に、プロトタイプを開発してユーザーと機能を検証します。

ビルドとテストを行う

- テストに基づいて機能の完全性を再定義することで、品質を向上させ、問題の再発を防ぎます。
- 承認テストを自動化します。
- すべての自動テストを監視し、その場で障害を処理するためのプロセスを確立します。
- 本番環境と非本番環境の両方でパフォーマンスを追跡し、現実的なトラフィックと負荷テストに基づいてサービスレベル目標 (SLO) を定義し、パフォーマンス要件を満たすための拡張機能を提供します。
- 構成ファイルから機密データを抽出し、構成を自動化および監視するツールを提供します。

リリース

- 依存関係 (データベースのリリースなど) 、回帰テスト、トラッキングをサポートし、デプロイメントを自動化します。
- ビルドが成功するたびに、コードを本番環境に段階的にリリースします。
- 機能フラグ (トグル) の効率的な管理：ランタイム設定をサポートし、使用状況を監視し、開発サイクルを通じてフラグを維持し、カテゴリーごとに所有者を割り当てます。
- ビルドパイプラインにトレーサビリティを提供し、トリガー、失敗通知、正常終了を追跡します。
- 自動デプロイプロセスを実行し、継続的デリバリーで「ゼロタッチ」コード更新をテストします。
- ダウンタイムがゼロで完全に自動化されたブルー/グリーンデプロイ方法論を使用します。
- データベーススキーマの変更が、すべての開発環境と実稼働環境で一貫して実装されていることを確認します。

運用

- 通知システムと統合された DevOps トリアージランブックを作成します。
- 監視および通知システムがサービスレベル目標 (SLO) を満たし、しきい値、ヘルスチェック、非標準HTTPレスポンス、および予期せぬ結果をサポートしていることを確認します。
- 効果的なリスク管理と災害復旧プロセスを確立します。
- ビジネス要件と法的要件を満たすログのローテーションと保存戦略を策定します。
- 製品のパフォーマンスを追跡し、新機能の効果を測定し、指標が期待どおりでない場合はアラートを表示するダッシュボードを開発します。

最適化

- パフォーマンスと品質の測定に基づいて、プロセスを定期的に見直し、改善します。
- 問題の再発を防ぐため、根本原因分析と予防プロセスを導入します。
- 製品の状態を把握するデータ主導型の指標を提供し、すべての通知とアクションがこれらの指標に基づいていることを確認します。

準備状況

- 部門横断的なチーム (ビジネスパートナー、開発者、テスター、アーキテクトを含む) をモダナイゼーションの取り組みに専念させます。

次のステップ

このガイドでは、アプリケーションをモダナイズするための段階的かつ漸進的なアプローチを提供しました。ワークストリームの範囲を、短期、中期、長期のテクノロジーとデリバリーから検討することから始めることをお勧めします。チーム全体でより効果的なデリバリーとエンジニアリングプラクティスを推進するために、ツール、フレームワーク、プラクティスにもっと投資することが期待できます。モダナイゼーションプロセスの目標を設定し、モダナイゼーションを計画している各アプリケーションを理解し、各アプリケーションに最適なモダナイゼーションのアプローチを選択します。進捗状況をモニターし、最適化する。

AWS は組織のビジネス開発チームと連携して、テクノロジーのモダナイゼーションを推進し、アプリケーションのアーキテクチャ、設計、開発、モバイル対応、テスト、コラボレーションソリューションなど、エンドツーエンドのニーズを管理します。これらのサービスは、実証済みのプロセス、インテリジェントオートメーション、データとパターンの活用、オープンスタンダード、適切な人材を組み合わせて、レガシーアプリケーションのモダナイズを支援します。主な目標は以下のとおりです。

- レガシーテクノロジーをモダナイズするための完全に自動化されたツールベースのアプローチを提供するプラットフォームの開発を支援し、それをモダナイゼーションに関する総合的な知識と組み合わせます。
- 顧客の成果と規模に重点を置いたモダナイゼーションエンジニアリング能力を備えた、フルスタックチームで MVP を構築しましょう。

AWS プロフェッショナルサービスとAWS パートナーは、テクノロジーやビジネス顧客の上級利害関係者と直接連携して、企業のモダナイゼーションを推進すると同時に、エンドカスタマーに繰り返し価値を提供します。

リソース

関連ガイド

- AWS クラウドでアプリケーションをモダナイズするための戦略
- AWS クラウド内のアプリケーションのモダナイゼーション準備状況の評価
- AWS クラウドでのオペレーションのモダナイズ
- マイクロサービスへのモノリスの分解
- AWS サーバーレスサービスを使用したマイクロサービスの統合
- AWS Change Acceleration 6-Pointフレームワークと組織変更管理ツールキット

AWS リソース

- AWS ドキュメント
- AWS 全般のリファレンス
- 「AWS 用語集」

ドキュメント履歴

以下の表は、本ガイドの重要な変更点について説明したものです。今後の更新に関する通知を受け取る場合は、RSS フィード をサブスクライブできます。

変更	説明	日付
ガイダンスの拡大	—	2023 年 5 月 25 日
初版発行	—	2020 年 12 月 18 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。