実装ガイド

AWS での生成 AI アプリケーションビルダー

AWS での生成 AI アプリケーションビルダー: 実装ガイド

Table of Contents

このソリューションを使用すると、生成人工知能 (AI) アプリケーションの開発、迅速な実験、デプロイが容易になります

AWS での生成 AI アプリケーションビルダーを使用すると、AI に関する深い経験がなくても、生成人工知能 (AI) アプリケーションの開発、迅速な実験、デプロイが容易になります。この AWS ソリューションは、以下を支援することで、開発を加速し、実験を合理化します。

- ビジネス固有のデータやドキュメントの取り込み
- 大規模言語モデル (LLM) のパフォーマンスの評価と比較
- AI エージェントを使用した複数ステップのタスクとワークフローの実行
- 拡張可能なアプリケーションの迅速な構築、エンタープライズグレードのアーキテクチャによるこれらのアプリケーションのデプロイ

AWS での生成 AI アプリケーションビルダーには、以下との統合が含まれています。

- Amazon Bedrock で利用可能な LLM
- Amazon SageMaker AI でデプロイした LLM
- 検索拡張生成 (RAG) 向け Amazon Bedrock のナレッジベース
- セーフガードの実装とハルシネーション低減のための Amazon Bedrock のガードレール
- タスクのオーケストレーションと完了を実行するエージェントワークフローを構築するための Amazon Bedrock エージェント
- Amazon Bedrock AgentCore は、ランタイムサポートが拡張された本番環境対応の AI エージェントを構築、デプロイ、管理します。
- エンタープライズデータとツール統合用のモデルコンテキストプロトコル (MCP) サーバー

さらに、このソリューションでは、LangChain コネクタを使用して、任意のモデルに接続できます。これらのコネクタは、このソリューションを使用してデプロイする AWS Lambda 関数で利用できます。コード不要のデプロイウィザードの使用を開始して、会話型検索、生成 AI 搭載チャットボット、テキスト生成、テキスト要約のための生成 AI アプリケーションを構築できます。

この実装ガイドでは、AWS での生成 AI アプリケーションビルダーの概要、そのリファレンスアーキテクチャとコンポーネント、デプロイを計画する際の考慮事項、Amazon Web Services (AWS) クラウドにソリューションをデプロイするための設定手順について説明します。

このガイドは、既存の環境への AWS での生成 AI アプリケーションビルダーの導入を検討しているソリューションアーキテクト、ビジネスの意思決定者、DevOps エンジニア、データサイエンティスト、クラウドプロフェッショナルを対象としています。

このナビゲーションテーブルを使用すると、次の質問に対する回答をすばやく見つけることができます。

質問内容	参照先
このソリューションの実行に必要なコストを確認する。 このソリューションを実行するための推定コストは、デプロイするコンポーネントとクエリの数によって異なります。 米国東部 (バージニア北部) リージョンにおいてデフォルトのパラメータで 100 人のアクティブユーザーが使用するデプロイダッシュボードを実行する場合の推定コストは、1 か月あたり 20.12 USD です。 LLM を使用して 1 人のビジネスユーザーが 1 日あたり 100 件のクエリを実行する、RAG なしでデプロイする Text ユースケースのコストは、1 か月あたり約 12.39 USD です。 1 日あたり 8,000 件のインタラクションをサポートする Amazon Kendra インデックスを使用した RAG 対応ユースケースのコストは、1 か月あたり約 204.26 USD で、別途ナレッジベースのコストが発生します。	コスト
このソリューションのセキュリティ上の考慮事項を理解する。	セキュリティ

質問内容	参照先
このソリューションのクォータを計画する方法を確認する。	クォータ
どの AWS リージョンでこのソリューションをサポートしているのかを確認する。	サポートしている AWS リージョン
このソリューションに含まれている AWS CloudFormation テンプレートを表示またはダウンロードして、このソリューションのインフラストラクチャリソース (スタック) を自動的にデプロイする。	AWS CloudFormation テンプレート
ソースコードにアクセスし、オプションで AWS Cloud Development Kit (AWS CDK) を使用してソリューションをデプロイする。	GitHub リポジトリ

機能とメリット

AWS での生成 AI アプリケーションビルダーソリューションは、以下の機能を提供します。

迅速な実験

このソリューションにより、ユーザーは設定の異なる複数のインスタンスをデプロイして出力とパフォーマンスを比較する手間のかかる作業を省き、迅速に実験を行うことができます。さまざまな LLM、プロンプトエンジニアリング、エンタープライズナレッジベース、ガードレール、AI エージェント、その他のパラメータについて複数の設定を試すことができます。

選択と設定可能性

Amazon Bedrock で利用可能なモデルなど、さまざまな LLM への事前構築済みコネクタを使用できるため、このソリューションでは、選択したモデルのみでなく、お好みの AWS サービスや主要な FM サービスを柔軟にデプロイできます。Amazon Bedrock エージェントを有効にして、さまざまなタスクやワークフローを実行することもできます。

エージェントビルダー

完全なライフサイクル管理を使用して、本番稼働対応の AI エージェントを構築およびデプロイします。システムプロンプトの設定、エンタープライズツールとデータアクセス用のモデルコンテキストプロトコル (MCP) サーバーの統合、会話中のコンテキスト保持のためのメモリ機能の有効化を行います。エージェントは Amazon Bedrock AgentCore にデプロイされ、ランタイムサポートが拡張され、リアルタイムストリーミングレスポンスが提供されます。

ワークフロービルダー

階層的な委任を使用して、複数のエージェントビルダーエージェントを複雑なワークフローにオーケストレーションします。マルチステップタスクを処理するために、特殊なエージェントビルダーエージェントを自律的に選択して調整する、スーパーバイザーエージェントを作成します。既存のエージェントビルダーデプロイを再利用しながら、エージェントの説明、委任戦略、ワークフローレベルのメモリを設定します。

本番環境対応

AWS Well-Architected の設計原則に基づいて構築されたこのソリューションは、高可用性と低レイテンシーを実現するエンタープライズグレードのセキュリティとスケーラビリティを提供し、高パフォーマンス基準でアプリケーションへのシームレスな統合を実現します。

拡張可能なモジュール型アーキテクチャ

既存のプロジェクトを統合するか、追加の AWS サービスをネイティブに接続することで、このソリューションの機能を拡張できます。このアプリケーションはオープンソースであるため、付属の LangChain オーケストレーションレイヤーや Lambda 関数を使用して、任意のサービスに接続できます。

AWS Systems Manager の機能である Service Catalog AppRegistry および Application Manager との統合

このソリューションには、CloudFormation テンプレートとその基盤となるリソースを AWS Service Catalog AppRegistry と AWS Systems Manager Application Manager の両方にアプリケーションとして登録するための Service Catalog AppRegistry リソースが含まれています。この統合により、ソリューションのリソースを一元管理できます。

エージェントビルダーと Bedrock エージェントのユースケース

このソリューションには、AI エージェントを操作するための 2 つの異なるアプローチがあり、それぞれが異なるユースケースと要件に適しています。

機能	Bedrock エージェントユースケース	エージェントビルダー
目的	事前定義された Amazon Bedrock エージェントを呼び出す	カスタムエージェントのビルド、デプロイ、管理
設定	エージェント ID とエイリアス ID のみ	エージェントの完全な設定: システムプロンプト、モデル、MCP サーバー、メモリ
デプロイ	シンプルな呼び出しレイヤー	AgentCore Runtime でのエージェントライフサイクルの完了
Runtime – 。	Amazon Bedrock エージェントサービス	Strands SDK を使用した Amazon Bedrock AgentCore
ツール統合	Bedrock エージェントコンソールで設定	モデルコンテキストプロトコル (MCP) サーバーと組み込みの Strands ツール
メモリ	Bedrock エージェントによって管理 (最大 30 日間)	設定可能な短期保持と長期保持を備えた AgentCore Memory
カスタマイズ	事前デプロイされたエージェント設定に制限	プロンプト、モデル、ツール、動作を完全に制御
次の用途に適しています	既存のエージェントの迅速なデプロイ	カスタムエージェント開発と本番デプロイ

> **Note**
>
> どちらのオプションも、リアルタイムストリーミング、会話履歴、エンタープライズグレードのセキュリティをサポートしています。

ワークフロービルダー

ワークフロービルダーは、作業を専門のエージェントビルダーのエージェントに委任するスーパーバイザーエージェントを作成することでマルチエージェントオーケストレーションを可能にします。各ワークフローは以下で構成されます。

- スーパーバイザーエージェント: ユーザーリクエストを受け取り、専門エージェントを調整するエントリポイントエージェント
- 専門エージェント: スーパーバイザーのタスク委任先となるエージェントビルダーのユースケース
- エージェントをツールとしてパターン化: スーパーバイザーは各エージェントビルダーエージェントをツールとして登録し、使用するエージェントを自律的に選択

機能	エージェントビルダー	ワークフロービルダー
目的	単一のカスタムエージェントを構築およびデプロイする	複数のエージェントビルダーエージェントをオーケストレーションする
Agent Type	MCP ツールを備えた単一エージェント	スーパーバイザーエージェント + 複数のエージェントビルダーエージェント
ツール統合	MCP サーバーと Strands ツール	ツールとして登録されたエージェントビルダーエージェント
委任	直接ツール呼び出し	自律的なエージェント選択と委任
複雑さ	単一エージェントタスク	マルチステップ、マルチエージェントワークフロー
エージェント再利用	該当なし	既存のエージェントビルダーデプロイを再利用
次の用途に適しています	フォーカスされた単一ドメインタスク	複数の専門分野を必要とする複雑なワークフロー

> **Note**
>
> - ワークフローでは、専門エージェントとして少なくとも 1 つのエージェントビルダーユースケースが必要です。
> - すべての専門エージェントは、GAAB にデプロイされたエージェントビルダーユースケースである必要があります。

ユースケース

企業データに関する質問への回答

LLM やその他の基盤モデルは、多くの自然言語処理 (NLP) タスクで優れたパフォーマンスを発揮できるように、大量のデータコーパスで事前トレーニングされています。ただし、ほとんどの基盤モデルや LLM は静的であり、事前トレーニング済みであるため、新しいトピック、専門的なトピック、または独自トピックに関する質問に正確に回答する能力が制限されています。プロンプトベース学習を使用すると、LLM の強力な NLP およびテキスト生成機能を活用して、企業データに比べ、より豊富なカスタマーエクスペリエンスを提供できます。

迅速な生成 AI プロトタイピング

このソリューションには、さまざまなモデルプロバイダーやユースケースがあらかじめバンドルされています。使いやすいデプロイウィザードを使用すると、事前構築済みのユースケースをデプロイして、さまざまな生成 AI プロトタイプやワークロードを迅速に実験できます。

複数の LLM の比較と実験

LLM によってパフォーマンスはさまざまであるため、アプリケーション固有のニーズを考慮すると、特定の LLM がその他の LLM よりも独自のアプリケーションに適している場合があります。これは、パフォーマンス、精度、コスト、創造性、またはその他の多くの要因に関連する理由が考えられます。このソリューションを使用すると、複数のユースケースを迅速にデプロイできるため、ニーズに合ったものが見つかるまで、さまざまな設定を試して比較できます。

概念と定義

このセクションでは、主要な概念について説明し、このソリューション固有の用語を定義します。

管理者ユーザー

管理者ユーザーとは、このガイドのコンテキストではデプロイに含まれるコンテンツの管理を担当するユーザーを指します。このようなユーザーは、デプロイダッシュボードの UI にアクセスでき、主にビジネスユーザーエクスペリエンスのキュレーションを担当します。このユーザーが、主にこのソリューションの対象となるお客様です。

ビジネスユーザー

ビジネスユーザーとは、このガイドのコンテキストではユースケースのデプロイ対象となる個人を指します。ビジネスユーザーは、ナレッジベースのコンシューマーであり、LLM の評価と実験を担当するお客様です。

デプロイダッシュボード

デプロイダッシュボードは、管理者ユーザーがユースケースを表示、管理、作成するための管理コンソールとして機能するウェブインターフェイスです。このダッシュボードにより、LLM を活用したさまざまな AI/ML ワークロードを迅速に実験および反復して本番稼働化できます。

DevOps ユーザー

DevOps ユーザーとは、このガイドのコンテキストでは、AWS アカウント内でのソリューションのデプロイ、インフラストラクチャの管理、ソリューションの更新、パフォーマンスのモニタリング、ソリューションの全体的なヘルスとライフサイクルの維持を担当するユーザーを指します。

ユースケース

ユースケースは、ソリューション全体から分離されたアプリケーションであり、LLM と統合して、新規アプリケーションや既存のアプリケーションに自然言語インターフェイスを追加できるようにすることで、カスタマーエクスペリエンスを向上することができます。ユースケースは、デプロイダッシュボードを介してデプロイすることも、ユーザーがデプロイすることもできます。

> **Note**
>
> AWS 用語の一般的なリファレンスについては、「AWS 用語集」を参照してください。

アーキテクチャの概要

このセクションでは、このソリューションでデプロイされるコンポーネントのリファレンス実装アーキテクチャ図を示します。

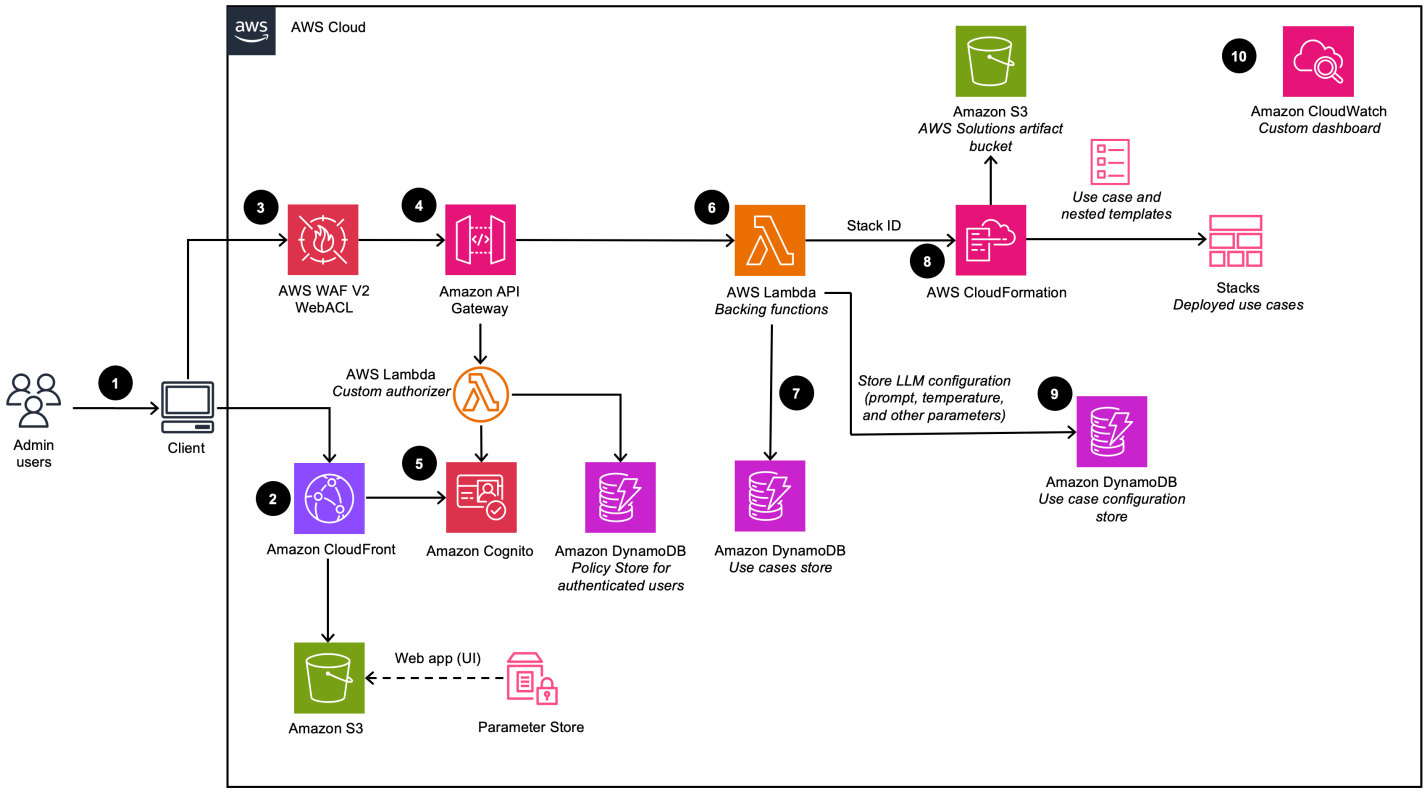
アーキテクチャ図

さまざまなユースケースとビジネスニーズをサポートするために、このソリューションでは 6 つの AWS CloudFormation テンプレートが用意されています。

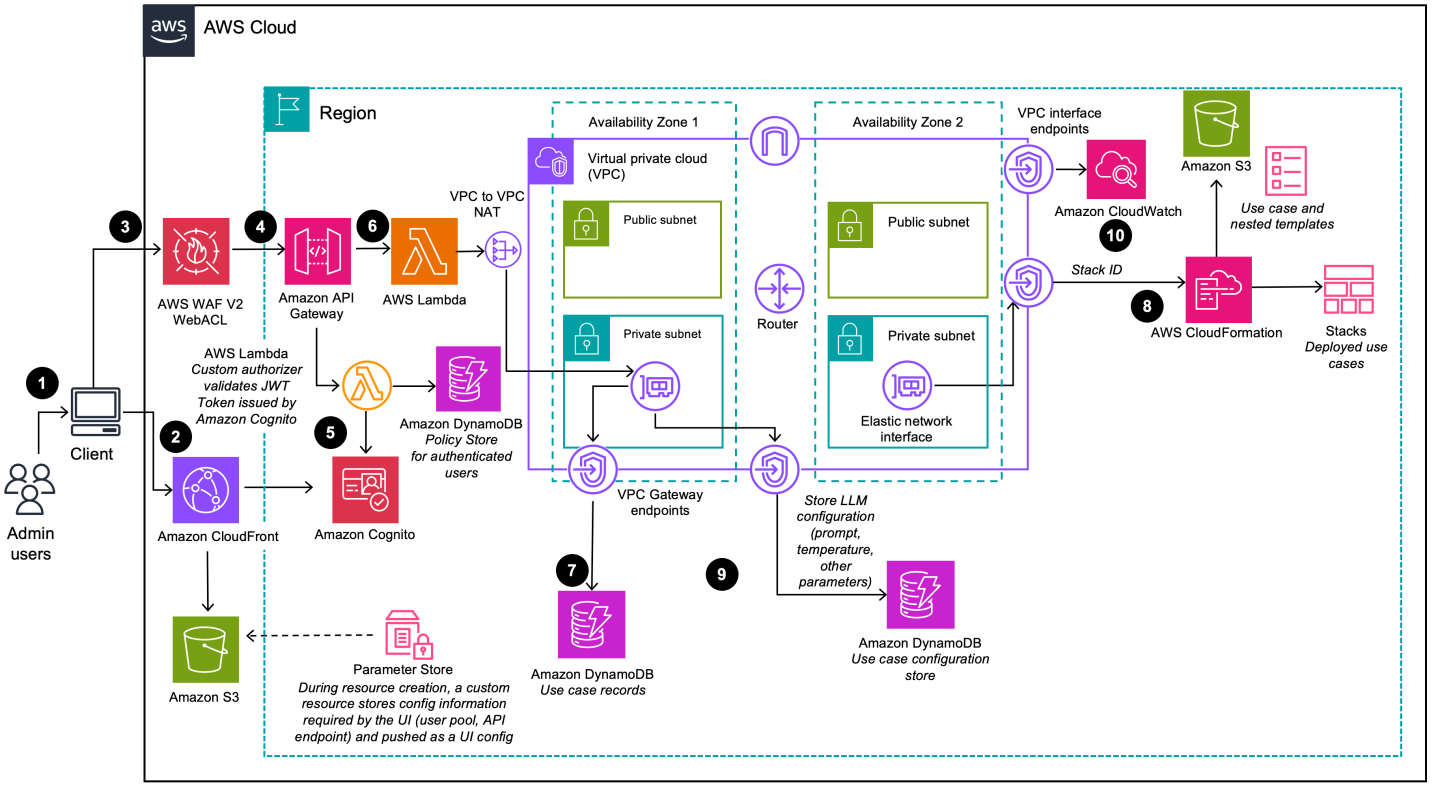
1. デプロイダッシュボード - デプロイダッシュボードは、管理者ユーザーがユースケースを表示、管理、作成するための管理コンソールとして機能するウェブインターフェイスです。このダッシュボードにより、LLM を活用したさまざまな AI/ML ワークロードを迅速に実験および反復して本番稼働化できます。
2. Text ユースケース - Text ユースケースでは、生成 AI を使用して自然言語インターフェイスを体験できます。このユースケースは、新規または既存のアプリケーションに統合でき、デプロイダッシュボードからデプロイすることも、提供された URL を通じて個別にデプロイすることもできます。
3. Bedrock エージェントユースケース - Bedrock エージェントユースケースでは、既存の Bedrock エージェントを使用してタスクを完了したり、繰り返しワークフローを自動化したりできます。
4. MCP サーバー - MCP サーバーのユースケースにより、AI アプリケーションへの標準化されたツールとリソースアクセスを提供するモデルコンテキストプロトコルサーバーのデプロイと管理が可能になります。既存の Lambda 関数、API、外部 MCP サーバーをラップするためのゲートウェイメソッドと、カスタムコンテナ化された MCP サーバーをデプロイするためのランタイムメソッドの両方をサポートします。
5. エージェントビルダー - エージェントビルダーを使用すると、完全な設定制御、MCP サーバー統合、メモリ管理機能を使用して、Amazon Bedrock AgentCore で本番稼働対応の AI エージェントを作成およびデプロイできます。
6. ワークフロービルダー - ワークフロービルダーを使用すると、複雑なマルチエージェントワークフローの Agents as Tools 委任パターンを使用して複数のエージェントビルダーエージェントをオーケストレーションするスーパーバイザーエージェントを作成できます。

デプロイダッシュボード

デプロイダッシュボードのアーキテクチャを示しています (VPC オプションを無効にしてデプロイした場合)



デプロイダッシュボードアーキテクチャを示しています (VPC オプションを有効にしてデプロイした場合)



> **Note**
>
> AWS CloudFormation のリソースは、AWS Cloud Development Kit (AWS CDK) のコンストラクトで作成されています。

AWS CloudFormation テンプレートを使用してデプロイされたこのソリューションコンポーネントの大まかなプロセスフローは次のとおりです。

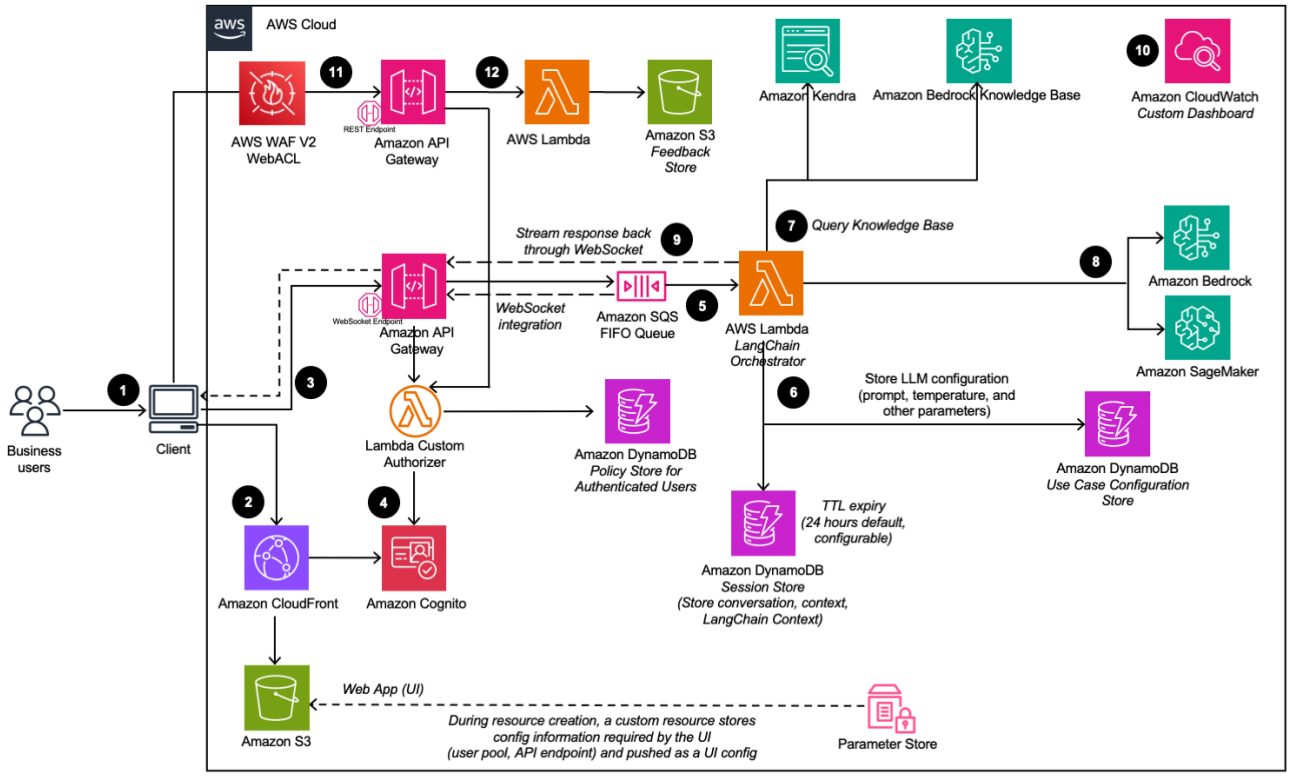
1. 管理者ユーザーは、デプロイダッシュボードのユーザーインターフェイス (UI) にログインします。
2. Amazon CloudFront が、Amazon Simple Storage Service (Amazon S3) バケットでホストされているウェブ UI を提供します。
3. AWS WAF は API を攻撃から保護します。このソリューションでは、ウェブアクセスコントロールリスト (ウェブ ACL) と呼ばれる一連のルールを設定して、設定可能なユーザー定義のウェブセキュリティルールと条件に基づき、ウェブリクエストを許可、ブロック、またはカウントします。
4. ウェブ UI は、Amazon API Gateway を使用して公開される一連の REST API を活用します。

5. Amazon Cognito はユーザーを認証し、CloudFront ウェブ UI と API Gateway の両方をサポートします。
6. AWS Lambda は、REST エンドポイントのビジネスロジックを提供します。このバッキング Lambda 関数は、AWS CloudFormation を使用してユースケースのデプロイを実行するために必要なリソースを管理および作成します。
7. Amazon DynamoDB はデプロイのリストを保存します。
8. 管理者ユーザーが新しいユースケースを作成すると、バッキング Lambda 関数は、リクエストされたユースケースの CloudFormation スタック作成イベントを開始します。
9. デプロイウィザードで管理者ユーザーが提供するすべての LLM 設定オプションは、DynamoDB に保存されます。デプロイでは、この DynamoDB テーブルを使用して、実行時に LLM を設定します。
10. このソリューションは、Amazon CloudWatch を使用してさまざまなサービスから運用メトリクスを収集し、ソリューションのパフォーマンスと運用状態をモニタリングできるカスタムダッシュボードを生成します。

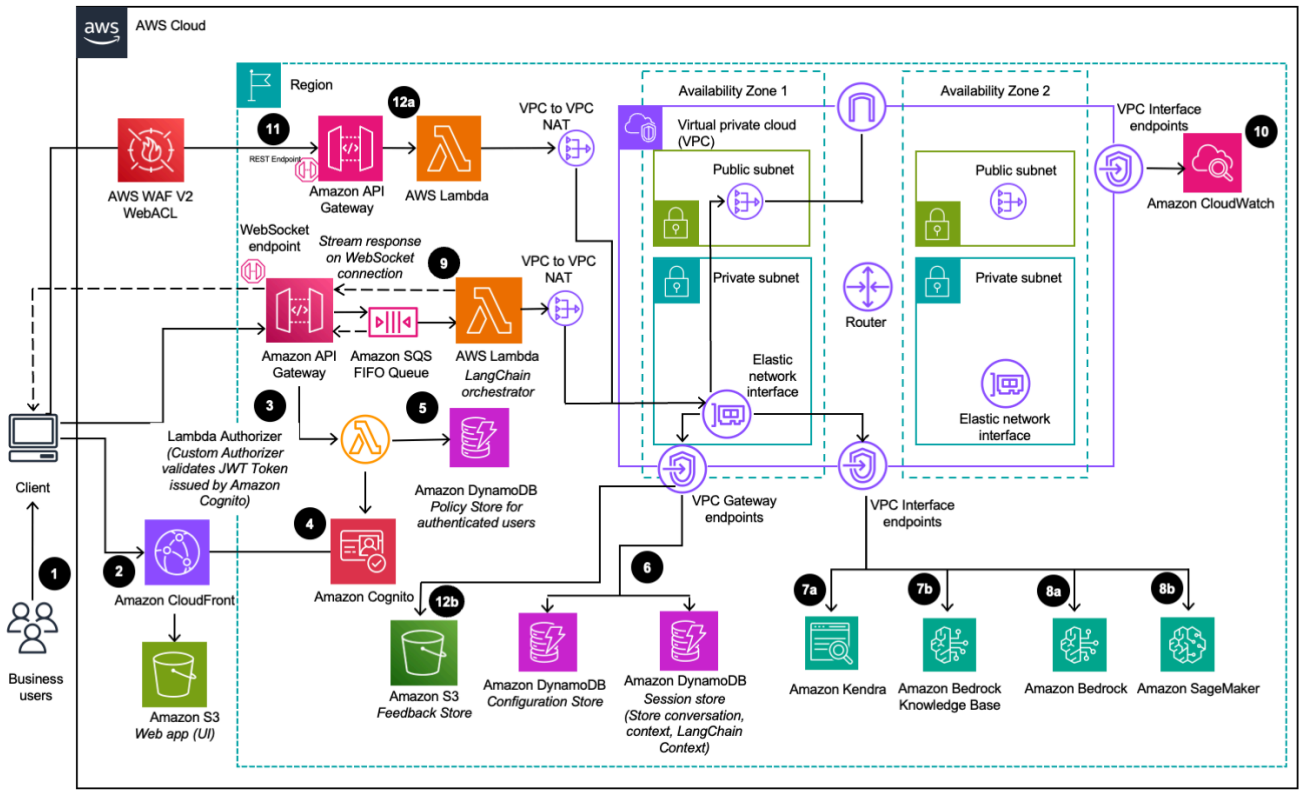
> **Note**
>
> - このソリューションを Amazon VPC にデプロイする場合、データはプライベートネットワーク内でルーティングされます。
> - デプロイダッシュボードはほとんどの AWS リージョンで起動できますが、デプロイされたユースケースには、サービスの可用性に基づいて特定の制限があります。詳細については、「サポートされている AWS リージョン」を参照してください。

Text ユースケース

Text ユースケースのアーキテクチャを示しています (VPC オプションを無効にしてデプロイした場合)



Text ユースケースのアーキテクチャを示しています (VPC オプションを有効にしてデプロイした場合)



AWS CloudFormation テンプレートを使用してデプロイされたこのソリューションコンポーネントの大まかなプロセスフローは次のとおりです。

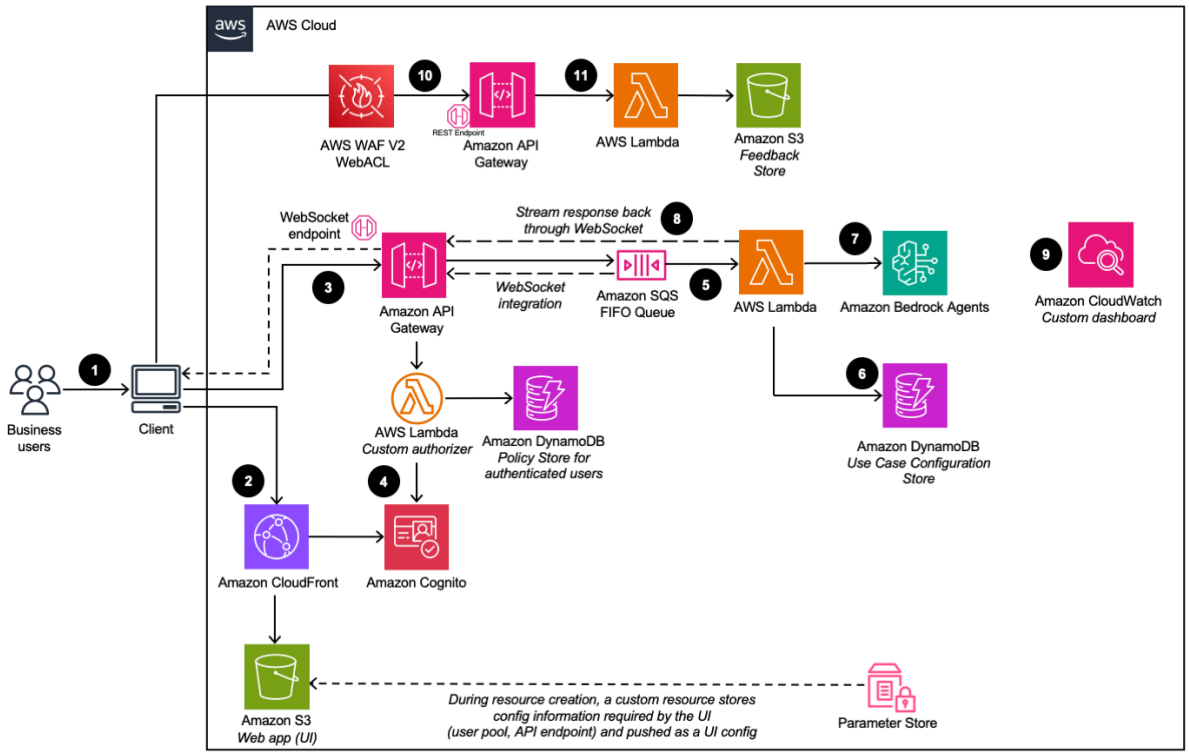
1. 管理者ユーザーは、デプロイダッシュボードを使用してユースケースをデプロイします。ビジネスユーザーは、ユースケースの UI にログインします。
2. CloudFront は、S3 バケットでホストされているウェブ UI を提供します。
3. ウェブ UI は、API Gateway を使用して構築された WebSocket 統合を活用します。API Gateway は、認証ユーザーが属する Amazon Cognito グループに基づいて適切な AWS Identity and Access Management (IAM) ポリシーを返すカスタム Lambda オーソライザー関数によってサポートされています。ポリシーは DynamoDB に保存されます。
4. Amazon Cognito はユーザーを認証し、CloudFront ウェブ UI と API Gateway の両方をサポートします。
5. ビジネスユーザーからの受信リクエストは、API Gateway から Amazon SQS キューに渡され、その後 LangChain Orchestrator に渡されます。LangChain Orchestrator は、ビジネスユーザーからのリクエストに応えるためのビジネスロジックを提供する Lambda 関数とレイヤーのコレクションです。キューにより、API Gateway と Lambda 統合の非同期操作が可能になります。キューは Lambda 関数に接続情報を渡し、その結果を API Gateway WebSocket 接続に直接送信して、長時間実行される推論呼び出しをサポートします。

6. LangChain Orchestrator は、Amazon DynamoDB を使用して、設定された LLM オプションと必要なセッション情報 (チャット履歴など) を取得します。
7. デプロイでナレッジベースが有効になっている場合、LangChain Orchestrator は Amazon Kendra または Amazon Bedrock ナレッジベースを利用して検索クエリを実行し、ドキュメントの抜粋を取得します。
8. LangChain Orchestrator は、ナレッジベースのチャット履歴、クエリ、コンテキストを使用して最終プロンプトを作成し、Amazon Bedrock または Amazon SageMaker AI でホストされている LLM にリクエストを送信します。
9. LLM から応答が返されると、LangChain Orchestrator は API Gateway WebSocket 経由で応答をストリーミングし、クライアントアプリケーションで使用できるようにします。
10. このソリューションは、Amazon CloudWatch を使用してさまざまなサービスから運用メトリクスを収集し、デプロイのパフォーマンスと運用状態をモニタリングできるカスタムダッシュボードを生成します。
11. フィードバック収集が有効になっている場合、Amazon API Gateway を活用した REST API エンドポイントがユーザーフィードバックの収集に使用できます。
12. フィードバックバッキング Lambda は、送信されたフィードバックを追加のユースケース固有のメタデータ (使用されているモデルなど) で補強し、後で DevOps ユーザーによる分析とレポートのために Amazon S3 にデータを保存します。

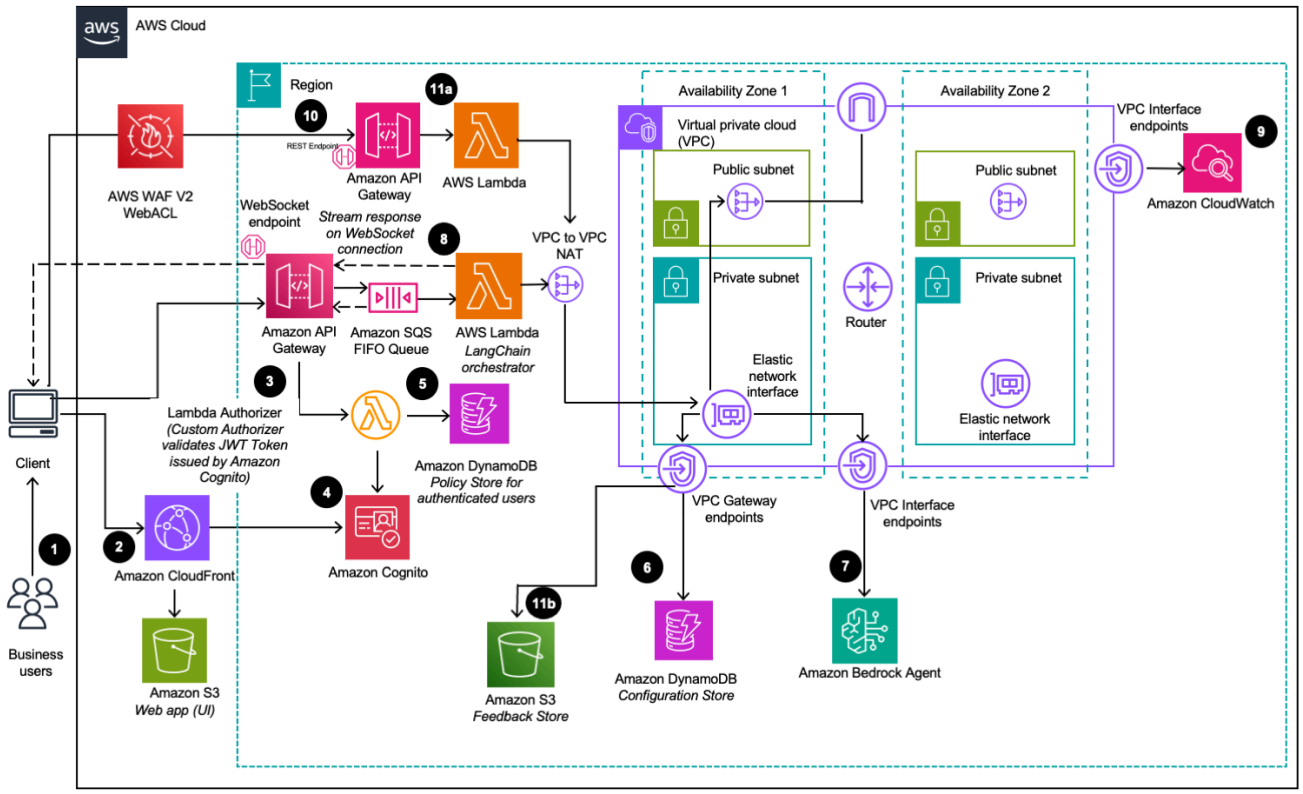
> **Note**
>
> このソリューションを Amazon VPC にデプロイする場合、データはプライベートネットワークにルーティングされます。

Bedrock エージェントユースケース

Bedrock エージェントユースケースのアーキテクチャを示しています (VPC オプションを無効にしてデプロイした場合)



Bedrock エージェントユースケースのアーキテクチャを示しています (VPC オプションを有効にしてデプロイした場合)



AWS CloudFormation テンプレートを使用してデプロイされたこのソリューションコンポーネントの大まかなプロセスフローは次のとおりです。

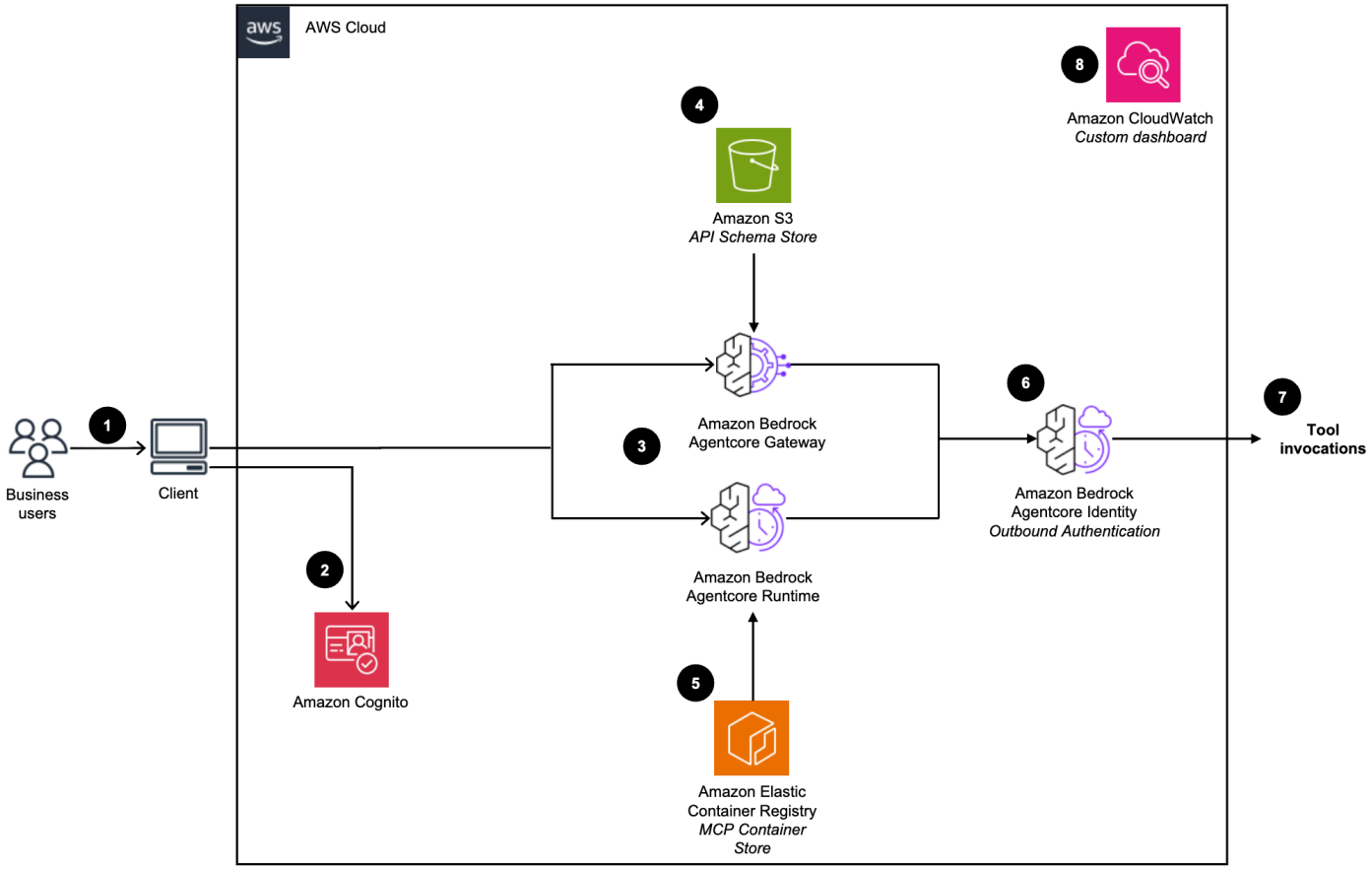
1. 管理者ユーザーは、デプロイダッシュボードを使用してユースケースをデプロイします。ビジネスユーザーは、ユースケースの UI にサインインします。
2. CloudFront は、S3 バケットでホストされているウェブ UI を提供します。
3. ウェブ UI は、API Gateway を使用して構築された WebSocket 統合を活用します。API Gateway は、認証ユーザーが属する Amazon Cognito グループに基づいて適切な AWS Identity and Access Management (IAM) ポリシーを返すカスタム Lambda オーソライザー関数によってサポートされています。ポリシーは DynamoDB に保存されます。
4. Amazon Cognito はユーザーを認証し、CloudFront ウェブ UI と API Gateway の両方をサポートします。
5. ビジネスユーザーからの受信リクエストは、API Gateway から Amazon SQS キューに渡され、AWS Lambda 関数に渡されます。キューにより、API Gateway と Lambda 統合の非同期操作が可能になります。キューは Lambda 関数に接続情報を渡し、その結果を API Gateway WebSocket 接続に直接送信して、長時間実行される推論呼び出しをサポートします。
6. AWS Lambda 関数は Amazon DynamoDB を使用して、必要に応じてユースケース設定を取得します。

7. ユーザー入力と関連するユースケース設定に基づき、AWS Lambda 関数はリクエストペイロードを作成し、設定済みの Amazon Bedrock エージェントに送信してユーザーの意図を実行します。
8. Amazon Bedrock エージェントから応答が返されると、Lambda 関数は API Gateway WebSocket を介して応答をストリーミングし、クライアントアプリケーションで使用できるようにします。
9. このソリューションは、Amazon CloudWatch を使用してさまざまなサービスから運用メトリクスを収集し、デプロイのパフォーマンスと運用状態をモニタリングできるカスタムダッシュボードを生成します。
10. フィードバック収集が有効になっている場合、Amazon API Gateway を活用した REST API エンドポイントがユーザーフィードバックの収集に使用できます。
11. フィードバックバッキング Lambda は、送信されたフィードバックを追加のユースケース固有のメタデータで補足し、後で DevOps ユーザーによる分析とレポートのために Amazon S3 にデータを保存します。

> **Note**
>
> このソリューションを Amazon VPC にデプロイする場合、データはプライベートネットワーク内でルーティングされます。

MCP サーバーのユースケース

MCP サーバーのユースケースアーキテクチャを示しています



MCP サーバーのユースケースにより、Amazon Bedrock AgentCore でのモデルコンテキストプロトコルサーバーのデプロイと管理が可能になります。MCP サーバーは、AI アプリケーションがツール、リソース、エンタープライズデータソースにアクセスするための標準化されたインターフェイスを提供します。

このソリューションは、次の 2 つのデプロイ方法をサポートします。

- ゲートウェイメソッド: 既存の Lambda 関数、REST API、または外部 MCP サーバーを MCP ツールとしてラップし、プロトコル変換を自動的に処理します。
- ランタイムメソッド: Amazon ECR イメージからカスタムコンテナ化された MCP サーバーをデプロイします。

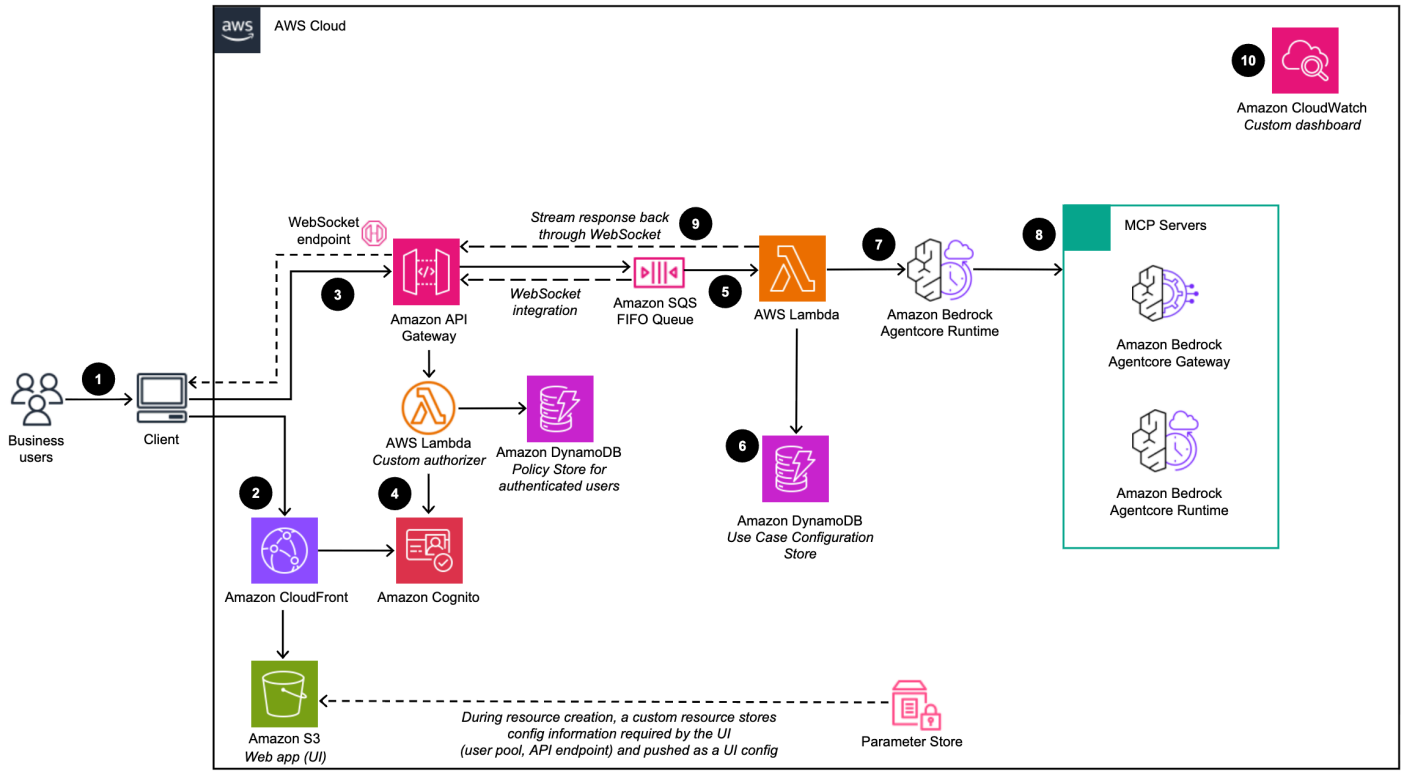
MCP サーバーデプロイの大まかなプロセスフローは次のとおりです。

1. 管理者ユーザーは、ゲートウェイまたはランタイムデプロイ方法を選択して、デプロイダッシュボードを使用して MCP サーバーのユースケースをデプロイします。

2. このアクションは Amazon Cognito で認証されます。
3. ゲートウェイデプロイの場合、ソリューションは既存の Lambda 関数、APIs、または外部 MCP サーバーを MCP 準拠のツールに変換する Amazon Bedrock AgentCore Gateway を作成します。ランタイムデプロイの場合、ソリューションは、提供された ECR イメージを使用してコンテナ化された MCP サーバーを Amazon Bedrock AgentCore Runtime にデプロイします。
4. ゲートウェイデプロイは、Amazon S3 にアップロードされた場所から必要な API/Lambda/Smithy スキーマを取得するか、または MCP サーバー URL エンドポイントに直接接続します。
5. ランタイムデプロイは、Amazon Elastic Container Registry (ECR) からユーザーが提供するコンテナ化された MCP サーバーを取得します。
6. MCP サーバーは Amazon Bedrock AgentCore Identity OAuth クライアントで計測されます。
7. MCP サーバーは、エージェントが検出できるように、関連するツールを /mcp エンドポイントで利用できるようにします。
8. Amazon CloudWatch は、モニタリングとトラブルシューティングのために MCP サーバーのデプロイから運用メトリクスとログを収集します。

エージェントビルダーのユースケース

エージェントビルダーアーキテクチャを示しています



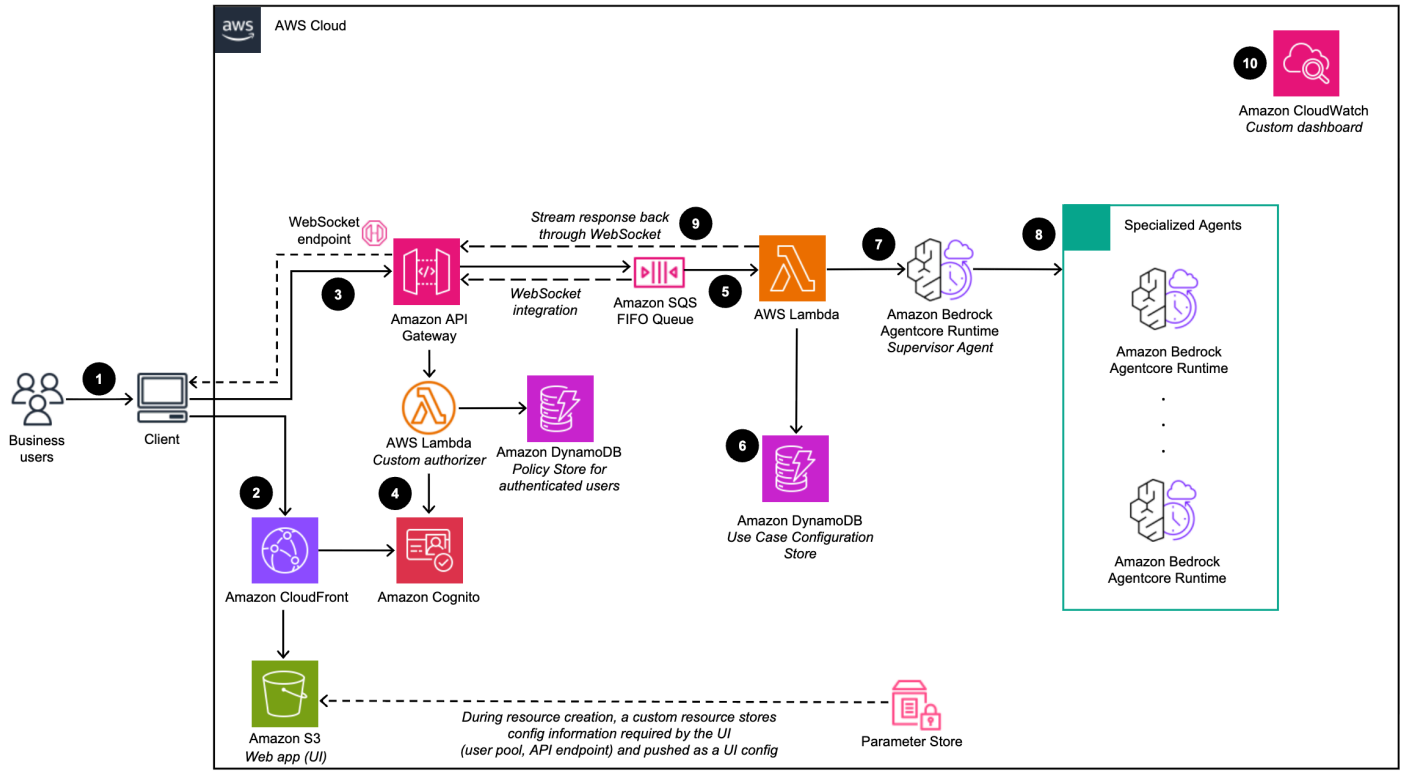
AWS CloudFormation テンプレートを使用してデプロイされたこのエージェントビルダーコンポーネントの大まかなプロセスフローは次のとおりです。

1. 管理者ユーザーは、デプロイダッシュボードを使用してユースケースをデプロイします。ビジネスユーザーは、ユースケースの UI にサインインします。
2. CloudFront は、S3 バケットでホストされているウェブ UI を提供します。
3. ウェブ UI は、API Gateway を使用して構築された WebSocket 統合を活用します。API Gateway は、認証ユーザーが属する Amazon Cognito グループに基づいて適切な AWS Identity and Access Management (IAM) ポリシーを返すカスタム Lambda オーソライザー関数によってサポートされています。ポリシーは DynamoDB に保存されます。
4. Amazon Cognito はユーザーを認証し、CloudFront ウェブ UI と API Gateway の両方をサポートします。
5. ビジネスユーザーからの受信リクエストは、API Gateway から Amazon SQS キューに渡され、AWS Lambda 関数に渡されます。キューにより、API Gateway と Lambda 統合の非同期操作が可能になります。キューは Lambda 関数に接続情報を渡し、その結果を API Gateway WebSocket 接続に直接送信して、長時間実行される推論呼び出しをサポートします。
6. AWS Lambda 関数は、DynamoDB からエージェント設定を取得します。
7. ユーザー入力と関連するユースケース設定を使用して、AWS Lambda 関数は Amazon Bedrock AgentCore Runtime で実行されるリクエストペイロードを構築してエージェントに送信します。
8. エージェントは関連付けられた MCP サーバーに接続し、ツールをストランドエージェントインスタンスに登録します。次に、エージェントはツールの説明とタスク要件に基づいてアクションを自律的に選択して実行します。
9. Amazon Bedrock AgentCore Runtime から応答が返されると、Lambda 関数は API Gateway WebSocket を介して応答をストリーミングし、クライアントアプリケーションで使用できるようにします。

> **Note**
>
> - エージェント処理は Lambda 実行タイムアウト (15 分) に制限されています。

ワークフロービルダーのユースケース

ワークフロービルダーアーキテクチャを示しています



AWS CloudFormation テンプレートを使用してデプロイされたこのワークフロービルダーコンポーネントの大まかなプロセスフローは次のとおりです。

1. 管理者ユーザーは、デプロイダッシュボードを使用してワークフローをデプロイし、専門エージェントとして含める エージェントビルダーエージェントを選択します。
2. CloudFront は、S3 バケットでホストされているウェブ UI を提供します。
3. ウェブ UI は、API Gateway を使用して構築された WebSocket 統合を活用します。API Gateway は、認証ユーザーが属する Amazon Cognito グループに基づいて適切な AWS Identity and Access Management (IAM) ポリシーを返すカスタム Lambda オーソライザー関数によってサポートされています。ポリシーは DynamoDB に保存されます。
4. Amazon Cognito はユーザーを認証し、CloudFront ウェブ UI と API Gateway の両方をサポートします。
5. ビジネスユーザーからの受信リクエストは、API Gateway から Amazon SQS キューに渡され、AWS Lambda 関数に渡されます。キューにより、API Gateway と Lambda 統合の非同期操作が可能になります。
6. AWS Lambda 関数は、特殊なエージェントビルダーのエージェントのリストなど、DynamoDB からワークフロー設定を取得します。

7. ユーザー入力とワークフロー設定を使用して、Lambda はスーパーバイザーエージェントをホストする Amazon Bedrock AgentCore Runtime にリクエストを送信します。
8. スーパーバイザーエージェントは、AgentCore Runtime 環境内のすべての特殊なエージェントビルダーエージェントのローカルインスタンスを作成します。これらの特殊なエージェントは、Agents as Tools パターンを使用してツールとして登録されます。その後、スーパーバイザーは、エージェントの説明とタスク要件に基づいて、作業を自律的に選択し、専門エージェントに委任します。
9. スーパーバイザーエージェントは、特殊なエージェントから結果を集約し、最終レスポンスを定式化し、API Gateway Websocket を介してクライアントアプリケーションにストリーミングされるように Lambda に返します。

> **Note**
>
> - ワークフロー処理は Lambda 実行タイムアウト (15 分) に制限されています。

AWS Well-Architected の設計に関する考慮事項

このソリューションは、AWS Well-Architected フレームワークのベストプラクティスに基づいて設計されました。これにより、ユーザーは信頼性が高く、安全で、効率的で、費用対効果の高いワークロードをクラウド上で設計し運用することができます。

このセクションでは、このソリューションを構築する際に AWS Well-Architected フレームワークの設計原則とベストプラクティスがどのように適用されたかを説明します。

運用上の優秀性

このセクションでは、オペレーショナルエクセレンスの柱に関する原則とベストプラクティスを用いてこのソリューションをどのように設計したかを説明します。

- Amazon CloudFormation を使用して、Infrastructure as Codeとしてソリューションを構築しました。
- Lambda 関数はカスタムメトリクスを CloudWatch とカスタムの CloudWatch ダッシュボードにプッシュして、ソリューションの状態をモニタリングします。
- ソリューションコンポーネントは高度にモジュール化されているため、デプロイするコンポーネントを柔軟に選択できます。

セキュリティ

このセクションでは、このソリューションを設計する際に、セキュリティの柱の原則とベストプラクティスをどのように適用したかについて説明します。

- デプロイダッシュボードとすべてのユースケースは Amazon Cognito で認証および承認されます。
- すべてのサービス間通信に IAM ロールを使用します。
- すべてのソリューションロールは最小特権のアクセスに従います。つまり、必要最小限の権限のみが付与されます。
- S3 バケット、DynamoDB、Amazon Kendra を含むすべてのデータストレージでは、保管時の暗号化が行われます。

信頼性

このセクションでは、信頼性の柱に関する原則とベストプラクティスを用いてこのソリューションをどのように設計したかを説明します。

- サーバーレスパラダイムに基づくアーキテクチャ。
- オンデマンドの水平方向スケーラビリティ、基盤となるインフラストラクチャの障害からの自動復旧を実現するアーキテクチャを構築しました。
- このアーキテクチャには、基盤となるエンドポイントに負荷をかけないように、リクエストのバッファリングとスロットリングが含まれています。

パフォーマンス効率

このセクションでは、パフォーマンス効率の柱に関する原則とベストプラクティスを用いてこのソリューションをどのように設計したかを説明します。

- このソリューションは、オンデマンドスケーリングが可能なフルマネージドのサーバーレス NoSQL データベースである DynamoDB を使用します。
- このソリューションでは、Amazon S3 をオブジェクトストレージとして使用し、ウェブサイトを (CloudFront を介して) ホストして、低コストでスケーラブルなイレブンナインの耐久性を実現しています。

コスト最適化

このセクションでは、このソリューションを設計する際に、コスト最適化の柱の原則とベストプラクティスをどのように適用したかを説明します。

- 可能な限りサーバーレスアーキテクチャを使用するようにソリューションが構築されているため、お支払いは使用した分のみです。

持続可能性

このセクションでは、このソリューションを設計する際に、持続可能性の柱の原則とベストプラクティスをどのように適用したかを説明します。

- このソリューションのモジュール式のコンポーネント化されたアーキテクチャにより、個々のユースケースに合わせてリソースを柔軟にプロビジョニングできます。
- このアーキテクチャはサーバーレスのコンピューティングとストレージを使用しており、リソースの利用を最適化します。
- このソリューションはクラウドベースのソリューションであるため、共有リソース、ネットワーク、電力冷却、物理設備のメリットを享受します。

アーキテクチャの詳細

このセクションでは、このソリューションを構成するコンポーネントと AWS のサービス、およびこれらのコンポーネントがどのように連携するのかについてのアーキテクチャの詳細について説明します。

このソリューションで使用している AWS のサービス

AWS のサービス	説明
Amazon API Gateway:	コア。デプロイダッシュボード用の REST API とユースケース用の WebSocket API を提供します。
AWS CloudFormation	コア。このソリューションは CloudFormation テンプレートとして配布され、CloudFormation によりソリューションの AWS リソースがデプロイされます。
Amazon CloudFront +	コア。Amazon S3 でホストされているウェブコンテンツを提供します。
Amazon Cognito	コア。API のユーザー管理と認証を行います。
Amazon DynamoDB	コア。デプロイダッシュボードのデプロイ情報と設定の詳細を保存します。Text ユースケースでは、チャット履歴と会話 ID を保存し、会話履歴とクエリの曖昧さ回避を可能にします。
AWS Lambda	コア。このソリューションでは、Lambda 関数を使用して次のことを行います。 * REST API および WebSocket API のエンドポイントをサポートする * 各ユースケースオーケストレーターのコアロジックを処理する * CloudFormation デプロイ時のカスタムリソースを実装する

AWS のサービス	説明
Amazon S3	コア。静的ウェブコンテンツをホストします。
Amazon CloudWatch	サポート。ソリューションのリソースから CloudWatch Logs にログを発行し、CloudWatch メトリクスにメトリクスを発行します。また、このデータを表示するための CloudWatch ダッシュボードも作成されます。
AWS Systems Manager	サポート。アプリケーションレベルのリソースの監視と、リソース運用およびコストデータの可視化を提供します。Parameter Store に設定データを保存するためにも使用されます。
AWS WAF	サポート。API Gateway デプロイの前に配置され、保護を提供します。
Amazon Bedrock	オプション。基盤モデルまたはカスタマイズされたモデル、Amazon Bedrock エージェント、Amazon Bedrock ナレッジベースへのアクセスに使用されます。Amazon Bedrock の統合は、データが AWS ネットワーク外に出ないようにするために推奨されます。
Amazon Bedrock AgentCore	オプション このソリューションは Amazon Bedrock AgentCore を活用し、MCP サーバー接続の実行とサポート、エージェントビルダーおよびワークフローユースケースを提供します。
Amazon Elastic Container Registry (Amazon ECR)	オプション。エージェントビルダーのデプロイの場合、ECR はエージェントコンテナイメージを保存して配布します。このソリューションは、ECR プルスルーキャッシュを使用して、GAAB チームのパブリック ECR リポジトリから構築済みのエージェントイメージを自動的に取得します。

AWS のサービス	説明
AWS Distro for OpenTelemetry (ADOT)	オプション。エージェントビルダーのデプロイの場合、ADOT はエージェントのオブザーバビリティの自動計測を提供し、エージェントオペレーションの分散トレースと構造化ログ記録を可能にします。
Amazon Kendra	オプション。Text ユースケースで、管理者ユーザーはオプションで Amazon Kendra インデックスを接続し、LLM との会話のナレッジベースとして使用できます。これにより、LLM に新しい情報を取り込み、その情報を応答で使用できるようになります。
Amazon SageMaker AI;	オプション。Amazon SageMaker AI 推論エンドポイントと統合することで、AWS アカウントとリージョン内でホストされている基盤モデルにアクセスできます。データが AWS ネットワーク外に出ないようにするには、この統合が推奨されます。 **Note** 推論エンドポイントと同じリージョンにソリューションをデプロイする必要があります。

AWS のサービス	説明
Amazon Virtual Private Cloud	オプション。VPC 対応設定でコンポーネントをデプロイするオプションが用意されています。VPC 対応設定でソリューションをデプロイする場合、ソリューションに VPC を作成させるか、ソリューションのデプロイ先と同じアカウントとリージョンにある既存の VPC を使用するか (Bring Your Own VPC) を選択できます。ソリューションが VPC を作成する場合、サブネット、セキュリティグループとそのルール、ルートテーブル、ネットワーク ACL、NAT ゲートウェイ、インターネットゲートウェイ、VPC エンドポイント、およびそのポリシーを含む必要なネットワークコンポーネントが自動的に作成されます。

デプロイダッシュボード

API Gateway カスタムオーソライザー

表面化では、API Gateway の Lambda カスタムオーソライザーは、すべての API コール (RESTful ベースと WebSocket ベースの両方) で使用され、特定のユーザーが所属グループに基づいてアクションを実行するアクセス許可を持っているかどうかを検証します。このカスタムオーソライザーは、各グループのポリシーを含む DynamoDB テーブルに基づいています。API の呼び出すと、API Gateway はカスタムオーソライザーの Lambda 関数を呼び出します。この関数は、提供された Amazon Cognito アクセストークンをデコードして、ユーザーが属するユーザーグループを判定します。次に、ポリシーテーブルにグループ名でクエリが実行され、そのグループの関連するポリシーが返されます。

新しいユースケースがデプロイされるたびに、管理ポリシーが更新され、そのユースケースの API に対する execute-api:Invoke アクションを許可する新しいステートメントが保存されます。ユースケースが削除されると、対応するステートメントがポリシーから削除されます。

個別のユースケース用に作成されたグループの場合、ポリシーには 1 つのステートメントしか存在せず、そのユースケースの API でのみ execute-api:Invoke アクションを実行できます。

この構造により、ユースケースのグループに属するすべてのユーザーがそのユースケースの API にアクセスできます。また、1 人のユーザーを手動で複数のグループに追加して、そのユーザーが複数のユースケースを使用できるようにすることもできます。

> **Warning**
>
> 既存のユーザーのグループに新しいユースケースへのアクセスを許可する場合は、ポリシーテーブル内の特定のグループのポリシーを手動で編集することもできます。ユースケースグループは、ユースケースが削除されると (手動で編集した場合でも) 削除されるため、ユースケースを削除するときは注意してください。

ユースケーススタックがデプロイダッシュボードを使用せずにスタンドアロンでデプロイされる場合、そのユースケースの API にアクセスできる単一のユーザーを含む Amazon Cognito ユーザープールがそのデプロイ用に作成されます。このユーザープールはこのユースケースにのみ属し、他のスタンドアロンのデプロイ間では共有されません。

Text ユースケース

ストリーミングのサポート

チャットアプリケーションにおいて、レイテンシーは応答性の高いユーザーエクスペリエンスを実現するための重要なメトリクスとなります。LLM の推論処理に数秒から数分かかる可能性があることから、顧客にコンテンツをどう提供するかが課題となります。このため、一部の LLM プロバイダーでは、呼び出し元への応答ストリーミングを可能にしています。推論全体が完了するのを待ってから応答を返す代わりに、トークンが利用可能になり次第返すことができます。

この機能の使用をサポートするため、Text ユースケースではチャットエクスペリエンスを支えるために WebSocket API を使用するよう設計されています。この WebSocket は API Gateway を介してデプロイされます。WebSocket API を使用すると、チャットセッションの開始時に接続を作成し、そのソケットを介して応答をストリーミングできます。これにより、フロントエンドアプリケーションのユーザーエクスペリエンスが向上します。

> **Note**
>
> モデルがストリーミングサポートを提供している場合でも、ソリューションが WebSocket API を介して応答をストリーミングできるとは限りません。ソリューションで、各モデルプ

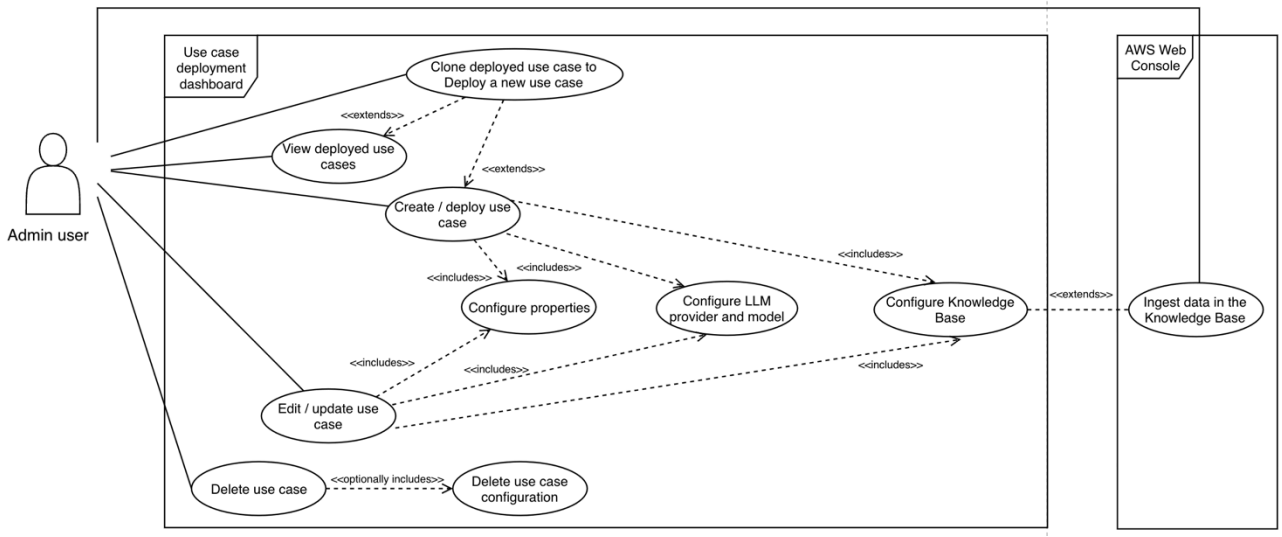
ロバイダーのストリーミングをサポートするカスタムロジックを有効にする必要があります。ストリーミングが利用可能な場合、管理者ユーザーはデプロイ時にこの機能を有効または無効にできます。

AWS での生成 AI アプリケーションビルダーソリューションの仕組み

管理者ユーザーは、主にデプロイダッシュボードを使用して新規および既存のユースケースのデプロイを表示、作成、管理します。このダッシュボードを通じて、管理者ユーザーは次のアクションを実行できます。

- デプロイのリストを表示する
- 新しいデプロイを作成する
- 既存のデプロイを編集する
- デプロイ設定のクローンを作成して新しいデプロイを作成する
- デプロイを削除する (CloudFormation 削除によりリソースをプロビジョニング解除する)
- デプロイの設定詳細を完全に削除する

デプロイダッシュボードの管理者ユーザー向けのユースケース図を示しています

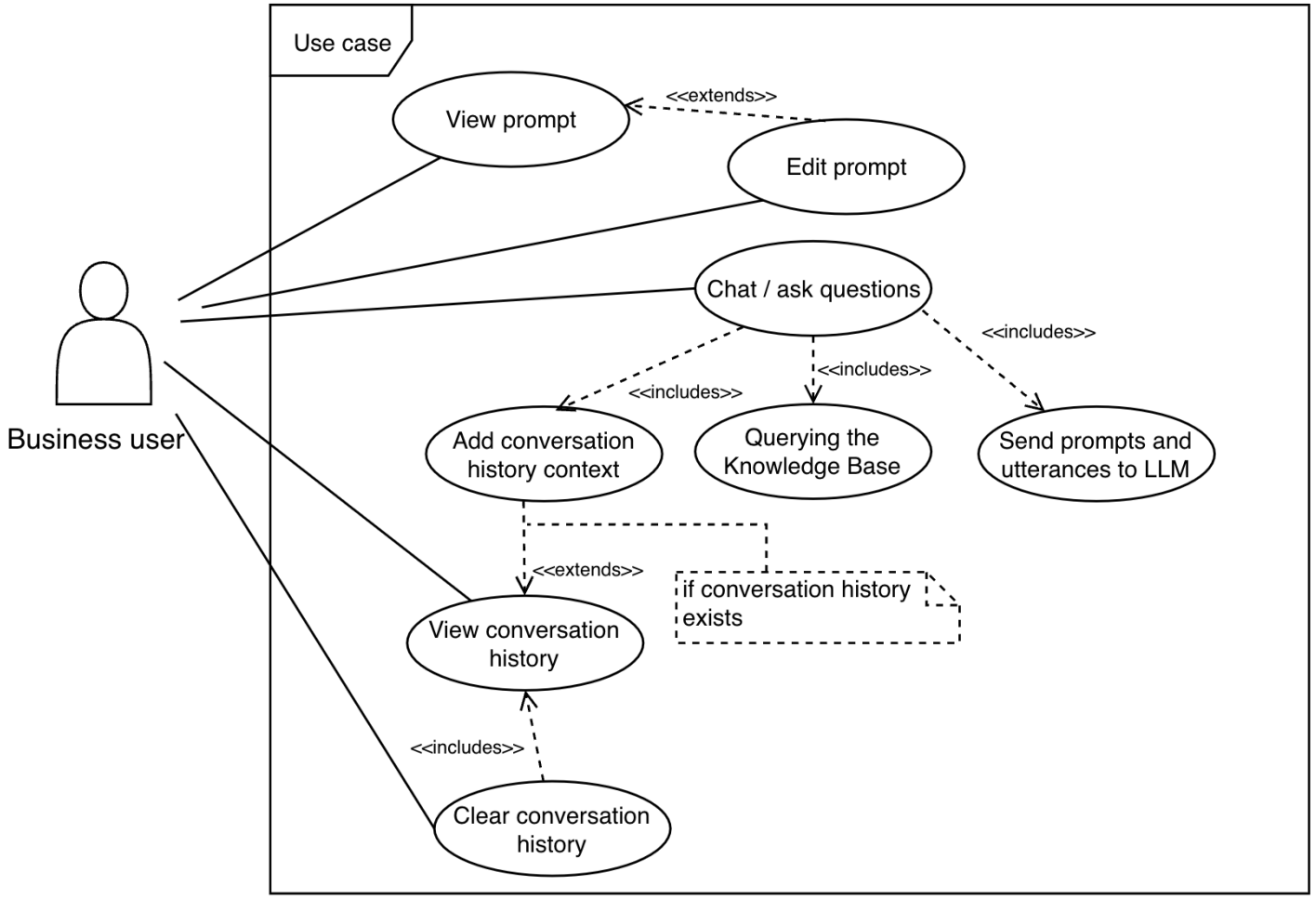


> **Note**
>
> 管理者ユーザーは、AWS コンソールに直接アクセスできない場合があります。この場合、管理者ユーザーは DevOps ユーザーと協力して、Kendra ナレッジベースへのデータの取り込みなどのアクションをサポートする必要があります。

Text ユースケースでは、ビジネスユーザーは LLM とのチャットを可能にするユーザーインターフェイスにアクセスできます。この設定の詳細は、管理者ユーザーが設定したデプロイ設定によって制御されます。Text ユースケースでは、ビジネスユーザーは次のアクションを実行できます。

- チャットインターフェイス経由でメッセージを送信する
- 会話履歴を表示する
- 会話履歴を消去する
- プロンプトを表示する
- プロンプトを編集する

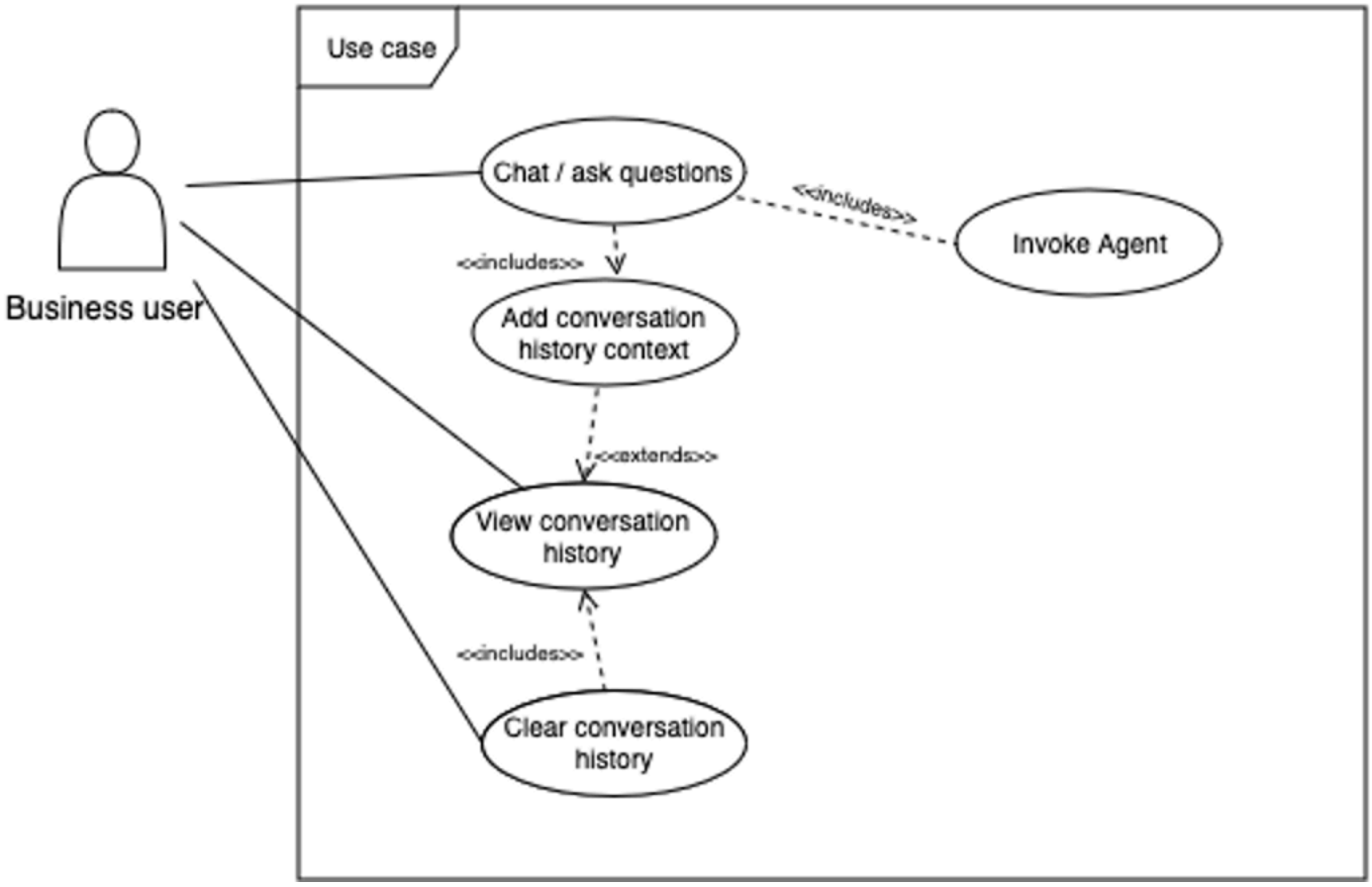
Text ユースケースのビジネスユーザー向けのユースケース図を示しています



Bedrock エージェントのユースケースでは、ビジネスユーザーは設定済みの Amazon Bedrock エージェントとチャットするための UI にアクセスできます。管理者ユーザーは、デプロイ設定でこれらの詳細を設定できます。Bedrockエージェントのユースケースでは、ビジネスユーザーは次のアクションを実行できます。

- チャットインターフェイス経由でメッセージを送信する
- 会話履歴を表示する
- 会話履歴を消去する

Bedrock エージェントのユースケースのビジネスユーザー向けのユースケース図を示しています



エージェントビルダー

エージェントビルダーは、Amazon Bedrock AgentCore で本稼働対応の AI エージェントを作成、デプロイ、管理するためのプラットフォームを提供します。このセクションでは、技術的なコンポーネントと実装の詳細について説明します。

AgentCore 統合

エージェントビルダーは、事前構築されたエージェントイメージで設定ベースのデプロイアプローチを使用して、高速で安全でスケーラブルなエージェントデプロイを可能にします。

構築済みのエージェントイメージ

エージェントコンテナイメージは、CI/CD パイプライン中に GAAB チームによって構築され、パブリック ECR リポジトリに公開されます。各イメージバージョンは、GAAB ソリューションバージョ

ン (v4.0.0 → gaab-strands-agent:v4.0.0 など) に関連付けられています。イメージは Strands SDK に基づいており、以下が含まれます。

- エージェントランタイム環境
- MCP クライアント統合
- メモリ管理機能
- OpenTelemetry 計装

ECR プルスルーキャッシュ

このソリューションでは、ECR プルスルーキャッシュを使用して、エージェントイメージをパブリック ECR リポジトリからお客様のプライベート ECR に自動的に配布します。この AWS マネージドサービスは、以下を実行します。

- 初回プル時にイメージをキャッシュ (2～5 分の遅延)
- カスタムイメージコピーロジックを不要化
- 後続デプロイ向けにローカルイメージを利用可能に
- デプロイごとに一意のキャッシュルールを作成し、競合を回避

ストレージの設定

エージェント設定は、既存のユースケース設定とともに DynamoDB に保存されます。それぞれの設定には以下が含まれます。

- システムプロンプトテンプレート
- モデルプロバイダーとモデル ID
- モデルパラメータ (温度、max_tokens)
- MCP サーバーのリファレンスとエンドポイント
- メモリ設定 (長期メモリ切り替え)
- デプロイメタデータ

イメージバージョンレジストリ

DynamoDB テーブルは、使用可能なエージェントイメージのバージョンとそのキャッシュ URI を追跡し、バージョン管理と下位互換性を可能にします。

エージェントの設定

システムプロンプト

システムプロンプトは、エージェントの動作、パーソナリティ、機能を定義します。管理者ユーザーは、次のことができます。

- エージェントビルダー UI を使用してデフォルトのテンプレートを編集する
- ツールの使用方法とレスポンスのフォーマットに関する手順を含める
- いつでもデフォルトテンプレートにリセットする

モデルの選択

エージェントビルダーは v4.0.0 で Amazon Bedrock モデルをサポートしています。

- モデルプロバイダー: Amazon Bedrock (v4.0.0 のオプションのみ)
- モデルの選択: Claude、Nova、およびその他の Bedrock モデル
- モデルパラメータ: 温度、max_tokens、top_p、モデル固有の設定

MCP サーバー統合

モデルコンテキストプロトコルサーバーは、エージェントにエンタープライズツールとデータへのアクセスを提供します。

- GET /mcp API エンドポイントによるサーバー検出
- コードを変更しない動的設定
- 認証とエンドポイント管理
- エージェントへのツール機能の露出

ストリーミングと処理

リアルタイムストリーミング

エージェントビルダーは、リアルタイムのレスポンスストリーミングに AgentCore ブリッジから WebSocket へのサーバー送信イベント (SSE) を使用します。

- Lambda 関数が AgentCore Runtime への SSE 接続を確立する

- ストリームは API Gateway WebSocket にブリッジされる
- クライアントへのトークンごとのレスポンス配信を有効にする
- 長時間実行されるリクエストの接続を維持する

処理の制約

v4.0.0 でのエージェント処理は、Lambda 実行タイムアウトに制限されています。

- 最大処理時間: 15 分
- 同期処理モデル
- 会話エージェントと中程度のワークフローに最適
- v4.1 以降で予定されている非同期サポートの拡張

メモリ管理

短期メモリ

カスタム MemoryHookProvider を使用するすべてのエージェントに対してデフォルトで有効になっています。

- Strands コールバックハンドラーを介して会話イベントをキャプチャ
- コンテキスト分離のために actorId と sessionId で整理する
- セッション内の会話コンテキストを維持する
- AgentCore Memory との自動統合

長期メモリ

strands_tools の AgentCore Memory Tool を使用するオプション機能:

- エージェントビルダー UI の簡単な切り替え
- デフォルト設定のセマンティックメモリ戦略
- 自然ツール呼び出しによるエージェント制御アクセス
- 抽出されたインサイトをセッション間で保存する
- conversationId を sessionId として使用する

オブザーバビリティ

AWS OpenTelemetry Distro (ADOT)

エージェントはコンテナビルド中に自動的に計測されます。

- エージェントオペレーションの自動トレース生成
- サービス境界間の分散トレース
- 相関 ID を使用した構造化ログ記録
- CloudWatch トランザクション検索との統合

認証フロー

ユーザーは、ユーザーグループに基づいて DynamoDB から IAM ポリシーを取得するカスタム Lambda オーソライザーによって検証された JWT トークンを使用して Amazon Cognito を介して認証します。

ワークフロービルダー

ワークフロービルダーは、エージェントをツールとして委任するパターンを使用して複数のエージェントビルダーのエージェントをオーケストレーションするスーパーバイザーエージェントを作成することで、マルチエージェントオーケストレーションを有効にします。

ワークフローアーキテクチャ

主要コンポーネント:

- スーパーバイザーエージェント: ユーザーリクエストを受け取り、専門エージェントに委任するエントリポイントエージェント
- 専門エージェント: スーパーバイザーのツールとして登録されたエージェントビルダーのユースケース
- エージェントレジストリ: エージェント設定とメタデータを保存する DynamoDB テーブル
- オーケストレーションレイヤー: Agents as Tools パターンの Strands SDK 実装

エージェントのインスタンス化

ローカルエージェントの作成

すべての専門エージェントは、同じ AgentCore Runtime 内でローカルにインスタンス化されます。

1. DynamoDB からエージェント設定を取得
2. 各エージェントビルダーのエージェントのローカルインスタンスを作成
3. 各エージェントは独自の MCP サーバー接続を維持
4. スーパーバイザーエージェントが専門エージェントをツールとして登録
5. Strands SDK がエージェントの選択と委任を管理

デプロイを計画する

このセクションでは、デプロイを計画する際のコスト、セキュリティ、リージョン、クォータの考慮事項について説明します。

> **Important**
>
> このソリューションでは、AI 生成モデルにアクセスするための主要なサービスとして、Amazon Bedrock を活用します。ソリューション内でモデルを使用できるようにするには、まずモデルへのアクセスをリクエストする必要があります。詳細については、「Amazon Bedrock ユーザーガイド」の「Model access」を参照してください。

サポートしている AWS リージョン

> **Important**
>
> このソリューションは、必要に応じて Amazon Bedrock と Amazon Kendra サービスを使用します。これは現時点では、一部の AWS リージョンでは利用できません。このソリューションは、これらのサービスが利用可能な AWS リージョンで起動する必要があります。リージョン別の AWS サービスの最新情報については、AWS リージョン別のサービスのリストを参照してください。

AWS での生成 AI アプリケーションビルダーは、以下の AWS リージョンでサポートされます。

リージョン名	
米国東部 (オハイオ)	カナダ (中部)
米国東部 (バージニア北部)	欧州 (フランクフルト)
米国西部 (北カリフォルニア)	欧州 (アイルランド)
米国西部 (オレゴン)	欧州 (ロンドン)
アジアパシフィック (ムンバイ)	欧州 (ミラノ)

リージョン名	
アジアパシフィック (ソウル)	欧州 (パリ)
アジアパシフィック (シンガポール)	欧州 (ストックホルム)
アジアパシフィック (シドニー)	中東 (バーレーン)
アジアパシフィック (東京)	南米 (サンパウロ)

> **Note**
>
> AWS 外でアクセスする基盤モデルをデプロイで使用する場合は、API が利用可能なリージョンについてモデルプロバイダーに確認してください。プロバイダーの API が特定のリージョンでしか利用できない場合、高レイテンシーやタイムアウトなどの不安定性が生じる可能性があります。組織の法務チームやコンプライアンスチームに確認して、リージョンの境界を越えるデータに関する考慮事項を評価することも重要です。

Cost

この AWS ソリューションでは、使用したリソースに対してのみ課金され、最低料金やセットアップ料金は発生しません。ユーザーには、生成 AI のユースケースを起動するために使用するダッシュボードと、デプロイされるすべてのユースケースに対して課金されます。デプロイされるユースケースのコストは、設定によって異なります。設定例:

1. シンプルなデプロイダッシュボードは、1 か月あたり約 20 USD です。
2. シンプルな本番対応のチャットボットのユースケースをデフォルト設定で米国東部 (バージニア北部) にデプロイする場合、Amazon Bedrock を利用し、ドキュメントにはアクセスしないと、1 か月あたり約 200 USD になります。
3. Amazon VPC ユースケースのスケールしたシステムの場合、数万のドキュメントに対して 1 日あたり 8,000 件のクエリをサポートし、コストは 1 か月あたり約 1,500 USD です。ユースケースのコストは、さまざまなモデルプロバイダーの Text ユースケース、検索拡張生成 (RAG) を有効にするかなど、設定によって異なります。

ワークロードの説明	推定コスト (USD/月)
デプロイダッシュボードのコスト例	20 USD/ 月
テキストベースの概念実証のコスト例 (デプロイダッシュボードと単一の Text ユースケース、1 日あたり最大 100 回のインタラクションを含む)	40 USD/ 月
高度にスケーラブルな生成 AI クエリエンジンのコスト例 (デプロイダッシュボード、単一の Text ユースケース、最大 10 万ドキュメントの RAG 用の Amazon Kendra インデックス、1 日あたり最大 8,000 件のクエリ、VPC を有効化)	1,500 USD/ 月
エージェントベースの概念実証のコスト例 (デプロイダッシュボード、Amazon Bedrock ナレッジベースと Amazon Bedrock ガードレールが有効になっている 1 つの Bedrock エージェントユースケース、1 日あたり最大 100 件のインタラクションを含む)	840 USD/ 月
MCP サーバーのコスト例 (デプロイダッシュボード、Lambda 統合用のゲートウェイメソッドを使用した 1 つの MCP サーバーのユースケース、1 日あたり最大 100 件のツール呼び出しを含む)	22 USD/月
エージェントビルダーのコスト例 (デプロイダッシュボード、MCP 統合と長期メモリが有効になっている 1 つのエージェントビルダーのユースケース、1 日あたり最大 100 件のインタラクションを含む)	55 USD/ 月

ワークロードの説明	推定コスト (USD/月)
ワークフロービルダーのコスト例 (デプロイダッシュボード、3 つのエージェントビルダーのエージェントを含む 1 つのワークフロー、1 日あたり最大 100 回のインタラクションを含む)	109 USD/ 月

> **Important**
>
> これらの例は、特定のワークロードのコストを見積もるサポートの目的でのみ提供されています。使用する LLM、設定、または AWS のサービスが異なると、コストが変わる場合があります (サーバーレス/オンデマンド課金と比べたプロビジョン済み/時間課金など)。コスト管理には、AWS Cost Explorer を使用して予算を策定することをお勧めします。価格は変更されることがあります。詳細については、このソリューションで使用する AWS のサービスごとに料金ウェブページを参照してください。

デプロイダッシュボードを実行する場合のコスト例

次の表は、米国東部 (バージニア北部) リージョンの 100 アクティブユーザーで、デフォルトパラメータを含むデプロイダッシュボードを使用した場合の 1 か月間のコスト (1 か月あたり約 20 USD) の内訳を示しています。

AWS のサービス	ディメンション	コスト [USD]
API Gateway、DynamoDB、CloudFront、Amazon S3、Lambda、Systems Manager Parameter Store	キャッシュを有効にしない場合の 1 か月あたり 5,000 回の 512 KB の REST API コール	1.97 USD
Amazon Cognito	高度なセキュリティ機能を有効にし、SAML または OIDC フェデレーションを介してサインインするユーザーなし、1	5.55 USD

AWS のサービス	ディメンション	コスト [USD]
	か月あたり 100 人のアクティブユーザー	
AWS WAF	1 つのウェブ ACL と 7 つの定義済みルールにわたる 10,000 件のウェブリクエスト、ルールグループなし	12.60 USD
デプロイダッシュボードの合計コスト		20.12 USD

テキストベースの概念実証のコスト例

デプロイダッシュボードでは、一度に多くのユースケースをデプロイできます。次の表は、1 日あたり 100 件のクエリを LLM で実行する 1 人のビジネスユーザーに対して、RAG なしでデプロイされたユースケースのコスト内訳を説明しています。クエリは WebSocket でテキストメッセージとして送信され、ストリーミングが有効になっていることを前提に、応答はトークンとしてストリーミングで返されます。Amazon Bedrock Nova Pro モデルを使用すると、このユースケースを実行するコストは約 20 USD/月です。

AWS のサービス	ディメンション	コスト [USD]
API Gateway (WebSocket)、CloudFront、Lambda、Amazon S3、AWS Systems Manager Parameter Store	1 日あたり 100 件のチャットインタラクション。平均メッセージサイズは、メッセージあたり 32 KB、各接続は 5 分。	0.61 USD
CloudWatch	テスト用に冗長モードをオンにした状態で 1.5 GB の CloudWatch ログ	7.23 USD
Amazon DynamoDB	会話履歴テーブル、1 GB のストレージ	3.05 USD

AWS のサービス	ディメンション	コスト [USD]
	LLM 設定テーブル、1 GB のストレージ	
ユースケースコストの小計 (LLM を除く)		10.89 USD
Amazon Bedrock (Nova Pro)	1 日あたり 100 件のインタラクションの前提: * 1 日あたり 190,000 の入力トークンの月別コスト = 0.152 USD × 30 日 * 1 日あたり 16,000 の出力トークンの月別コスト = 0.0512 USD × 30 日	6.10 USD
Amazon Bedrock (Nova Pro) の合計アプリケーションコスト	10.89 USD (ユースケースのコスト) + 6.10 USD (Amazon Bedrock のコスト)	17.00 USD

> **Note**
>
> AWS ネットワーク外のサービスに対して行われた推論呼び出しのコストは、これらの見積もりに含まれていません。AWS モデルプロバイダーを使用しない場合は、LLM プロバイダーの料金ガイドを参照してください。
> AWS サービスの料金ガイドは、「Amazon Bedrock の料金」と「Amazon SageMaker AI の料金」で確認できます。

高度にスケーラブルな生成 AI クエリエンジンのコスト例

次の表は、Amazon Bedrock の Nova Pro モデルを LLM とする RAG 対応ユースケースのコスト内訳を示しています。Bedrock ナレッジベースを追加すると、このユースケースのコストは約 1,300 USD/月になります

AWS のサービス	ディメンション	コスト [USD]
API Gateway (WebSocket)	1 日あたり 8000 件のチャットインタラクション。平均メッセージサイズは、メッセージあたり 32 KB、各接続は 5 分。	38.89 USD
CloudFront	1 か月あたり 240,000 件のリクエスト、100 GB のデータをインターネットに転送し、1 GB のデータをオリジンに転送する場合	8.76 USD
Amazon Bedrock (Nova Pro)	前提: 入力トークン = promptTemplate (400) + context (400) + chatHistory (1,080) + クエリ入力トークン (20) = 1,900 出力トークン = 160 (平均) 1 日あたり 8,000 件のトランザクションの場合、 日次入力トークンコスト (1,900 x 8,000 = 15,200,000 トークン x トークンあたりの料金 0.0008/1000) 日次出力トークンのコスト (160 x 8,000 = 1,280,000 トークン x トークンあたりの料金 0.0032/1000) 月別コスト ((12.16 USD + 4.10 USD) x 30)	487.80 USD

AWS のサービス	ディメンション	コスト [USD]
CloudWatch	ログに取り込んだ 5 GB のデータと 1 つのダッシュボードを使用する 24 のメトリクス	9.72 USD
DynamoDB	会話履歴を追跡するための DynamoDB テーブル、各レコードで最大 1 KB のデータ、1 日あたり 8,000 回の読み取りと書き込み	11.70 USD
Lambda	コンテナサイズ - 128 MB、512 MB のエフェメラルストレージ、 認証に使用する 2 つの Lambda 関数 コンテナサイズ - 256 MB、512 MB のエフェメラルストレージ、1 秒あたり 5 件のリクエスト、平均コンピューティング時間 20 秒	20.89 USD
ユースケースのコスト合計		577.76 USD/月 + ナレッジベースコスト (以下を参照)

> **Note**
>
> AWS ネットワーク外のサービスに対して行われる API コールのコストは、これらの見積もりに含まれていません。Amazon Bedrock を使用しない場合は、LLM プロバイダーの料金ガイドを参照してください。

ナレッジベースを追加する場合のコスト

ナレッジベースのコストは、使用するナレッジベースのタイプと、ナレッジベースで使用される基盤ベクトルストア (Bedrock の場合) によって異なります。ナレッジベースのプロビジョンと管理は、このソリューションの範囲に含まれていません。

Amazon Bedrock ナレッジベース

このソリューションでは、Amazon Bedrock ナレッジベースに関連するリソースを管理またはプロビジョンは行いません。Amazon Bedrock を使用する場合、ナレッジベース機能自体の使用にはコストは発生しません。ただし、ユースケースが各クエリで使用する埋め込みモデルの使用に対して料金が発生します。さらに、ナレッジベースの基盤ベクトルストアでは (Amazon OpenSearch Service のインデックスや Amazon Relational Database Service 内のデータベースなど) に、ここで提供したり計算したりできない関連コストが発生します。

上記の高度にスケーラブルな生成 AI クエリエンジンのシナリオの場合、Amazon Bedrock 埋め込みモデルを呼び出すためにこのサービスで発生するコストは次のとおりです。

AWS のサービス	ディメンション	コスト [USD]
Amazon Bedrock (Amazon Titan Text Embeddings V2)	1 クエリあたり 1,900 入力トークンで、1 日あたり 8,000 件のクエリ = 15,200,000 トークン = 1 日あたり 0.30 USD 日別コスト x 30 日 = 9.00 USD の月額コスト	9.00 USD
Amazon OpenSearch Service (Serverless) の使用例	4 つの OpenSearch Compute Unit (OCU) を使用する基本的なサーバーレス設定 (最低料金) = 1 日あたり 23.04 USD 日別コスト x 30 日 = 691.20 USD **Note** これは概算値であり、ワークロードによって	691.20 USD

AWS のサービス	ディメンション	コスト [USD]
	はさらに多くの OCU が必要になります。既にプロビジョン済みの OpenSearch リソースを使用する場合のコストはこれより低くなります。	
追加コストの合計		700.20 USD

Amazon Kendra

このソリューションでは、Kendra インデックスを自動的にプロビジョンすることも、ユーザー独自のインデックスを使用することもできます。上記の高度にスケーラブルな生成 AI クエリエンジンに適した設定を実行する場合のコストは次のとおりです。

AWS のサービス	ディメンション	コスト [USD]
Amazon Kendra	Amazon Kendra Enterprise Edition と 0～50 のデータソース、1 日あたり 0～8,000 件のクエリ、最大 100,000 件のドキュメント	1,008.00 USD

> **Note**
>
> Amazon Kendra インデックスはユースケース間で共有できます。ただしこれにより、インデックスあたりのクエリ数が増加する可能性があります。これが Amazon Kendra Enterprise Edition の範囲外となる場合は、追加料金が適用されます。

ユースケースで Amazon VPC を有効にする場合の追加コスト

次の表は、2 つの AZ にデプロイされたユースケースで Amazon VPC を有効にする場合のコスト内訳を示しています。

AWS のサービス	ディメンション	コスト [USD]
Amazon NAT Gateway	前提条件: 2 つの AZ にデプロイ、各 AZ に 1 つの NAT ゲートウェイ。NAT ゲートウェイを介して 100 GB のデータ処理を 730 時間、1 か月あたり 100 GB のデータ処理	74.70 USD
AWS PrivateLink (VPC エンドポイント)	前提条件: 2 つの AZ にデプロイ、各 AZ に 1 つのプライベートサブネット、1 つの VPC エンドポイント、2 つの Elastic Network Interface (ENI)。 6 つの VPC エンドポイント、VPC エンドポイントあたり 2 つの ENI、1 か月で 730 時間、1,024 GB のデータを処理	97.84 USD
パブリック IPv4 アドレス	前提: 2 つの AZ にデプロイ、各 AZ に 1 つのパブリックサブネット、各パブリックサブネットに 1 つの NAT ゲートウェイ。各 NAT ゲートウェイには 1 つのアクティブなパブリック IPv4 が設定されている。 2 つのアクティブなパブリック IPv4 アドレス x 730 時間 /	7.30 USD

AWS のサービス	ディメンション	コスト [USD]
	月 x 0.005 USD / 時間 = 7.3 USD	
追加料金 (Amazon VPC の場合)		179.93 USD

プロビジョンドスループットを使用する場合のコストへの影響

プロビジョンドスループットのコストは、プロビジョンしたモデルのタイプと契約期間、契約期間に選択されたモデルユニットによって異なります。プロビジョンドスループットの使用には追加コストがかかります。

詳細と最新の料金については、「Bedrock の料金」を参照してください。

クロスリージョン推論の使用コスト

クロスリージョン推論を使用する場合、追加のルーティングやデータ転送についての料金は発生しません。モデルについては、ソースまたはプライマリリージョンと同じ料金がトークンごとに課金されます。

エージェントベースの概念実証のコスト例

Amazon Bedrock エージェントを使用すると、使用するモデルやナレッジベース (RAG が有効になっている場合) など、エージェントを構成するコンポーネントと追加した追加機能に基づいて料金が発生します。次の表は、オンデマンド Claude 3.5 Sonnet モデル、Amazon Bedrock ナレッジベース、Amazon Bedrock ガードレールで設定した Bedrock エージェントユースケースのコスト内訳を説明しています。

Amazon Bedrock ナレッジベースを追加するコストと同様に、このソリューションでは Amazon Bedrock エージェントに関連するリソースの管理やプロビジョニングは行いません。このソリューションでは Amazon Bedrock ナレッジベースの使用にコストは発生しないとはいえ、以下のコストも発生します。

- 送信されるクエリごとの埋め込みモデルの使用コスト
- ナレッジベースで使用するベクトルストア (Amazon OpenSearch Service のインデックス、Amazon RDS 内のデータベースなど) のコスト

次の表では、クエリごとに 1,900 の入力トークンと 160 の出力トークンを使用して、1 日あたり 100 件のインタラクションがあることを想定しています。

> **Note**
>
> この Bedrock エージェントユースケース例では、外部 API を使用するように設定されたアクショングループがある場合には、これらのコストが追加されます。これらのコストは、この表の計算の範囲外です。

AWS のサービス	ディメンション	コスト [USD]
API Gateway (WebSocket)、CloudFront、Lambda、Amazon S3、Systems Manager Parameter Store	1 日あたり 100 チャットインタラクション、1 メッセージにつき平均メッセージサイズは 32 KB、1 接続につき 5 分。	0.61 USD
CloudWatch	テスト用に冗長モードをオンにした状態で 1.5 GB の CloudWatch ログ	7.23 USD
DynamoDB	1 KB のレコードサイズ用の LLM 設定テーブルと 1 GB ストレージ	0.25 USD
コストの小計 (LLM を除く)		8.09 USD
Anthropic Claude 3.5 Sonnet	* 1 日あたり 190,000 の入力トークンの日別コスト (0.003/1,000 トークン) = 0.57 USD + 日別コスト × 30 日 = 17.10 USD * 1 日あたり 16,000 の出力トークンの日別コスト (0.015/1,000 トークン) = 0.24 USD +	24.30 USD

AWS のサービス	ディメンション	コスト [USD]
	日別コスト × 30 日 = 7.20 USD	
Amazon Bedrock ナレッジベース用の Amazon Bedrock (Amazon Titan Text Embeddings V2)	1 日あたり 190,000 の入力トークンの日別コスト (0.00002/1000 トークン) = 0.004 日別コスト × 30 日 = 0.12 USD	0.12 USD
Amazon OpenSearch Service (Serverless) の使用例	4 つの OpenSearch Compute Unit (OCU) を搭載する基本的なサーバーレス構成 (最低請求額) = 1 日あたり 23.04 USD 日別コスト × 30 日 = 691.20 USD	691.20 USD

AWS のサービス	ディメンション	コスト [USD]
Amazon Bedrock ガードレール	190K トークンは、760,000 (190,000 × 4) 文字と 3,800 テキスト単位 (760K 文字/200) とほぼ同等です。 コンテンツフィルター、個人を特定できる情報 (PII) フィルター、機密情報フィルター (正規表現)、単語フィルターで設定されたガードレールの場合を考えてみます。 1 日のコンテンツフィルターのコスト (0.75/1000 テキストユニット) + PII フィルターのコスト (0.1 USD/1,000 テキストユニット) + 機密情報フィルター (正規表現) + ワードフィルター = 2.85 USD + 0.38 USD + 0 USD + 0 USD 月別コスト = 日別コスト × 30 日 = 96.90 USD	96.90 USD
Anthropic Claude 3.5 Sonnet でサポートされるエージェントのアプリケーションコスト合計	8.09 USD (ユースケースコスト) + 812.52 USD (その他のエージェント設定)	820.61 USD

> **Note**
>
> AWS モデルプロバイダーを使用しない場合は、LLM プロバイダーの料金ガイドを参照してください。AWS サービスの料金ガイドは、「Amazon Bedrock の料金」と「Amazon SageMaker AI の料金」で確認できます。

MCP サーバーのコスト例

MCP サーバーユースケースにより、Amazon Bedrock AgentCore でのモデルコンテキストプロトコルサーバーのデプロイと管理が可能になります。次の表は、ゲートウェイメソッドを使用して既存の Lambda 関数をラップする MCP サーバーのユースケースのコスト内訳を示しています。

このソリューションは、AgentCore Gateway のデプロイと設定を管理します。以下の料金が請求されます。

- インフラストラクチャコスト (API Gateway、Lambda、DynamoDB、CloudWatch、S3)
- AgentCore Gateway の消費量 (ツール呼び出しごと)
- Lambda 関数の実行コスト (Lambda ターゲットを持つゲートウェイメソッドの場合)
- 外部 API コスト (該当する場合、API または MCP サーバーターゲットを使用するゲートウェイメソッドの場合)

Item	計算	Cost
Amazon API Gateway (REST API)	1 日あたり 100 回のツール呼び出し × 30 日 = 1 か月あたり 3,000 リクエスト	0.05 USD
AWS Lambda (オーケストレーション)	1 日あたり 100 回の呼び出し × 30 日 × 1 秒平均 × 512 MB = 1 か月あたり 3,000 GB-秒	0.05 USD
Amazon DynamoDB	1 か月あたり 3,000 回の読み取り/書き込みリクエスト + 1 GB ストレージ	0.15 USD
Amazon CloudWatch	3,000 回の呼び出しに対する標準モニタリングとログ記録	1.00 USD
Amazon S3	設定ストレージとログ (最小使用量)	0.25 USD
Amazon Bedrock AgentCore Gateway	1 か月あたり 3,000 ツール呼び出し	0.05 USD

Item	計算	Cost
ターゲット Lambda 関数	1 日あたり 100 回の呼び出し × 30 日 × 0.5 秒 × 128 MB = 1 か月あたり 1,500 GB-秒	0.25 USD
合計月額コスト	1.75 USD (インフラストラクチャ) + 0.05 USD (AgentCore Gateway)	1.80 USD

> **Note**
>
> コストは、デプロイ方法 (ゲートウェイとランタイム)、ターゲットタイプ、使用パターンによって異なります。ランタイムメソッドのデプロイでは、ゲートウェイ料金ではなく AgentCore Runtime 料金が発生します。外部 API コストとカスタムコンテナホスティングコストは別途発生します。

エージェントビルダーのコスト例

エージェントビルダーを使用すると、Amazon Bedrock AgentCore にカスタムエージェントを作成してデプロイできます。次の表は、Claude 3.5 Sonnet、MCP サーバー統合、長期メモリが有効になっているエージェントビルダーのユースケースのコスト内訳を示しています。

このソリューションは、AgentCore Runtime のデプロイと設定を管理します。以下の料金が請求されます。

- インフラストラクチャコスト (API Gateway、Lambda、DynamoDB、CloudWatch、S3)
- AgentCore Runtime 消費量 (実際のエージェント実行時間に基づく CPU およびメモリ時間)
- 基盤モデル推論 (入力トークンと出力トークン)
- AgentCore Memory (短期イベントと長期保存/取得)

次の表では、1 日あたり 100 件のインタラクション、クエリごとに 1,900 の入力トークンと 160 の出力トークン、インタラクションあたりの平均エージェント実行時間が 5 秒であると想定しています。

AWS のサービス	ディメンション	コスト [USD]
API Gateway (WebSocket)、CloudFront、Lambda、Amazon S3、Systems Manager Parameter Store	1 日あたり 100 チャットインタラクション、1 メッセージにつき平均メッセージサイズは 32 KB、1 接続につき 5 分。	0.61 USD
CloudWatch	テスト用に冗長モードをオンにした状態で 1.5 GB の CloudWatch ログ	7.23 USD
DynamoDB	1 KB のレコードサイズ用の LLM 設定テーブルと 1 GB ストレージ	0.25 USD
インフラストラクチャコストの小計		8.09 USD
Amazon Bedrock AgentCore Runtime	* CPU: 1 vCPU × 5 秒 × 100 インタラクション = 125 vCPU-秒/日 = 0.140 vCPU-時間/日 + 日別コスト: 0.140 × 0.0895 USD = 0.013 USD + 月別コスト: 0.013 × 30 = 0.38 USD * メモリ: 512 MB (0.5 GB) × 5 秒 × 100 インタラクション = 250 GB-秒/日 = 0.069 GB-時間/日 + 日別コスト: 0.069 × 0.00945 USD = 0.0007 USD + 月別コスト: 0.0007 × 30 = 0.02 USD	0.40 USD
Anthropic Claude 3.5 Sonnet	* 1 日あたり 190,000 入力トークンの日別コスト (0.003/1,000 トークン) = 0.57 USD +	24.30 USD

AWS のサービス	ディメンション	コスト [USD]
	日別コスト × 30 日 = 17.10 USD * 1 日あたり 16,000 の出力トークンの日別コスト (0.015/1,000 トークン) = 0.24 USD + 日別コスト × 30 日 = 7.20 USD	
Amazon Bedrock AgentCore Memory	* 短期メモリ: 100 件の新規イベント/日 × 0.25 USD/1,000 件のイベント = 0.025 USD/日 + 月別コスト: 0.025 USD × 30 = 0.75 USD * 長期メモリストレージ (組み込み戦略): 100 レコード × 0.75 USD/1,000 レコード/月 = 0.075 USD/月 * 長期メモリ取得: 100 取得/日 × 0.50 USD/1,000 取得 = 0.05 USD/日 + 月別コスト: 0.05 USD × 30 = 1.50 USD	2.33 USD
Claude 3.5 Sonnet を使用したエージェントビルダーの合計アプリケーションコスト	8.09 USD (インフラストラクチャ) + 0.40 USD (AgentCore Runtime) + 24.30 USD (モデル) + 2.33 USD (メモリ)	35.12 USD

> **Note**
>
> AgentCore Runtime の料金は消費ベースです。実際のコストは以下によって異なります。
>
> - エージェントの実行時間 (アクティブな処理中の CPU とメモリの使用量)
> - インタラクションの数とその複雑さ

- MCP ツールの使用状況 (ツール実行用の追加の CPU/メモリ)
- メモリ設定 (短期メモリと長期メモリの有効化)

AgentCore の料金の詳細については、「Amazon Bedrock の料金」を参照してください。

> **Note**
>
> 外部 API またはサービスを呼び出す MCP サーバーを使用する場合、それらのコストは追加となり、この計算の範囲外です。同様に、AgentCore Browser または Code Interpreter ツールを使用する場合、消費ベースの料金は vCPU 時間あたり 0.0895 USD、GB 時間あたり 0.00945 USD の料金が適用されます。

ワークフロービルダーのコスト例

ワークフロービルダーは、複数のエージェントビルダーのエージェントをオーケストレーションするスーパーバイザーエージェントを作成します。次の表は、1 つのスーパーバイザーエージェントと 3 つの専門エージェントビルダーエージェントを含むワークフローのコスト内訳を示しています。すべて Claude 3.5 Sonnet で設定され、長期メモリが有効になっています。

前提: 1 日あたり 100 回のインタラクション、インタラクションごとに平均 2 回のエージェント委任、エージェントあたり 5 秒の実行時間。

AWS のサービス	ディメンション	コスト [USD]
API Gateway (WebSocket)、CloudFront、Lambda、Amazon S3、Systems Manager Parameter Store	1 日あたり 100 チャットインタラクション、1 メッセージにつき平均メッセージサイズは 32 KB、1 接続につき 5 分。	0.61 USD
CloudWatch	テスト用に冗長モードをオンにした状態で 1.5 GB の CloudWatch ログ	7.23 USD

AWS のサービス	ディメンション	コスト [USD]
DynamoDB	1 KB のレコードサイズ用の LLM 設定テーブルと 1 GB ストレージ	0.25 USD
インフラストラクチャコストの小計		8.09 USD
Amazon Bedrock AgentCore Runtime (スーパーバイザーエージェント)	* CPU: 1 vCPU × 5 秒 × 100 インタラクション = 0.140 vCPU-時間/日 × 30 = 0.38 USD * メモリ: 0.5 GB × 5 秒 × 100 インタラクション = 0.069 GB-時間/日 × 30 = 0.02 USD	0.40 USD
Amazon Bedrock AgentCore Runtime (3 つの専門エージェント)	* インタラクションあたり平均 2 回の委任 = 200 エージェント実行/日 * CPU: 1 vCPU × 5 秒 × 200 = 0.278 vCPU-時間/日 × 30 = 0.75 USD * メモリ: 0.5 GB × 5 秒 × 200 = 0.139 GB-時間/日 × 30 = 0.04 USD	0.79 USD
Anthropic Claude 3.5 Sonnet (スーパーバイザーエージェント)	* 入力: 190,000 トークン/日 × 0.003 USD/1,000 = 0.57 USD/日 × 30 = 17.10 USD * 出力: 16,000 トークン/日 × 0.015 USD/1,000 = 0.24 USD/日 × 30 = 7.20 USD	24.30 USD

AWS のサービス	ディメンション	コスト [USD]
Anthropic Claude 3.5 Sonnet (専門エージェント)	* インタラクションごとに平均 2 回の委任 * 入力: 380,000 トークン/日 × 0.003/1,000 USD = 1.14 USD/日 × 30 = 34.20 USD * 出力: 32,000 トークン/日 × 0.015/1,000 = 0.48 USD/日 × 30 = 14.40 USD	48.60 USD
Amazon Bedrock AgentCore Memory (スーパーバイザーエージェント)	* 短期: 100 イベント/日 × 0.25 USD/1,000 × 30 = 0.75 USD * 長期ストレージ: 100 レコード × 0.75 USD/1,000 = 0.08 USD * 長期取得: 100 取得/日 × 0.50 USD/1,000 × 30 = 1.50 USD	2.33 USD
Amazon Bedrock AgentCore Memory (専門エージェント)	* 短期: 200 イベント/日 × 0.25 USD/1,000 × 30 = 1.50 USD * 長期ストレージ: 200 レコード × 0.75 USD/1,000 = 0.15 USD * 長期取得: 200 取得/日 × 0.50 USD/1,000 × 30 = 3.00 USD	4.65 USD
ワークフロービルダー (3 つのエージェント) の合計アプリケーションコスト	8.09 USD (インフラストラクチャ) + 1.19 USD (AgentCore Runtime) + 72.90 USD (モデル) + 6.98 USD (メモリ)	89.16 USD

Note

- 委任レートが高くなるとトークンの消費量が比例的に増加します

AgentCore の料金の詳細については、「Amazon Bedrock の料金」を参照してください。

セキュリティ

AWS インフラストラクチャでシステムを構築すると、お客様と AWS の間でセキュリティ上の責任が分担されます。この責任共有モデルにより、AWS が、ホストオペレーティングシステムと仮想化レイヤーからサービスが運用されている施設の物理的なセキュリティに至るまでの要素を運用、管理、および制御するため、お客様の運用上の負担を軽減するのに役立ちます。AWS セキュリティの詳細については、「AWS クラウドセキュリティ」を参照してください。

Amazon Bedrock で基盤モデルを使用する

Amazon Bedrockは、Amazon Nova モデルから他の主要な基盤モデル (FM) まで、幅広いモデルコレクションをホストしています。Amazon Bedrock を使用する場合、すべてのモデルが AWS インフラストラクチャ内でホストされます。つまり、Amazon Bedrock を LLM プロバイダーとして使用する場合、すべての推論リクエストは AWS ネットワーク内に残り、ネットワークトラフィックがリージョン外に出ることはありません。

> **Note**
>
> Amazon Bedrock で利用できるすべての基盤モデル (FM) は、AWS が管理、所有する AWS インフラストラクチャ上で直接ホストされます。モデルプロバイダーは、プロンプトやそれに対する応答などの顧客データや Amazon Bedrock サービスログにアクセスすることはできません。Amazon Bedrock のセキュリティ体制に関する詳細については、「Amazon Bedrock ユーザーガイド」の「Data protection in Amazon Bedrock」を参照してください。

IAM ロール

IAM ロールを使用すると、AWS クラウドのサービスとユーザーに、きめ細かなアクセスポリシーとアクセス許可を割り当てることができます。このソリューションでは、リージョンのリソースを作成するためのアクセス権をソリューションの Lambda 関数に付与する IAM ロールが作成されます。

CloudWatch ログ

デプロイダッシュボードのモデル選択ページの [追加設定] を使用して、ユースケースをデプロイするときに詳細モードを有効にできます。詳細モードでは、デバッグや迅速な実験に役立つ詳細な CloudWatch ログが有効になります。

> **Note**
>
> 詳細モードを有効にすると、ナレッジベースから取得したドキュメント (RAG が有効になっている場合) とプロンプトもログに記録されます。これには機密情報が含まれる場合があります。

VPC

このソリューションは、Amazon VPC 設定に関して 2 つのオプションが提供されています。

1. ソリューションに Amazon VPC を構築させる
2. ソリューション内で使用するために BYO Amazon VPC (独自の Amazon VPC) を使用して管理する。

ソリューションに Amazon VPC を構築させる

ソリューションに Amazon VPC を構築させるオプションを選択すると、デフォルトでは、10.10.0.0/20 の CIDR 範囲を持つ 2-AZ アーキテクチャとしてデプロイされます。Amazon VPC IP Address Manager (IPAM) を、各 AZ に 1 つのパブリックサブネットと 1 つのプライベートサブネットで使用するオプションもあります。このソリューションでは、各パブリックサブネットに NAT ゲートウェイを作成し、プライベートサブネットに ENI を作成するように Lambda 関数を設定します。さらに、この設定はルートテーブルとそのエントリ、セキュリティグループとそのルール、ネットワーク ACL、VPC エンドポイント (ゲートウェイとインターフェイスエンドポイント) を作成します。

独自の Amazon VPC を管理する

Amazon VPC を使用してソリューションをデプロイする場合、AWS アカウントとリージョンの既存の Amazon VPC を使用するオプションがあります。高可用性を確保するために、少なくとも 2 つのアベイラビリティーゾーンで VPC を利用可能にすることをお勧めします。また、VPC とルートテーブルの設定には、次の VPC エンドポイントとそれに関連する IAM ポリシーが必要です。

デプロイメントダッシュボード用の Amazon VPC

1. DynamoDB のゲートウェイエンドポイント
2. S3 のゲートウェイエンドポイント。

3. CloudWatch のインターフェイスエンドポイント
4. AWS CloudFormation のインターフェイスエンドポイント

ユースケース用の Amazon VPC

1. DynamoDB のゲートウェイエンドポイント
2. S3 のゲートウェイエンドポイント。
3. CloudWatch のインターフェイスエンドポイント
4. Systems Manager Parameter Store のインターフェイスエンドポイント

> **Note**
>
> このソリューションで必要なのは `com.amazonaws.region.ssm` のみです。

5. Amazon Bedrock のインターフェイスエンドポイント (bedrock-runtime、agent-runtime、bedrock-agent-runtime)
6. オプション: デプロイで Amazon Kendra をナレッジベースとして使用する場合は、Amazon Kendra のインターフェイスエンドポイントが必要です。
7. オプション: デプロイで、Amazon Bedrock で任意の LLM を使用する場合は、Amazon Bedrock のインターフェイスエンドポイントが必要です。

> **Note**
>
> このソリューションに必要なのは `com.amazonaws.region.bedrock-runtime` のみです。

8. オプション: デプロイで、LLM に Amazon SageMaker AI を使用する場合は、Amazon SageMaker AI のインターフェイスエンドポイントが必要です。

> **Note**
>
> Bring your own VPC deployment (Bring-Your-Own-VPC デプロイ) オプションを使用する場合でも、ソリューションによって VPC 設定が削除または変更されることはありません。ただし、Create a VPC for me (VPC の自動作成) オプションでソリューションが作成する VPC はすべて削除されます。このため、ソリューションが管理する VPC をスタック/デプロイ間で共有する場合は注意が必要です。

例えば、デプロイ A では Create a VPC for me (VPC の自動作成) オプションを使用します。デプロイ B では、デプロイ A で作成された VPC を使用する Bring my own VPC (自分の VPC を使用) を使用します。デプロイ A がデプロイ B より前に削除されると、VPC が削除されてしまうためデプロイ B は機能しなくなります。また、デプロイ B は Lambda 関数によって作成された ENI を使用しているため、デプロイ A を削除するとエラーが発生し、残存リソースが保持される可能性があります。

Amazon CloudFront

このソリューションでは、Amazon S3 バケットでホストされる静的なウェブコンソールをデプロイします。レイテンシーを軽減し、セキュリティを向上させるために、このソリューションには、オリジンアクセスアイデンティティを持つ CloudFront ディストリビューションが含まれています。オリジンアクセスアイデンティティは、このソリューションのウェブサイトバケットにあるコンテンツに、パブリックアクセスを提供する CloudFront ユーザーです。詳細については、Amazon CloudFront デベロッパーガイドのオリジンアクセスアイデンティティを使用して Amazon S3 コンテンツへのアクセスを制限するを参照してください。

> **Note**
>
> CloudFront は、アカウントレベルのソフトクォータ制限として 20 の Response Header Policies を提供します。このソリューションは、セキュリティ上の目的でカスタム Response Header Policies を作成します。AWS での生成 AI アプリケーションビルダーまたはそのユースケースのデプロイが 20 を超える場合、クォータ制限に達したことが原因で新しいデプロイが失敗する可能性があります。

この問題を解決するには、以下の手順に従って、AWS Service Quotas コンソールで Response Header Policies クォータのクォータ引き上げをリクエストできます。

1. AWS Service Quotas コンソールを開きます。
2. ナビゲーションペインで、[AWS services (AWS のサービス)] を選択します。
3. [Amazon CloudFront] を検索して選択します。
4. [Response Header Policies] のクォータまでスクロールして、[Request quota increase] を選択します。
5. プロンプトに従って、AWS アカウントのクォータ制限の引き上げをリクエストします。

Response Header Policies クォータを引き上げると、AWS での生成 AI アプリケーションビルダーの新しいデプロイやそのユースケースで、クォータ制限が原因の失敗を回避できます。

クォータ

サービスクォータ (制限とも呼ばれます) は、AWS アカウントのサービスリソースまたはオペレーションの最大数です。

このソリューション内の AWS サービスのクォータ

このソリューションに実装されている各サービスに十分なクォータがあることを確認してください。詳細については、「AWS サービスクォータ」を参照してください。

次のリンクを使用すると、各サービスのページに移動できます。ページを切り替えずに、ドキュメント内のすべての AWS サービスのサービスクォータを表示するには、この PDF の「Service endpoints and quotas」ページの情報を参照してください。

Amazon Bedrock AgentCore のクォータ

エージェントビルダーのデプロイでは、次の Amazon Bedrock AgentCore サービスクォータに注意してください。

クォータ	米国東部 (バージニア北部)	その他のリージョン
アカウントあたりのアクティブなセッションワークロード	1,000	500
アカウントあたりのエージェント総数	1,000	1,000
アカウントあたりのバージョン	1,000	1,000

ソリューションをデプロイする

このソリューションは、AWS CloudFormation テンプレートとスタックを使用してデプロイを自動化します。CloudFormation テンプレートは、このソリューションに含まれる AWS リソースとそのプロパティを指定します。CloudFormation スタックは、テンプレートに記述されているリソースをプロビジョニングします。

デプロイプロセスの概要

ソリューションを起動する前に、コスト、アーキテクチャ、セキュリティなど、このガイドで説明されている考慮事項を確認してください。

> **Important**
>
> Amazon Bedrock を使用する場合は、使用の前にモデルへのアクセスをリクエストする必要があります。詳細については、「Amazon Bedrock ユーザーガイド」の「Model access」を参照してください。

デプロイ時間: 約 10 分

ステップ 1: デプロイダッシュボードスタックを起動する

ステップ 2: ユースケースをデプロイする

ステップ 3: デプロイダッシュボードウィザードを使用してユースケースをデプロイする

ステップ 4: デプロイ後の設定

必要に応じて、デプロイダッシュボードの UI または API を使用しない場合は、ユースケースをソリューションとは別にデプロイできます。

- スタンドアロンの Text ユースケースのデプロイ
- スタンドアロンの Bedrock エージェントユースケースのデプロイ

DynamoDB チャット設定を指定することもできます。

> **Important**
>
> このソリューションは、このソリューションの使用に関するオペレーションメトリクスを AWS (「データ」) に送信します。AWS ではこのデータを使用して、ユーザーがこのソリューション、関連サービスおよび製品をどのように使用しているかをよりよく理解し、提供するサービスや製品の改善に役立てます。AWS によるこのデータの収集には、AWS プライバシーポリシーが適用されます。

AWS CloudFormation テンプレート

このソリューションの CloudFormation テンプレートは、デプロイする前にダウンロードできます。

View template

generative-ai-application-builder-on-aws.template - このテンプレートを使用して、ソリューションと、関連するすべてのコンポーネントを起動します。デフォルト設定では、「このソリューションで使用している AWS のサービス」セクションに記載しているコアとサポートのサービスがデプロイされますが、特定のニーズに合わせてテンプレートをカスタマイズできます。

> **Note**
>
> AWS CloudFormation のリソースは、AWS Cloud Development Kit (AWS CDK) のコンストラクトで作成されています。

この AWS CloudFormation テンプレートは、AWS での生成 AI アプリケーションビルダーを AWS クラウドにデプロイします。

ステップ 1: デプロイダッシュボードスタックを起動する

このセクションのステップバイステップの手順に従って、ソリューションを設定してアカウントにデプロイします。

デプロイ時間: 約 10 分

1. AWS マネジメントコンソールにサインインし、`generative-ai-application-builder-on-aws.template` CloudFormation テンプレートを起動するボタンを選択します。

   Launch solution

2. テンプレートはデフォルトで米国東部 (バージニア北部) リージョンで起動します。別の AWS リージョンでソリューションを起動するには、コンソールのナビゲーションバーでリージョンセレクターを使用します。

   > **Note**
   >
   > このソリューションでは Amazon Kendra と Amazon Bedrock を使用しますが、これらのサービスは現在一部の AWS リージョンでは利用できません。これらの機能を使用する場合は、これらのサービスが利用可能な AWS リージョンでこのソリューションを起動する必要があります。リージョン別の最新情報については、AWS リージョン別のサービスのリストを参照してください。

3. [スタックの作成] ページで、正しいテンプレート URL が [Amazon S3 URL] テキストボックスに表示されていることを確認し、[次へ] を選択します。
4. [スタックの詳細を指定] ページで、ソリューションのスタックに名前を割り当てます。名前に使用する文字の制限に関する詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM と AWS STS クォータ」を参照してください。
5. [パラメータ] で、このソリューションのテンプレートパラメータを確認し、必要に応じて変更します。このソリューションでは、次のデフォルト値を使用します。

パラメータ	デフォルト	説明
Admin User Email	No	デプロイダッシュボードにアクセスできる管理者ユーザーの E メールアドレス。指定すると、ユースケースをデプロイおよび管理するアクセス許可を持つ Amazon Cognito グループとユーザーが作成されます。`placeholder@example.com` を使用してユーザーではなくグルー

パラメータ	デフォルト	説明
		プを作成することもできます。ユーザープールの設定については、「手動ユーザープール設定」を参照してください。
VpcEnabled	No	デプロイダッシュボードを VPC 内にデプロイする必要があるか
CreateNewVpc	No	VpcEnabled が Yes の場合にのみ使用できます。値が Yes の場合、スタックによって VPC が作成され、作成された VPC 内にソリューションがデプロイされます。 VpcEnabled が Yes で CreateNewVpc が No の場合、既存の VPC 設定 (ExistingVpcId、ExistingPrivateSubnetIds、ExistingSecurityGroupIds、VpcAzs) を指定する必要があります。
IPAMPoolId	(オプション入力)	IPAM を設定し、作成した ID を入力として指定して、このスタックのデプロイで使用する IP アドレス範囲を割り当てることができます。IPAM の詳細については、「IPAM の仕組み」を参照してください。

パラメータ	デフォルト	説明
DeployUI	Yes	デプロイダッシュボードは、ウェブユーザーインターフェイス (およびウェブデプロイに必要な AWS リソース) なしでデプロイできます。この場合、ソリューションは REST API エンドポイントを含むすべてのインフラストラクチャをデプロイします。このオプションは、独自のウェブインターフェイスをデプロイダッシュボード API と統合するのに便利です。
ExistingVpcId	(オプション入力)	作成した既存の VPC にソリューションをデプロイする場合にのみ必要です。
ExistingPrivateSubnetIds	(オプション入力)	作成した既存の VPC にソリューションをデプロイする場合にのみ必要です。Lambda 関数はこのサブネットにデプロイされます。
ExistingSecurityGroupIds	(オプション入力)	作成した既存の VPC にソリューションをデプロイする場合にのみ必要です。セキュリティグループにアウトバウンド TCP 接続のアクセス許可があることを確認します。
VpcAzs	(オプション入力)	作成した既存の VPC にソリューションをデプロイする場合にのみ必要です。

パラメータ	デフォルト	説明
CognitoDomainPrefix	(オプション入力)	作成した既存の Amazon Cognito ユーザープールにソリューションをデプロイする場合にのみ必要です。値を指定しない場合、ソリューションが値を生成します。
ExistingCognitoUserPoolId	(オプション入力)	作成した既存の Amazon Cognito ユーザープールにソリューションをデプロイする場合にのみ必要です。
ExistingCognitoUserPoolClient	(オプション入力)	作成した既存の Amazon Cognito ユーザープールにソリューションをデプロイする場合にのみ必要です。値を指定しない場合、ソリューションがユーザープールクライアントを作成します。このパラメータは、ExistingCognitoUserPoolId 値を指定する場合にのみ指定できます。

6. [次へ] を選択します。
7. [スタックオプションの設定] ページで、[次へ] を選択します。
8. [確認および作成] ページで、設定を確認して確定します。テンプレートが AWS Identity and Access Management (IAM) リソースを作成することを確認するチェックボックスをオンにします。
9. [送信] を選択してスタックをデプロイします。

AWS CloudFormation コンソールの [ステータス] 列でスタックのステータスを確認できます。約 10 分後に CREATE_COMPLETE ステータスが表示されます。

ステップ 2: ユースケースをデプロイする

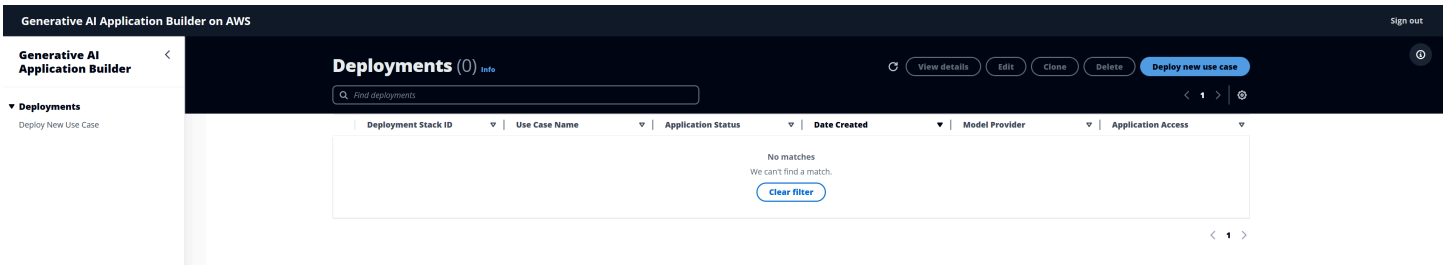
> **Important**
>
> スタックが正常にデプロイされると、設定した管理者ユーザーの E メールアドレスにサインアップ E メールが送信されます。管理者ユーザーはこれらの認証情報を使用してデプロイダッシュボードにサインインし、ウェブアプリケーションを使用できます。

> **Note**
>
> AWS マネジメントコンソールへのアクセス権を持つ DevOps ユーザーは、スタックの完了時にデプロイダッシュボード UI の CloudFront URL を管理者ユーザーに提供する必要があります。URL は CloudFormation スタックの [出力] タブにあります。

1. 管理者ユーザーとしてデプロイダッシュボードにサインインします。
2. アプリケーションのランディングページで、[Deploy new use case] を選択します。

   これにより、デプロイウィザードが起動し、ユースケースの作成手順が示されます。

デプロイダッシュボードのランディングページ - 新規デプロイを示しています



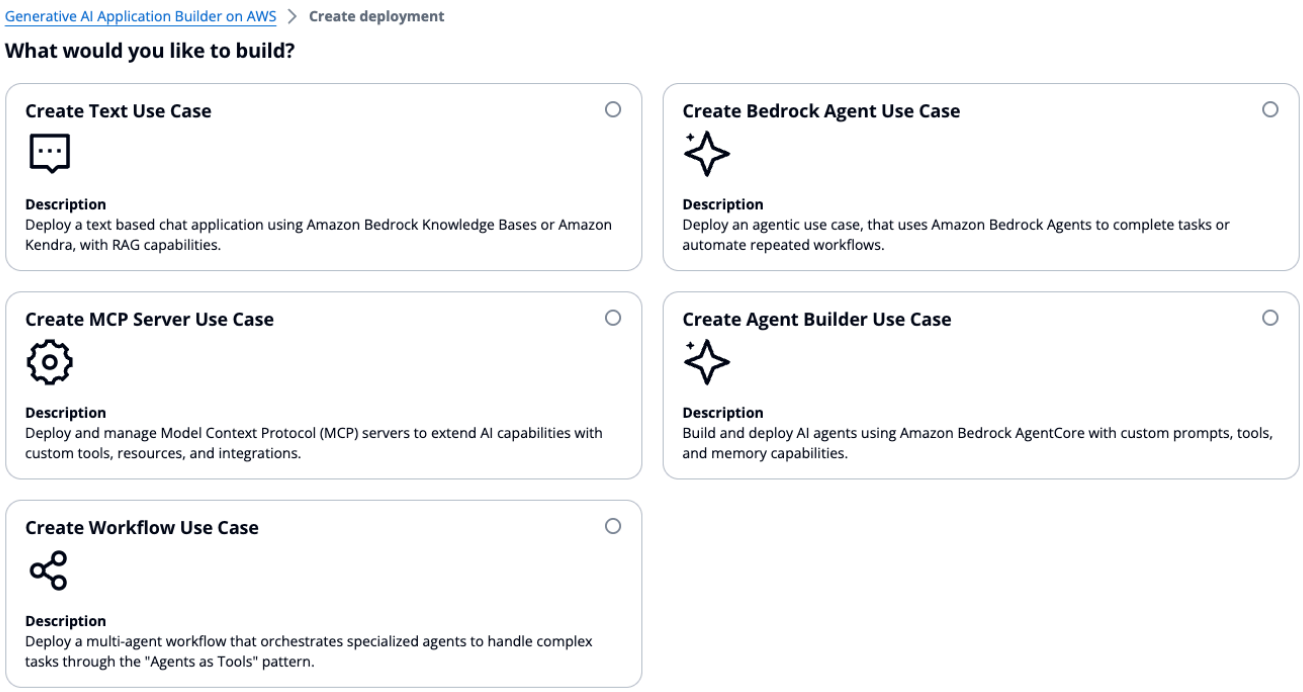
> **Note**
>
> デプロイにユーザーを追加する必要がある場合は、「Cognito ユーザープールの管理」を参照してください。

ステップ 3: デプロイダッシュボードウィザードを使用してユースケースをデプロイする

デプロイダッシュボードウィザードでは、次のいずれかを選択する必要があります。

- Text ユースケース - チャットアプリケーションをデプロイします (RAG 機能はオプション)。
- Bedrock エージェントユースケース - Amazon Bedrock エージェントを使用してタスクを完了したり、繰り返しワークフローを自動化したりします
- MCP サーバー - ゲートウェイまたはランタイムメソッドを使用して MCP サーバーをデプロイおよび管理します
- エージェントビルダー - MCP 統合とメモリ管理を使用して AgentCore にカスタムエージェントを構築およびデプロイします
- ワークフロービルダー - 階層的な委任を使用して複数のエージェントビルダーのエージェントをオーケストレーションします

5 つのオプションが表示されます。Text ユースケースの作成、Bedrock エージェントユースケースの作成、MCP サーバーユースケースの作成、エージェントビルダーユースケースの作成、ワークフローユースケースの作成。



ステップ 3a: Text ユースケースをデプロイする

このセクションでは、Text ユースケースをデプロイする手順について説明します。

ユースケースを選択する

[Create Text use case] を選択すると、UI で [Select use case] 画面が開きます。以下の情報を指定します。

- ユースケース名。
- ユースケースのデフォルトユーザー用にユースケースの Amazon Cognito ユーザープールに追加するオプションの E メールアドレス。このユーザーには、ユースケースとやりとりするためのアクセス許可が付与されます。
- このユースケースで UI をデプロイするかどうか。ユースケースで UI をデプロイしない場合は、デプロイされた API エンドポイントをアプリケーションで使用できます。

ユースケースの詳細

ユースケースの詳細ステップでは、デプロイの追加設定を行うことができます。

デフォルトでは、Text ユースケースでは、ソリューションによってデプロイダッシュボードがデプロイされるときに Amazon Cognito ユーザープールが作成・設定されます。ソリューションは、同じユーザープールに新しく作成されたクライアントを使用して新しいユースケースの認証を行います。ただし、ユースケースで独自の Amazon Cognito ユーザープールとクライアントを使用する場合は、このステップで既存のユーザープール ID とクライアント ID を指定できます。

> **Important**
>
> 管理者ユーザーは、Amazon Cognito ユーザープールがデプロイウィザードを通じて作成されたときに、デプロイされたすべてのユースケースにアクセスできます。デプロイ中に独自のユーザープールを指定する場合は、デプロイされたユースケースにアクセスするためのアクセス許可が管理者に必要です。
> また、Cognito のアプリクライアントで許可されたコールバック URL と許可されたサインアウト URL を更新する必要があります。これを実行するには:
>
> 1. Cognito コンソールに移動します。
> 2. [ユーザープール] を選択します。
> 3. 使用するユーザープールを選択します。

4. 左側のメニューで [アプリクライアント] を選択します。
5. 変更するアプリクライアントを選択します。
6. [ログインページ] タブを選択します。
7. [編集] をクリックして URL を追加します。
8. [Save changes] (変更の保存) をクリックします。

ユースケースにユーザーを追加する必要がある場合は、「Cognito ユーザープールの管理」セクションを参照してください。

ネットワーク設定を選択する

このウィザードステップでは、既存または新規の Amazon Virtual Private Cloud (Amazon VPC) を使用してユースケースをデプロイできます。既存の VPC を選択する場合は、この VPC で使用する VPC ID、最大 16 個のサブネット ID、最大 5 個のセキュリティグループ ID を指定する必要があります。既存の VPC を使用しない場合は、これらの設定は自動的に設定されます。

モデルを選択する

[モデルを選択する] ステップでは、ドロップダウンメニューからモデルプロバイダーを選択できます。[Bedrock] と [SageMaker] の 2 つのオプションがあります。

[SageMaker] を選択した場合は、SageMaker AI コンソールで SageMaker AI モデルエンドポイントを作成し、モデルが期待する入力スキーマと LLM 応答用の出力 JSONPath を指定することもできます。「LLM プロバイダーとしての Amazon SageMaker AI の使用」セクションと、ソリューションの GitHub リポジトリにある SageMaker AI ペイロードの例を参照してください。

[Amazon Bedrock] を選択すると、次の 4 つのオプションが表示されます。

- クイックスタートモデル - さまざまな価格/パフォーマンス特性を持つモデルのコレクションを使用してすぐに開始できます。最初のアプリケーションの構築に推奨されます。このオプションを使用すると、提供されたリストからモデル名を選択できます。
- その他の基盤モデル - さまざまな機能や専門分野を持つ幅広い基盤モデルにアクセスします。このオプションを使用すると、目的の Bedrock オンデマンド基盤モデルのモデル ID を入力できます。
- 推論プロファイル - 推論プロファイルは Bedrock のクロスリージョン推論を活用してピーク使用率バースト中に複数の AWS リージョンにリクエストをルーティングすることで、スループットを

向上させ。回復性を強化します。このオプションを使用すると、使用する推論プロファイルの ID を入力できます。

- プロビジョニング済みモデル - 一貫したパフォーマンスを必要とする本番ワークロード専用のスループットキャパシティ。このオプションを使用すると、Amazon Bedrock で使用するプロビジョニング済み/カスタムモデルの ARN を入力できます。

モデル選択ステップでは、モデルの詳細設定を選択することもできます。Amazon Bedrock ガードレール、Amazon Bedrock のプロビジョンドスループット、その他のモデルパラメータの詳細の設定については、「Advanced LLM Settings」を参照してください。

クロスリージョン推論

クロスリージョン推論は、Amazon Bedrock ユーザーが複数の AWS リージョンでコンピューティングを使用することで、計画外のトラフィックバーストをシームレスに管理できるようにします。クロスリージョン推論を使用するには、推論プロファイルが必要です。推論プロファイルは、設定された AWS リージョンからオンデマンドのリソースプールを抽象化したものです。ソースリージョンから送信された推論リクエストを、そのプールで設定された別のリージョンにルーティングできます。これにより、複数の AWS リージョンにトラフィックを分散でき、需要のピーク時に高いスループットと耐障害性を実現できます。

推論プロファイルは、サポートするモデルとリージョンにちなんで命名されます。使用するには、含まれているリージョンのいずれかから推論プロファイルを呼び出す必要があります。例えば、次の表に示すように、推論プロファイル ID `us.anthropic.claude-3-haiku-20240307-v1:0` では、選択したモデルの `us-east-1` リージョンと `us-west-2` リージョンを介したトラフィックの分散が許可されます。特定のモデルは、特定のリージョンの推論プロファイルでのみ使用できます。

推論プロファイル	推論プロファイル ID	含まれるリージョン
US Anthropic Claude 3 Haiku	`us.anthropic.claude-3-haiku-20240307-v1:0`	米国東部 (バージニア北部) (`us-east-1`) 米国西部 (オレゴン) (`us-west-2`)

モデル ID の代わりに推論プロファイル ID を使用する場合は、適切な推論プロファイル ID を特定する必要があります。詳細については、「Amazon Bedrock ユーザーガイド」の「Supported Regions and models for inference profiles」を参照してください。Amazon Bedrock コンソールでは、左側の

ナビゲーションメニューのクロスリージョン推論オプションに、これらの推論プロファイル ID が表示されます。

使用する推論プロファイル ID を特定したら、次のステップを実行してモデルの選択ステージでこれを使用できます。

1. モデルプロバイダーとして [Amazon Bedrock] を選択します。
2. [推論プロファイル] のラジオボタンオプションを選択します。
3. 表示されるテキストボックスに推論プロファイル ID を入力します。

推論プロファイルの詳細については、「Amazon Bedrock ユーザーガイド」の「Improve resilience with cross-region inference」を参照してください。

ナレッジベースを選択する

検索拡張生成 (RAG) を使用しないユースケースをデプロイする場合は、このステップをスキップできます。

ただし、デプロイの一環として RAG を有効にする場合は、事前設定された Amazon Kendra インデックス ID または Amazon Bedrock ナレッジベース ID を指定できるようになりました。ソリューションで使用するための新しい Amazon Kendra インデックスを作成することもできます。このソリューションは現在、RAG ベースのユースケースデプロイのナレッジベースとして Amazon Kendra と Amazon Bedrock ナレッジベースをサポートしています。

RAG ベースのデプロイで使用するナレッジベースへのデータの取り込みに関するガイドラインについては、「ナレッジベースの設定」セクションを参照してください。

高度な RAG 設定

ウィザードでは、RAG デプロイで使用する高度なオプションを選択できます。例えば、クエリがナレッジベースに送信されるたびに取得するドキュメントの数、ナレッジベースにドキュメントが見つからないときの LLM からの静的テキスト応答、LLM の応答にサニティチェック用のドキュメントソースを表示するかどうかなどを指定できます。また、Amazon Bedrock ナレッジベースで Amazon OpenSearch Serverless を使用する場合、ロールベースのアクセスコントロール (RBAC) や検索タイプの上書きなど、Amazon Kendra のナレッジベース固有の設定も指定できます。これらの詳細設定の詳細については、「高度なナレッジベースの設定」セクションを参照してください。

> **Note**
>
> ナレッジベースは、デプロイダッシュボードおよびユースケーススタックと同じアカウントとリージョンに存在する必要があります。

プロンプトとトークンの制限を選択する

このステップでは、LLM で使用するプロンプトを設定できます。プロンプトには、`{input}`、`{history}`、`{context}` などのプレースホルダーが必要になる場合があります。これらのプレースホルダーは、ユーザー入力、会話履歴、ナレッジベースから取得した情報をどこから参照するかを LLM に指示します。

- Bedrock モデルプロバイダーの場合、RAG 以外のユースケースに制限のないシステムプロンプトを指定する必要があります。ただし、Bedrock モデルプロバイダーのあいまいさ排除プロンプトには、少なくとも 2 つのプレースホルダー `{input}` と `{history}` が必要です。
- SageMaker モデルプロバイダー、システムプロンプト、曖昧さ解消プロンプトの場合、どちらも `{input}` と `{history}` の 2 つ以上のプレースホルダーが必要です。
- RAG ユースケースでは、モデルプロバイダーごとに、これに加えて `{context}` プレースホルダーが必要です。

詳細については、「プロンプトの設定」を参照してください。プロンプトのトークン制限サイズを選択する際は、「モデルトークンの制限を管理するためのヒント」セクションを参照することもできます。

マルチモーダル入力を有効にする

このステップでは、ユースケースに合わせてマルチモーダル入力機能を有効にすることができます。有効にすると、ユーザーはテキストクエリとともに画像やドキュメントをアップロードして送信できます。

サポートされているファイルタイプと制約:

- 画像: メッセージあたり最大 20 個の画像。各画像のサイズは 3.75 MB 以下、高さと幅は 8,000 ピクセル以下にする必要があります。サポートされている形式: png、jpeg、gif、webp
- ドキュメント: メッセージごとに最大 5 つのドキュメント。各ドキュメントのサイズは 4.5 MB 以下にする必要があります。サポートされている形式: pdf、csv、doc、docx、xls、xlsx、html、txt、md

マルチモーダル入力の使用方法:

1. ユースケースのデプロイ中に MultimodalEnabled パラメータを有効にする
2. チャットインターフェイスでは、ユーザーは次の 2 つの方法でファイルをアップロードできます。
   - チャット入力ボックスで、[ルール] ボタンをクリックする、または
   - チャットインターフェイスに直接ファイルをドラッグアンドドロップする
3. ファイルは Amazon S3 にアップロードされ、選択したモデルによって処理されます
4. アップロードされたファイルは 48 時間後に自動的に削除されます

ファイルステータスの追跡:

DevOps ユーザーは、アップロード時間と処理ステータスを含む DynamoDB 内のファイルメタデータをモニタリングできます。ファイルのステータスは次の通りです。

- 保留中 - ファイルのアップロードが開始されましたが、まだ完了していません。これは、署名付き URL が生成されたときの初期ステータスです。
- アップロード済み - ファイルは S3 に正常にアップロードされ、モデルによる処理の準備が整いました。
- 削除 - ファイルはユーザーによって削除され、処理のためにアクセスできなくなりました。
- 無効 - ファイルの検証チェックに失敗しました (ファイルタイプの不一致やセキュリティ検証の失敗など)。

アップロードされない保留中のステータスのファイルは、TTL の有効期限が切れると自動的にクリーンアップされます。アップロード済みステータスのファイルのみをモデルで処理できます。

S3 マルチモーダルバケットと DynamoDB メタデータテーブルは、それぞれキー `MultimodalDataBucketName` と `MultimodalDataMetadataTable` を使用してデプロイダッシュボードの出力で使用できます。

> **Note**
>
> すべてのモデルがマルチモーダル入力をサポートしているわけではありません。この機能を有効にする前に、選択したモデルが画像およびドキュメントの処理をサポートしていることを確認します。Amazon Bedrock ドキュメントでサポートされている基盤モデルを参照して、どのモデルが入力モダリティとして画像をサポートしているかを確認してください。

> **Important**
>
> ユーザーによってアップロードされたファイルは、48 時間のライフサイクルポリシーを使用して Amazon S3 に保存されます。アップロードされたファイルに関するメタデータは、会話履歴用の 24 時間の TTL を使用して Amazon DynamoDB に保存されます。

確認とデプロイ

このステップが完了したら、選択した設定内容を確認し、[Deploy Use Case] を選択します。新しいユースケースがデプロイされ、デプロイダッシュボードのビューに表示されて、管理できるようになります。

ステップ 3b: Bedrock エージェントユースケースをデプロイする

Bedrock エージェントユースケースは、ユースケース内で Amazon Bedrock エージェントを呼び出すための強力かつ安全なメカニズムを提供します。この機能により、開発者は、堅牢なセキュリティ対策を維持しながら、AI 搭載の自律型エージェントの機能を、基盤モデル、データソース、ソフトウェアアプリケーション、ユーザーとの会話などをまたいでマルチステップのタスクを編成および実行できるように、シームレスに統合できます。

前提条件

Amazon Bedrock エージェントを作成する前に、以下を準備してください。

1. AWS での生成 AI アプリケーションビルダーがデプロイされる AWS アカウント。Amazon Bedrock コンソールにアクセスできる必要があります。
2. Amazon Bedrock エージェントの作成および管理に必要な IAM アクセス許可。

Amazon Bedrock エージェントの作成

エージェントの作成に関する詳細な手順については、「Amazon Bedrock ユーザーガイド」の「Create and configure agent manually」を参照してください。次のようなオプションを設定できます。

- エージェント用の指示 (プロンプト)
- ユーザーの入力に基づいて追加情報を検索するためのナレッジベース
- エージェントが複数のセッション (最大 30 日間) にわたって情報を記憶できるようにするメモリ

Amazon Bedrock エージェントを作成したら、AWS での生成 AI アプリケーションビルダーの Bedrock エージェントユースケースのウィザードフローに進むことができます。そのためには、デプロイダッシュボードで [新しいユースケースをデプロイ] を選択し、[Bedrock エージェントユースケースの作成] を選択します。ウィザードに従い、次のステップを使用してユースケースを設定します。

ユースケースを選択する

このステップは、前述の Text ユースケースと同じです。

ネットワーク設定を選択する

このステップは、前述の Text ユースケースと同じです。

エージェントを選択する

このステップでは、作成した Amazon Bedrock エージェントのエージェント ID とエイリアス ID を指定する必要があります。

ステップ 3c: MCP サーバーのユースケースをデプロイする

MCP (モデルコンテキストプロトコル) サーバーのユースケースを使用すると、AI モデルやエージェントと統合できる MCP サーバーをデプロイおよび管理できます。MCP サーバーは、ツール、リソース、機能を AI アプリケーションに公開するための標準化された方法を提供します。既存の Lambda 関数と API から MCP サーバーを作成することも、コンテナイメージを使用してカスタム MCP サーバーをホストすることもできます。

前提条件

MCP サーバーのユースケースをデプロイする前に、以下があることを確認してください。

1. AWS での生成 AI アプリケーションビルダーがデプロイされる AWS アカウント。
2. Amazon Bedrock AgentCore リソースの作成および管理に必要な IAM アクセス許可。
3. 選択した作成メソッドによって異なります。
   - ゲートウェイメソッド (Lambda/API/MCP サーバー) の場合: Lambda 関数、対応するスキーマファイルを含む API エンドポイント (Lambda の場合は JSON 形式、API の場合は OpenAPI/Smithy)、または MCP サーバー URL エンドポイント
   - ランタイムメソッド (ECR) の場合: MCP サーバーの実装を含む Amazon ECR にプッシュされた Docker コンテナイメージ

MCP サーバーの作成方法

このソリューションは、MCP サーバーを作成するための 2 つの方法をサポートしています。

Lambda、API、または MCP サーバーから作成する (ゲートウェイメソッド)

このメソッドは、既存の Lambda 関数、REST API、または外部 MCP サーバーをラップする MCP ゲートウェイを作成し、MCP ツールとしてアクセスできるようにします。ゲートウェイは MCP と既存のサービス間のプロトコル変換を処理します。

- Lambda ターゲット: 関数の ARN と関数の入力/出力形式を記述する JSON スキーマファイルを提供することで、既存の Lambda 関数を統合します
- OpenAPI ターゲット: OAuth 2.0 または API キー認証をサポートする OpenAPI 仕様 (JSON または YAML 形式) を使用して REST API を統合します
- Smithy ターゲット: Smithy モデルファイル (.smithy または .json 形式) を使用して定義された API を統合します
- MCP サーバーターゲット: URL エンドポイントを介して外部 MCP サーバーに直接接続し、新しいインフラストラクチャをデプロイせずに既存の MCP サーバーを統合することができます

1 つの MCP ゲートウェイ内に、それぞれが異なるツールまたは機能を表す複数のターゲット (最大 10 個) を設定できます。

ECR イメージからのホスティング (ランタイムメソッド)

このメソッドでは、Amazon ECR イメージからコンテナ化された MCP サーバーをデプロイします。スタンドアロンサービスとして実行する必要があるカスタム MCP サーバー実装がある場合は、このアプローチを使用します。

- ECR イメージ URI を指定します (`:latest` や `:v1.0.0` などのタグを含める必要があります)。
- 必要に応じて、設定をコンテナに渡すように環境変数を設定します
- コンテナは MCP プロトコルを実装し、必要なエンドポイントを公開する必要があります

MCP サーバーのデプロイ

MCP サーバーのユースケースをデプロイするには、デプロイダッシュボードで [新しいユースケースをデプロイ] を選択し、[MCP サーバーユースケースの作成] を選択します。ウィザードに従い、次のステップを使用してユースケースを設定します。

ユースケースを選択する

このステップは、前述の Text ユースケースと同じです。

ネットワーク設定を選択する

現在、パブリックアクセスのみが有効化されており、ネットワーク設定では VPC はサポートされていません。

MCP サーバーの作成

このステップでは、MCP サーバーのデプロイを設定します。

MCP サーバーの作成メソッド

2 つの作成メソッドから選択します。

- Lambda、API、または MCP サーバーから作成: 既存の Lambda 関数、API 仕様、または外部 MCP サーバーエンドポイントから MCP ゲートウェイを作成します
- ECR イメージからのホスティング: コンテナイメージからカスタム MCP サーバーをデプロイします

> **Note**
>
> デプロイ後に作成メソッドを変更することはできません。メソッドを切り替える必要がある場合は、新しい MCP サーバーのユースケースをデプロイする必要があります。

ゲートウェイ設定 (Lambda/API/MCP サーバーメソッド用)

ゲートウェイメソッドを選択した場合は、1 つ以上のターゲットを設定します。

1. ターゲット名 (必須): このターゲット設定を識別するためのフレンドリ名
2. ターゲットの説明 (オプション): このターゲットの動作についての簡単な説明
3. ターゲットタイプ: 設定するターゲットのタイプを選択します。
   - Lambda: AWS Lambda 関数の場合
   - OpenAPI: OpenAPI 仕様の REST API の場合
   - Smithy: Smithy モデル定義を持つ API の場合
   - MCP サーバー: URL エンドポイント経由で外部 MCP サーバーに直接接続する場合

4. スキーマファイル (必須): ターゲットを記述するスキーマファイルをアップロードします。
   - Lambda の場合: 入出力形式を記述する JSON スキーマファイル。Lambda ツールスキーマの作成の詳細については、「Amazon Bedrock AgentCore デベロッパーガイド」の「Lambda ツールスキーマ」を参照してください。
   - OpenAPI の場合: OpenAPI 仕様ファイル (JSON または YAML)。OpenAPI スキーマの要件の詳細については、「Amazon Bedrock AgentCore デベロッパーガイド」の「OpenAPI スキーマ」を参照してください。
   - Smithy の場合: Smithy モデルファイル (.smithy または .json)。Smithy ターゲットの構築の詳細については、「Amazon Bedrock AgentCore デベロッパーガイド」の「Smithy ターゲットの構築」を参照してください。
5. Lambda 関数 ARN (Lambda ターゲットに必須): 統合する Lambda 関数の ARN
6. MCP サーバー URL (MCP サーバーターゲットに必須): 接続する外部 MCP サーバーの URL エンドポイント。URL は適切にエンコードされている必要があり、MCP サーバーは MCP プロトコルバージョン 2025-06-18 のツール機能をサポートしている必要があります。詳細については、「Amazon Bedrock AgentCore デベロッパーガイド」の「MCP サーバーターゲット」を参照してください。
7. アウトバウンド認証 (OpenAPI ターゲットに必須): REST API コールの認証を設定します。
   - 認証タイプ: OAuth 2.0 または API キーを選択します
   - アウトバウンド認証プロバイダー ARN: Amazon Bedrock AgentCore トークンボールトの認証情報プロバイダーの ARN
   - 追加設定: 認証タイプによって異なります。
     - OAuth 2.0 の場合: スコープとカスタムパラメータを設定します
     - API キーの場合: 場所 (ヘッダーまたはクエリパラメータ)、パラメータ名、オプションのプレフィックスを指定します

[別のターゲットを追加] を選択して、複数のターゲット (最大 10 個) を追加できます。各ターゲットは、MCP サーバーによって公開される個別のツールまたは機能を表します。

ECR 設定 (ECR イメージメソッドの場合)

ランタイムメソッドを選択した場合は、以下を指定します。

1. ECR イメージ URI (必須): Amazon ECR の Docker イメージの完全な URI
   - 形式: `account-id.dkr.ecr.region.amazonaws.com/repository-name:tag`
   - イメージはデプロイと同じ AWS リージョンに存在する必要があります

- タグが必要です (例: `:latest`、`:v1.0.0`)

2. 環境変数 (オプション): 実行時にコンテナに渡すキーと値のペアを設定します
   - これらを使用して、設定、認証情報、またはカスタムフラグを指定します。
   - 最大 10 個の環境変数を追加できます

確認とデプロイ

MCP サーバーを設定した後、選択した設定を確認し、[ユースケースのデプロイ] を選択します。新しい MCP サーバーユースケースがデプロイされ、デプロイダッシュボードのビューに表示されて、さらに管理できるようになります。

> **Note**
>
> MCP サーバーのデプロイでは、ゲートウェイ、ランタイム、ワークロード ID などのリソースが Amazon Bedrock AgentCore に作成されます。これらのリソースはソリューションによって自動的に管理され、ユースケースを削除するとクリーンアップされます。

ステップ 3d: エージェントビルダーユースケースをデプロイする

エージェントビルダーを使用すると、Amazon Bedrock AgentCore で本番環境対応の AI エージェントを作成、設定、デプロイできます。この機能は、システムプロンプト、モデル選択、MCP サーバー統合、メモリ管理を通じて、エージェントの動作を完全に制御します。

デプロイプロセスは基本的に Text ユースケースの場合と同じですが、いくつかの大きな違いがあります。

ユースケースを選択する

このステップは、前述の Text ユースケースと同じです。

ユースケースの詳細

このステップは、前述の Text ユースケースと同じです。

エージェントを設定する

このステップでは、システムプロンプト、使用可能な MCP サーバー/Strands ツール、メモリなどのコアエージェント設定を構成します。

システムプロンプト

システムプロンプトは、エージェントの動作、パーソナリティ、および機能を定義します。以下の操作を実行できます。

- デフォルトのシステムプロンプトテンプレートを編集する
- [デフォルトにリセット] ボタンを使用して元のテンプレートを復元する
- ツールの使用方法とレスポンスのフォーマットに関する手順を含める

MCP サーバー統合 (オプション)

モデルコンテキストプロトコルサーバーを設定して、エージェントにエンタープライズツールとデータへのアクセスを提供します。

1. ドロップダウンから使用可能な MCP サーバーを選択する
2. エージェントがアクセスできるすぐに使用できるツールを確認する

> **Note**
>
> デプロイする前に、MCP サーバーを設定してアクセス可能にする必要があります。サーバーのセットアップ手順については、MCP のドキュメントを参照してください。

メモリ設定

エージェントがコンテキストと知識を維持する方法を設定します。

- 短期メモリ: すべてのエージェントに対してデフォルトで有効になっています。セッション内の会話コンテキストを維持します。
- 長期メモリ: 切り替えて、セッション間でのインサイトの抽出と保存を有効にします。セマンティックメモリ戦略を備えた AgentCore Memory を使用します。

確認とデプロイ

このステップが完了したら、選択した設定内容を確認し、[Deploy Use Case] を選択します。エージェントビルダーのデプロイは通常 10～15 分で完了します。新しいユースケースがデプロイダッシュボードのビューに表示されて、さらに管理できるようになります。

ステップ 3e: ワークフローユースケースをデプロイする

ワークフロービルダーを使用すると、エージェントをツールとして委任するパターンを使用して複数のエージェントビルダーのエージェントをオーケストレーションするスーパーバイザーエージェントを作成できます。この機能を使用すると、既存のエージェントビルダーのデプロイを再利用して、複雑なマルチエージェントワークフローを構築できます。

デプロイプロセスは、エージェントビルダーと同様のパターンに従いますが、エージェントの検出と選択のための追加のステップがあります。

ユースケースを選択する

このステップは、前述の Text ユースケースと同じです。

ユースケースの詳細

このステップは、前述の Text ユースケースと同じです。

スーパーバイザーエージェントを設定する

このステップでは、専門のエージェントビルダーエージェントを調整するスーパーバイザーエージェントを設定します。

システムプロンプト

システムプロンプトは、スーパーバイザーエージェントが専門エージェントに作業を委任する方法を定義します。以下の操作を実行できます。

- デフォルトのシステムプロンプトテンプレートを編集する
- エージェントの選択と委任の手順を含める
- 複数のエージェントからの結果を集約する方法を定義する
- [デフォルトにリセット] ボタンを使用して元のテンプレートを復元する

> **Note**
>
> システムプロンプトは、各専門エージェントをいつどのように使用するかを明確に記述する必要があります。エージェントの説明は、適切な委任に不可欠です。

モデルの選択

スーパーバイザーエージェントの基盤モデルを選択します。スーパーバイザーエージェントは、このモデルを使用して以下を行います。

- ユーザーのリクエストを理解する
- 適切な専門エージェントを選択する
- エージェントの実行を調整する
- レスポンスを集計してフォーマットする

専門エージェントを選択する

このステップでは、スーパーバイザーが作業を委任できるエージェントビルダーのエージェントを選択します。

エージェントの追加

1. [エージェントを追加] をクリックして、エージェント選択ダイアログを開きます
2. リストから 1 つ以上のエージェントビルダーのエージェントを選択します
3. スーパーバイザーに提供されるエージェントの説明を確認します
4. 選択を確定します

> **Note**
>
> - ワークフローでは、専門エージェントとして少なくとも 1 つのエージェントビルダーのユースケースが必要です
> - ワークフローを作成する前に、すべての専門エージェントを正常にデプロイする必要があります

確認とデプロイ

以下を含むワークフロー設定を確認します。

- スーパーバイザーエージェントのシステムプロンプトとモデル
- 専門エージェントのリスト
- [メモリの設定]

[ユースケースのデプロイ] を選択します。ワークフローのデプロイは通常 15～20 分で完了します。新しいワークフローがデプロイダッシュボードのビューに表示されて、さらに管理できるようになります。

ステップ 4: デプロイ後の設定

このセクションでは、デプロイ後にソリューションを設定する際の推奨事項を説明します。

Amazon S3 バケットのバージョニング、ライフサイクルポリシー、クロスリージョンレプリケーション

このソリューションでは、作成したバケットにライフサイクル設定を適用しません。次の構成を推奨します。

- 本番デプロイのライフサイクル設定を指定する。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「バケットに S3 ライフサイクル設定を設定する」を参照してください。
- ソリューションをデプロイするユースケースに基づいて、Amazon S3 バケットのバージョニングとクロスリージョンレプリケーションを有効にする。

Amazon DynamoDB のバックアップ

このソリューションでは、複数の目的で DynamoDB を使用します (「このソリューションで使用している AWS のサービス」を参照)。このソリューションでは、作成したテーブルのバックアップは有効にしません。本番デプロイでは、この機能のバックアップを作成することをお勧めします。詳細については、「DynamoDB テーブルのバックアップ」と「DynamoDB での AWS Backup の使用」を参照してください。

Amazon CloudWatch のダッシュボードとアラーム

このソリューションは、CloudWatch にカスタムダッシュボードをデプロイして、カスタムの発行済みメトリクスと AWS サービスメトリクスからグラフをレンダリングします。CloudWatch アラームを作成し、ソリューションをデプロイするユースケースに基づいて通知を追加することをお勧めします。

Amazon CloudWatch Logs

Lambda ログは有効期限が切れないように設定され、API Gateway ログは有効期限が 10 年に設定されています。各ロググループの有効期限は、企業のレコード保持ポリシーに合わせて更新することができます。

TLS v1.2 以降の証明書を使用するカスタムウェブドメイン

このソリューションは、CloudFront を使用してウェブ UI とエッジ最適化 API Gateway をデプロイします。CloudFront のドメインでは TLS v1.2 以降の証明書は適用されません。Amazon Route 53 を使用してカスタムドメインを作成する、AWS Certificate Manager を使用して証明書を作成する、または組織に既存の証明書がある場合はその証明書を使用することをお勧めします。

詳細については、Amazon Route 53 デベロッパーガイドおよび「API Gateway で REST API カスタムドメインのセキュリティポリシーを選択する」を参照してください。

Amazon Kendra によるスケーリング

このソリューションでは、Amazon Kendra を使用して、取り込まれたドキュメント全体で NLP を活用したインテリジェント検索を実行できます。大規模なワークロードの場合は、次の CloudFormation パラメータを使用して Amazon Kendra の容量を増やすことができます。

パラメータ	デフォルト	説明
Amazon Kendra additional query capacity	0	インデックスの余分なクエリキャパシティおよび GetQuerySuggestions キャパシティの量。インデックス用の追加キャパシティユニット 1 つにつき、1 日あたり約 8,000 件のクエリに対応します。
Amazon Kendra additional storage capacity	0	インデックスの余分なストレージキャパシティーの量。キャパシティユニット 1 つにつき 30 GB のストレージキャパシティまたは 100,000 件の

パラメータ	デフォルト	説明
		ドキュメント (いずれか早く達した方) に対応します。
Amazon Kendra edition	`Developer`	Amazon Kendra には、インデックスを作成するための Developer と Enterprise のエディションが用意されています。Amazon Kendra の各エディションの違いの詳細については、Amazon Kendra の料金表を参照してください。

これらの CloudFormation パラメータの値を変更するには、スタックのデプロイ時に適切な値を選択します。クエリとストレージのキャパシティユニットの詳細については、「Adjusting capacity」を参照してください。

> **Note**
>
> Text ユースケースのデプロイで RAG を有効にしない場合、Amazon Kendra インデックスは使用も作成もされません。

Idp フェデレーションを使用する SSO のセットアップ

このソリューションでは、SAML または OIDC ベースの ID フェデレーションをサポートする外部 ID プロバイダーとの統合が可能です。ソリューションのデプロイ時に、デプロイダッシュボードと各ユースケース用に、Amazon Cognito ユーザープールと個別のアプリケーションクライアント統合が作成されます。外部 Idp に基づいて、Amazon Cognito デベロッパーガイドの「Configuring identity providers for your user pool」セクションに記載されている手順に従い、SSO をセットアップするデプロイダッシュボードまたはユースケースのアプリクライアント統合を選択します。

ユーザーグループ情報を RAG ベースのアーキテクチャのナレッジベースまたはベクトルストアに渡すには、外部 Idp のユーザーグループを Amazon Cognito ユーザーグループにマッピングする必要があります。このソリューションでは、初期構成としての Lambda 関数トリガーが提供されており、トークン生成前フェーズにマッピングされます。この Lambda 関数には group_mapping.json ファイルが含まれており、グループマッピングを行うにはこのファイルを更新する必要がありま

す。Amazon Cognito でサポートされている Lambda トリガーについては、「Customizing user pool workflows with Lambda triggers」を参照してください。

ユーザープールの手動設定

デプロイ中に管理者またはデフォルトのユーザー E メールを渡さないことを選択した場合は、Amazon Cognito で適切なユーザーグループを手動で作成して、正しいアクセス許可を確保する必要があります。

1. デプロイダッシュボードで、Cognito ユーザープールに `Admin` という名前のグループを作成します。
2. ユースケースごとに、Cognito ユーザープールに `${UseCaseName}-Users` という名前のグループを作成します。ここで、`${UseCaseName}` はデプロイされたユースケースの名前です。

これらのグループは、承認メカニズムが正しく機能するために必要です。アクセスを許可するユーザーは、適切なグループに追加する必要があります。

`placeholder@example.com` が渡された場合、Cognito グループが作成されますが、関連付けられたユーザーを作成してグループに割り当てる必要があります。

ログイン画面のカスタマイズ

このソリューションでは、Amazon Cognito がホストする UI を使用してログインページをレンダリングします。組み込みのサインインページをカスタマイズするには、「Amazon Cognito デベロッパーガイド」の「Customizing the built-in sign-in and sign-up webpages」を参照してください。

セキュリティに関するその他の考慮事項

ソリューションをデプロイするユースケースに基づいて、次のセキュリティ上の推奨事項を確認してください。

- カスタマーマネージド AWS KMS 暗号化キー - このソリューションでは、追加費用のかからない AWS マネージド AWS KMS キーがデフォルトで使用されます。ユースケースを確認して、カスタマーマネージド AWS KMS キーを使用するようにソリューションを更新する必要があるかどうかを判断してください。
- API Gateway スロットリングルール - このソリューションは、API Gateway にデフォルトのスロットリングルールを設定してデプロイされます。ユースケースと予想されるトランザクション量に基づいて、API のスロットリングを設定することをお勧めします。詳細については、「Amazon

API Gateway デベロッパーガイド」の「API Gateway のスループットを向上させるために REST API へのリクエストをスロットリングする」を参照してください。

- AWS CloudTrail を有効にする - 推奨されるセキュリティ対策として、ソリューションがデプロイされている AWS アカウントで AWS CloudTrail を有効にして、AWS アカウントで API コールをログ記録することを検討してください。詳細については、「CloudTrail ユーザーガイド」を参照してください。
- ドリフト検出 - CloudFormation スタックでドリフト検出を設定して、デプロイされたソリューションスタックへの意図しない変更や悪意のある変更を特定し、通知を受け取ることをお勧めします。詳細については、「Implementing an alarm to automatically detect drift in AWS CloudFormation stacks」を参照してください。
- Cognito JSON ウェブトークン (JWT) - このソリューションは、Amazon Cognito が発行した JWT を使用して REST API エンドポイントで認証を行います。このソリューションでは、ID トークンとアクセストークンの有効期限は 5 分に設定されています。ユーザーがログアウトすると、新しいトークンを生成できなくなります (更新トークンは失効します)。ただし、現在のトークンの有効期限が切れるまでは、API エンドポイントへのリクエストは有効なトークンがあるため正常に認証されます。ユースケースのセキュリティ上の考慮事項を確認し、トークンの有効期間を調整してください。

ライフサイクルポリシーのカスタマイズ:

本番環境デプロイの場合は、保持要件に基づいてライフサイクルポリシーを確認して調整します。「Amazon Simple Storage Service ユーザーガイド」の「バケットのライフサイクル設定の指定」を参照してください。

マルチモーダルファイルストレージとライフサイクル

ユースケースでマルチモーダル入力機能を有効にした場合 (MultimodalEnabled を Yes に設定)、ソリューションはアップロードされたファイルを保存する Amazon S3 バケットと、ファイルメタデータを追跡する DynamoDB テーブルを作成します。

デフォルトのライフサイクルポリシー:

- S3 ファイル: 48 時間後に自動的に削除されます
- DynamoDB メタデータ: レコードは 24 時間後に期限切れになります (会話履歴 TTL)

セキュリティに関する考慮事項:

- ファイルはユースケース ID、ユーザー ID、会話 ID、メッセージ ID でパーティション化され、代わりに UUID 名でファイルが保存されます。UUID とファイル名のマッピングは、DynamoDB メタデータテーブルで確認できます
- ユーザーは、自分の会話内でアップロードしたファイルにのみアクセスできます
- ファイルタイプの検証は、マジックナンバー検出を使用して実行されます
- Amazon GuardDuty Malware Protection for S3 を有効にして、アップロードされたファイルをスキャンして悪意のあるコンテンツがないか確認することをお勧めします。

スタンドアロンの Text ユースケースのデプロイ

このセクションのステップバイステップの手順に従って、ソリューションを設定してアカウントにデプロイします。

デプロイ時間: 約 10 ~ 30 分

1. AWS マネジメントコンソールにサインインし、CloudFront テンプレートを起動するボタンを選択します。

BedrockChat.template	Launch solution
SageMakerChat.template	Launch solution

2. テンプレートはデフォルトで米国東部 (バージニア北部) リージョンで起動します。別の AWS リージョンでソリューションを起動するには、コンソールのナビゲーションバーでリージョンセレクターを使用します。

   注: このソリューションでは Amazon Kendra と Amazon Bedrock を使用しますが、これらのサービスは現在一部の AWS リージョンでは利用できません。これらの機能を使用する場合は、これらのサービスが利用可能な AWS リージョンでこのソリューションを起動する必要があります。リージョン別の最新情報については、AWS リージョン別のサービスのリストを参照してください。

3. [スタックの作成] ページで、正しいテンプレート URL が [Amazon S3 URL] テキストボックスに表示されていることを確認し、[次へ] を選択します。

4. [スタックの詳細を指定] ページで、ソリューションのスタックに名前を割り当てます。名前に使用する文字の制限に関する詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM と AWS STS クォータ」を参照してください。
5. [パラメータ] で、このソリューションのテンプレートパラメータを確認し、必要に応じて変更します。このソリューションでは、次のデフォルト値を使用します。

UseCaseUUID	<_####_>	アプリケーション内でデプロイされたこのユースケースを識別するための 36 文字の UUIDv4。
UseCaseConfigRecordKey	<_####_>	ランタイム時にチャットプロバイダー Lambda が必要とする設定を含むレコードに対応するキー。テーブル内のレコードには、この値に一致する key 属性と、必要な設定を含む config 属性が必要です。このレコードは、使用中の場合はデプロイプラットフォームによって入力されます。このユースケースをスタンドアロンでデプロイする場合は、UseCaseConfigTableName で定義されたテーブルに、手動で作成したエントリを追加する必要があります。
UseCaseConfigTableName	<_####_>	スタックは、この名前のテーブルからキー UseCaseConfigRecordKey で設定を読み込みます。
ExistingRestApiId	(オプション入力)	使用する既存の API Gateway REST API ID。指定しない場合、新しい API Gateway REST API が作成されます。

		通常、デプロイダッシュボードからデプロイするときに提供されます。 注: 既存の API を使用すると、複数のスタンドアロンユースケースをデプロイする必要がある場合に、リソースの重複を減らしAPI の管理を簡素化できます。スタンドアロンのユースケースに既存の API を提供する場合、API に必要なルート (複数可) と想定されるモデルが設定されていることを確認する責任があります。必要な事前設定済みの /details ルート (チャット中にユースケースの詳細を取得) と、オプションで /feedback ルート (FeedbackEnabled が Yes に設定されている場合、LLM チャットレスポンスのフィードバックの収集を有効にする) を設定する必要があります。さらに、ExistingApiRootResourceId、ExistingCognitoUserPoolId、および ExistingCognitoGroupPolicyTableName も指定する必要があります。

ExistingApiRootResourceId	(オプション入力)	使用する既存の API Gateway REST API ルートリソース ID。REST API ルートリソース ID は、API の「リソース」セクションでルートリソース (/) を選択することで、AWS コンソールから取得できます。リソース ID がリソース詳細パネルに表示されます。または、REST API で describe API コールを実行して、ルートリソース ID を見つけることもできます。
FeedbackEnabled	No	[いいえ] に設定すると、デプロイされたユースケーススタックはフィードバック機能にアクセスできなくなります。
ExistingModelInfoTableName	(オプション入力)	モデル情報とデフォルト値を含む DynamoDB テーブルの名前。デプロイプラットフォームによって使用されます。省略すると、モデルのデフォルト値を格納する新しいテーブルが作成されます。

DefaultUserEmail	placeholder@example.com	このユースケースのデフォルトユーザーの E メール。この E メールの Amazon Cognito ユーザーが作成され、ユースケースへのアクセスに使用されます。指定しない場合、Cognito グループとユーザーは作成されません。placeholder@example.com を使用してユーザーではなくグループを作成することもできます。ユーザープールの設定については、「手動ユーザープール設定」を参照してください。
ExistingCognitoUserPoolId	(オプション入力)	このユースケースの認証に使用する既存の Amazon Cognito ユーザープールの UserPoolId。通常、デプロイダッシュボードからデプロイする場合に指定しますが、このユースケーススタックをスタンドアロンでデプロイする場合は省略できます。
CognitoDomainPrefix	(オプション入力)	Cognito ユーザープールクライアントのドメインを指定する場合は、値を入力します。値を指定しない場合、デプロイによって値が生成されます。

ExistingCognitoUserPoolClient	(オプション入力)	既存のユーザープールクライアント (アプリクライアント) を使用する場合に指定します。ユーザープールクライアントを指定しない場合、新しいクライアントが作成されます。このパラメータは、既存のユーザープール ID が指定されている場合にのみ指定できます。
ExistingCognitoGroupPolicyTableName	(オプション入力)	ユーザーグループポリシーを格納する DynamoDB テーブルの名前。これは、ユースケースの API でカスタムオーソライザーによって使用されます。通常、デプロイプラットフォームからデプロイする際に入力を指定できますが、このユースケーススタックをスタンドアロンでデプロイする場合は省略できます。
RAGEnabled	`true`	true に設定すると、デプロイされたユースケーススタックは、RAG 機能を提供するために作成された、指定の Amazon Kendra インデックスを使用します。`false` に設定すると、ユーザーは LLM と直接やり取りします。

KnowledgeBaseType	Bedrock	RAG に使用するナレッジベースタイプ。RAGEnabled が `true` の場合にのみ設定されます。Bedrock または Kendra を使用できます。 注: RAGEnabled が true の場合にのみ該当します。
ExistingKendraIndexId	(オプション入力)	ユースケースで使用する既存の Kendra インデックスのインデックス ID。何も指定されておらず、KnowledgeBaseType が Kendra の場合、新しいインデックスが作成されます。 注: RAGEnabled が `true` で、KnowledgeBaseType が `Kendra` の場合にのみ該当します。
NewKendraIndexName	(オプション入力)	このユースケース用に新しく作成される Kendra インデックスの名前。ExistingKendraIndexId が指定されていない場合にのみ適用されます。 注: RAGEnabled が true で、KnowledgeBaseType が Kendra の場合にのみ該当します。

NewKendraQueryCapacityUnits	0	このユースケース用に新しく作成される Amazon Kendra インデックスの追加クエリキャパシティーユニット。ExistingKendraIndexId が指定されていない場合にのみ適用されます。「CapacityUnitsConfiguration」を参照してください。 注: RAGEnabled が `true` で、KnowledgeBaseType が `Kendra` の場合にのみ該当します。
NewKendraStorageCapacityUnits	`0`	このユースケース用に新しく作成される Amazon Kendra インデックスの追加ストレージキャパシティユニット。ExistingKendraIndexId が指定されていない場合にのみ適用されます。「CapacityUnitsConfiguration」を参照してください。 注: RAGEnabled が `true` で、KnowledgeBaseType が `Kendra` の場合にのみ該当します。

NewKendraIndexEdition	(オプション入力)	このユースケース用に新しく作成される Amazon Kendra インデックスに使用する Amazon Kendra のエディション。ExistingKendraIndexId が指定されていない場合にのみ適用されます。「Amazon Kendra Editions」を参照してください。 注: RAGEnabled が `true` で、KnowledgeBaseType が `Kendra` の場合にのみ該当します。
BedrockKnowledgeBaseId	(オプション入力)	RAG ユースケースで使用する Bedrock ナレッジベースの ID。ExistingKendraIndexId または NewKendraIndexName が指定されている場合は指定できません。 注: RAGEnabled が `true` で、KnowledgeBaseType が `Bedrock` の場合にのみ該当します。
VpcEnabled	No	スタックのリソースを VPC 内にデプロイするべきかどうか。
CreateNewVpc	No	ソリューションで新しい VPC を作成し、このユースケースで使用する場合は、`Yes` を選択します。 注: VpcEnabled が `Yes` の場合にのみ該当します。

IPAMPoolId	(オプション入力)	Amazon VPC IP Address Manager を使用して CIDR 範囲を割り当てる場合は、使用する IPAM プール ID を指定します。 注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
ExistingVpcId	(オプション入力)	ユースケースに使用する既存の VPC の VPC ID。 注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
ExistingPrivateSubnetIds	(オプション入力)	Lambda 関数のデプロイに使用する既存のプライベートサブネットのサブネット ID のカンマ区切りリスト。 注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
ExistingSecurityGroupIds	(オプション入力)	Lambda 関数の設定に使用する既存の VPC のセキュリティグループのカンマ区切りリスト。 注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。

VpcAzs	(オプション入力)	VPC のサブネットが作成される AZ のカンマ区切りリスト 注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
UseInferenceProfile	No	設定されたモデルが Bedrock の場合、Bedrock 推論プロファイルを使用しているかどうかを指定できます。これにより、スタックのデプロイ時に必要な IAM ポリシーが確実に設定されます。詳細については、次の「https://docs.aws.amazon.com/bedrock/latest/userguide/cross-region-inference.html」を参照してください。
DeployUI	はい	このデプロイでフロントエンド UI をデプロイするかどうかを選択します。No を選択すると、API をホストするインフラストラクチャ、API の認証、バックエンド処理のみが作成されます。

6. [次へ] を選択します。
7. [スタックオプションの設定] ページで、[次へ] を選択します。
8. [レビュー] ページで、設定を確認して確定します。テンプレートが AWS Identity and Access Management (IAM) リソースを作成することを確認するチェックボックスをオンにします。
9. [スタックの作成] を選択してスタックをデプロイします。

AWS CloudFormation コンソールの [ステータス] 列でスタックのステータスを確認できます。約 10 ～ 30 分で CREATE_COMPLETE ステータスが表示されます。

スタンドアロンの Bedrock エージェントユースケースのデプロイ

このセクションのステップバイステップの手順に従って、ソリューションを設定してアカウントにデプロイします。

デプロイ時間: 約 10 ~ 30 分

1. AWS マネジメントコンソールにサインインし、CloudFront テンプレートを起動するボタンを選択します。

BedrockAgent.template	Launch solution

2. テンプレートはデフォルトで米国東部 (バージニア北部) リージョンで起動します。別の AWS リージョンでソリューションを起動するには、コンソールのナビゲーションバーでリージョンセレクターを使用します。

> **Note**
>
> このソリューションでは Amazon Bedrock を使用しますが、このサービスは現在、一部の AWS リージョンでは利用できません。これらの機能を使用する場合は、これらのサービスが利用可能な AWS リージョンでこのソリューションを起動する必要があります。リージョン別の最新情報については、AWS リージョン別のサービスのリストを参照してください。

3. [スタックの作成] ページで、正しいテンプレート URL が [Amazon S3 URL] テキストボックスに表示されていることを確認し、[次へ] を選択します。
4. [スタックの詳細を指定] ページで、ソリューションのスタックに名前を割り当てます。命名文字の制限については、「AWS Identity and Access Management ユーザーガイド」の「{https---docs-aws-amazon-com-https---docs-aws-amazon-com-IAM-latest-UserGuide-reference-iam-limits-html} [IAM と AWS STS クォータ]」を参照してください。
5. [パラメータ] で、このソリューションのテンプレートパラメータを確認し、必要に応じて変更します。このソリューションでは、次のデフォルト値を使用します。

パラメータ	デフォルト値	説明
UseCaseUUID	*<_####_>*	アプリケーション内でデプロイされたこのユースケースを識別するための 36 文字の UUIDv4。
UseCaseConfigRecordKey	*<####>*	チャットプロバイダーの Lambda 関数が実行時に必要とする設定を含むレコードに対応するキー。 テーブル内のレコードには、この値に一致する key 属性と、必要な設定を含む config 属性が必要です。 このレコードは、使用中の場合はデプロイプラットフォームによって入力されます。このユースケースをスタンドアロンでデプロイする場合は、UseCaseConfigTableName で定義されたテーブルに、手動で作成したエントリを追加する必要があります。
UseCaseConfigTableName	*<####>* `	スタックは、ここで提供されたテーブルからユースケース設定を読み込み、UseCaseConfigRecordKey で定義されたレコードキーを使用します。

パラメータ	デフォルト値	説明
DefaultUserEmail	`placeholder@example.com`	このユースケースのデフォルトユーザーの E メール。このソリューションでは、この E メールの Amazon Cognito ユーザーを作成して、ユースケースへのアクセスに使用します。

パラメータ	デフォルト値	説明
ExistingRestApiId	(オプション入力)	使用する既存の API Gateway REST API ID。指定しない場合、新しい API Gateway REST API が作成されます。通常、デプロイダッシュボードからデプロイするときに提供されます。 注: 既存の API を使用すると、複数のスタンドアロンユースケースをデプロイする必要がある場合に、リソースの重複を減らしAPI の管理を簡素化できます。スタンドアロンのユースケースに既存の API を提供する場合、API に必要なルート (複数可) と想定されるモデルが設定されていることを確認する責任があります。必要な事前設定済みの /details ルート (チャット中にユースケースの詳細を取得) と、オプションで /feedback ルート (FeedbackEnabled が Yes に設定されている場合、LLM チャットレスポンスのフィードバックの収集を有効にする) を設定する必要があります。さらに、ExistingApiRootResourceId、ExistingCognitoUserPoolId、および ExistingCognitoGroupPolicyT

パラメータ	デフォルト値	説明
		ableName も指定する必要があります。
ExistingApiRootResourceId	(オプション入力)	使用する既存の API Gateway REST API ルートリソース ID。REST API ルートリソース ID は、API の「リソース」セクションでルートリソース (/) を選択することで AWS コンソールから取得できます。リソース ID はリソースの詳細パネルに表示されます。または、REST API で describe API コールを実行して、ルートリソース ID を見つけることもできます。
FeedbackEnabled	No	[いいえ] に設定すると、デプロイされたユースケーススタックはフィードバック機能にアクセスできなくなります。
CognitoDomainPrefix	(オプション入力)	Amazon Cognito ユーザープールクライアントのドメインを指定する場合は、値を入力します。値を指定しない場合、ソリューションが値を生成します。

パラメータ	デフォルト値	説明
ExistingCognitoUserPoolId	(オプション入力)	このユースケースの認証に使用する既存の Amazon Cognito ユーザープールの UserPoolId。注: 通常、デプロイダッシュボードからデプロイする場合にこの ID を指定しますが、このユースケーススタックをスタンドアロンでデプロイする場合は省略できます。
ExistingCognitoUserPoolClient	(オプション入力)	既存のユーザープールクライアント (アプリクライアント) を使用する場合に指定します。ユーザープールクライアントを指定しない場合、ソリューションがクライアントを作成します。このパラメータは、ExistingCognitoUserPoolId を指定した場合にのみ指定できます。
ExistingCognitoGroupPolicyTableName	(オプション入力)	ユーザーグループポリシーを格納する DynamoDB テーブルの名前。これは、ユースケースの API でカスタムオーソライザーによって使用されます。注: 通常、デプロイダッシュボードからデプロイする場合にこの名前を指定しますが、このユースケーススタックをスタンドアロンでデプロイする場合は省略できます。

パラメータ	デフォルト値	説明
VpcEnabled	No	スタックのリソースを VPC 内にデプロイするかどうか。
CreateNewVpc	No	ソリューションで新しい VPC を作成し、このユースケースで使用する場合は Yes を選択します。注: このパラメータは、VpcEnabled が Yes の場合にのみ該当します。
IPAMPoolId	(オプション入力)	IPAM を使用して CIDR 範囲を割り当てる場合は、使用する IPAM プール ID を指定します。注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
ExistingVpcId	(オプション入力)	ユースケースに使用する既存の VPC の VPC ID。注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
ExistingPrivateSubnetIds	(オプション入力)	Lambda 関数のデプロイに使用する既存のプライベートサブネットのサブネット ID のカンマ区切りリスト。注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。

パラメータ	デフォルト値	説明
ExistingSecurityGroupIds	(オプション入力)	Lambda 関数の設定に使用する既存の VPC のセキュリティグループのカンマ区切りリスト。注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
VpcAzs	(オプション入力)	VPC のサブネットが作成される AZ のカンマ区切りリスト 注: VpcEnabled が Yes で、CreateNewVpc が No の場合にのみ該当します。
BedrockAgentId	<####>	使用する Amazon Bedrock エージェントの ID。
BedrockAgentAliasId	<####>	使用する Amazon Bedrock エージェントのエイリアス ID。
DeployUI	Yes	このデプロイでフロントエンドチャット UI をデプロイするかどうかを選択します。No を選択すると、API をホストするインフラストラクチャ、API の認証、バックエンド処理が、チャット UI なしで作成されます。

6. [次へ] を選択します。
7. [スタックオプションの設定] ページで、[次へ] を選択します。
8. [確認] ページで、設定を確認して確定します。テンプレートが IAM リソースを作成することを確認するチェックボックスを選択します。

9. [スタックの作成] を選択してスタックをデプロイします。

AWS CloudFormation コンソールの [ステータス] 列でスタックのステータスを確認できます。約 10 ～30 分で CREATE_COMPLETE ステータスが表示されます。

DynamoDB チャット設定の指定

ユースケースをデプロイする場合、UseCaseConfigRecordKey と UseCaseConfigTableName は必須の CloudFormation パラメータです。これらのパラメータは通常、デプロイダッシュボードによって自動的に設定されます。デプロイダッシュボードのスタックは、このテーブルの作成と設定を処理し、デプロイ API への呼び出しによってパラメータが自動入力されます。

スタンドアロンでデプロイする場合は、次の作業を行う必要があります。

1. key のハッシュキーを持つ DynamoDB テーブルを作成します。
2. ユースケースの設定情報を含むレコードを、`{key: some_use_case_key, config: {your_configuration}.` の形式でテーブルに作成します。
3. デプロイ時に、選択した UseCaseConfigTableName パラメータおよび UseCaseConfigRecordKey (この例では `some_use_case_key`) パラメータをユースケーススタックに渡します。

スタンドアロンデプロイに適した設定を作成するには、デプロイダッシュボードから必要なユースケースを作成し、設定テーブルからレコードをコピーします。それ以外の場合は、以下の Bedrock デプロイの例を参考に、独自の設定を作成できます。

```
{
 "UseCaseName": "SampleUseCase",
 "ConversationMemoryParams": {
 "ConversationMemoryType": "DynamoDB",
 "HumanPrefix": "H",
 "AiPrefix": "A",
 "ChatHistoryLength": 20
 },
 "KnowledgeBaseParams": {
 "KnowledgeBaseType": "Bedrock",
 "NumberOfDocs": 2,
 "ScoreThreshold": 0,
 "ReturnSourceDocs": false,
 "BedrockKnowledgeBaseParams": {
```

```
 "BedrockKnowledgeBaseId": "SOME_ID",
 "OverrideSearchType": null
 }
 },
 "LlmParams": {
 "ModelProvider": "Bedrock",
 "BedrockLlmParams": { "ModelId": "anthropic.claude-v2" },
 "PromptParams": {
 "PromptTemplate": "some prompt",
 "MaxPromptTemplateLength": 187500,
 "MaxInputTextLength": 187500,
 "UserPromptEditingEnabled": true,
 "DisambiguationEnabled": true,
 "DisambiguationPromptTemplate": "some prompt"
 },
 "ModelParams": {},
 "Temperature": 1,
 "RAGEnabled": true,
 "Streaming": true,
 "Verbose": false
 }
}
```

Service Catalog AppRegistry によるソリューションのモニタリング

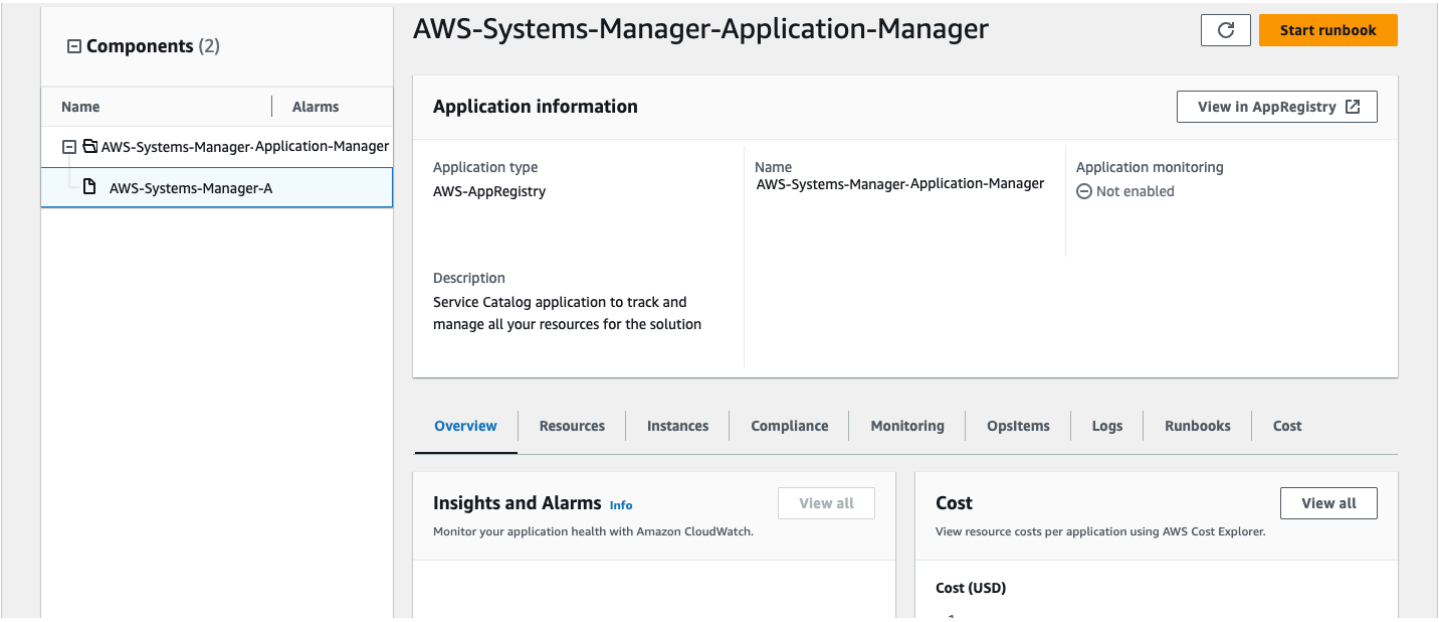
このソリューションには、CloudFormation テンプレートとその基礎となるリソースを、Service Catalog AppRegistry と Systems Manager Application Manager の両方にアプリケーションとして登録するための Service Catalog AppRegistry リソースが含まれています。

Systems Manager Application Manager は、このソリューションとリソースをアプリケーションレベルで確認できるため、次のようなことが可能になります。

- リソース、スタックや AWS アカウントにデプロイされたリソースのコスト、このソリューションに関連するログを一元的にモニタリングします。
- このソリューションのリソースの運用データをアプリケーションのコンテキストで表示します。これには、デプロイステータス、CloudWatch アラーム、リソース設定、運用上の問題などが含まれます。

次の図では、Application Manager のソリューションスタックでのアプリケーションビューの例を示しています。

Application Manager のソリューションスタックの図

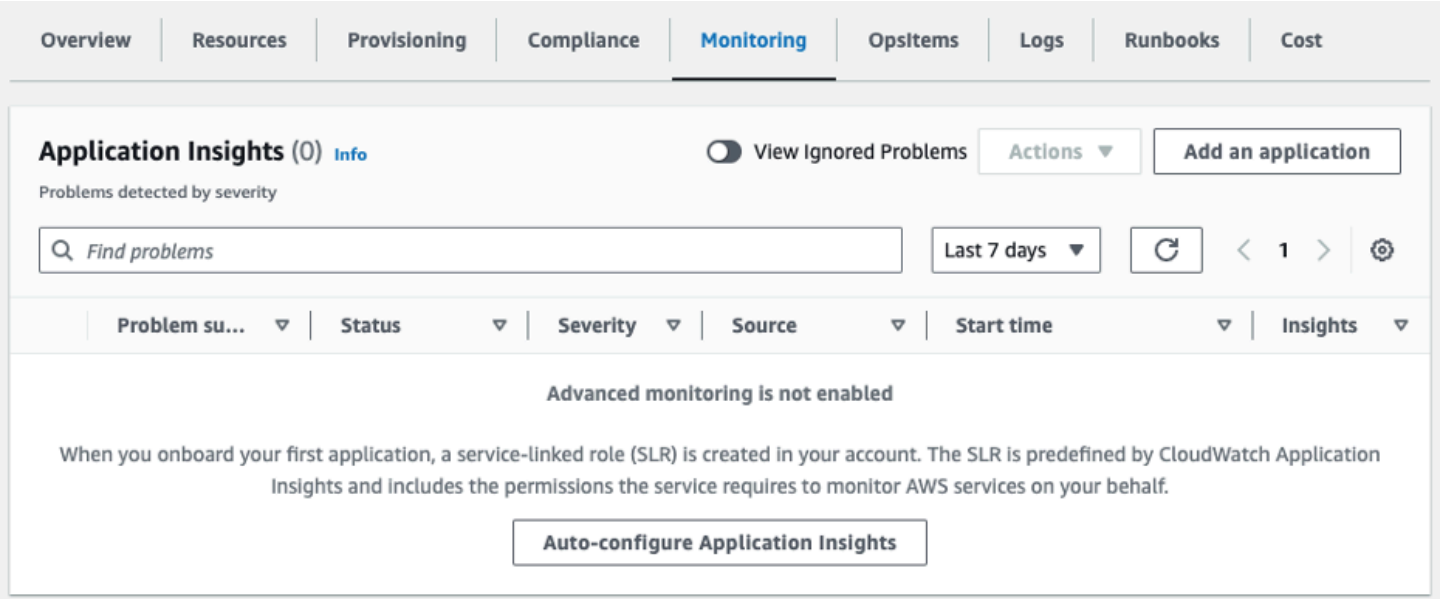


CloudWatch Application Insights アクティブ化する

1. Systems Manager コンソールにサインインします。
2. [Application Manager] を選択します。
3. [アプリケーション] で、このソリューションのアプリケーション名を検索して選択します。

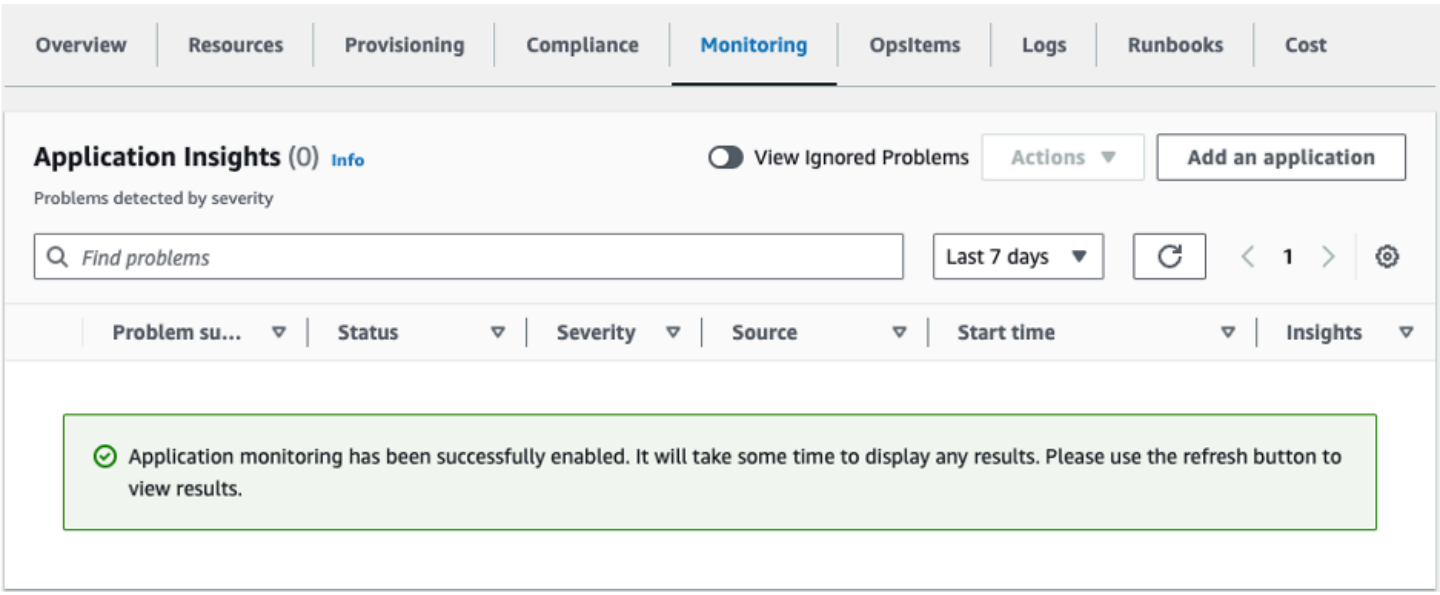
   アプリケーション名は、[アプリケーションソース] 列の [App Registry] と、ソリューション名、リージョン、アカウント ID、またはスタック名の組み合わせで構成されます。
4. [コンポーネント] ツリーで、アクティブにするアプリケーションスタックを選択します。
5. [モニタリング] タブの [Application Insights] で、[Application Insights を自動設定] を選択します。

Application Insights ダッシュボードには、問題が検出されず、自動設定のオプションが表示されます。



アプリケーションのモニタリングが有効になり、次のステータスボックスが表示されます。

モニタリングのアクティベーションに成功したことを示す Application Insights ダッシュボード



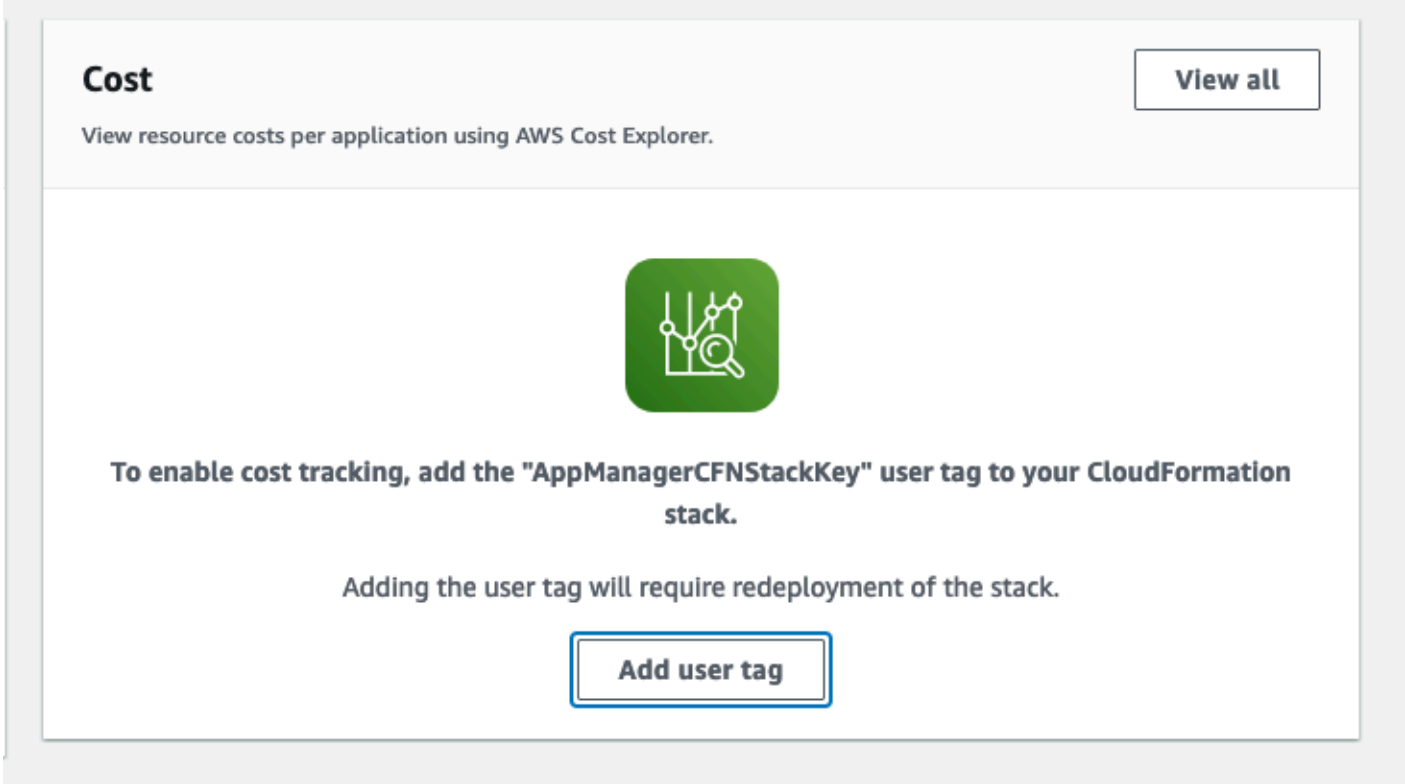
ソリューションに関連するコストタグを確認する

ソリューションに関連するコスト配分タグを有効にしたら、コスト配分タグを確認してこのソリューションのコストを確認する必要があります。次の手順で、コスト配分タグを確認します。

1. Systems Manager コンソールにログインします。
2. ナビゲーションペインで、[Application Manager] を選択します。
3. [アプリケーション] で、このソリューションのアプリケーション名を選択します。

   アプリケーション名は、[アプリケーションソース] 列の [App Registry] と、ソリューション名、リージョン、アカウント ID、またはスタック名の組み合わせで構成されます。
4. [概要] タブのコストで、[ユーザータグを追加] を選択します。

   アプリケーションの [コスト] の [ユーザータグを追加] 画面のスクリーンショット



5. [ユーザータグを追加] ページで、「`confirm`」と入力し、[ユーザータグを追加] を選択します。

アクティベーションプロセスが完了してタグデータが表示されるまでに最大 24 時間かかる場合があります。

ソリューションに関連するコスト配分タグをアクティブにする

Cost Explorer をアクティブ化したら、このソリューションに関連するコスト配分タグをアクティブ化して、このソリューションのコストを確認する必要があります。コスト配分タグは、組織の管理アカウントからのみアクティブ化できます。コスト配分タグをアクティブ化するには:

1. AWS Billing and Cost Management コンソールにサインインします。
2. ナビゲーションペインで、[コスト配分タグ] を選択します。
3. [コスト配分タグ] ページで、AppManagerCFNStackKey タグをフィルタリングし、表示された結果から同タグを選択します。
4. [アクティブ化] を選択します。

AWS Cost Explorer

アプリケーションおよびアプリケーションコンポーネントに関連するコストの概要は、AWS Cost Explorer との統合 (最初にアクティブ化する必要があります) により、Application Manager コンソール内で確認できます。Cost Explorer では、AWS リソースのコストと使用状況を時系列で表示することで、コストを管理できます。ソリューションに対して Cost Explorer をアクティブ化するには:

1. AWS Cost Management コンソールにサインインします。
2. ナビゲーションペインで [Cost Explorer] を選択し、ソリューションの経時的なコストと使用状況を表示します。

ソリューションを更新する

このソリューションを既にデプロイ済みの場合は、この手順に従ってソリューションの CloudFormation スタックを更新し、最新の機能や拡張機能を入手してください。アップグレードプロセスには、次の 3 つのパートがあります。

- ステップ 1: デプロイダッシュボードを更新する
- ステップ 2: ユースケースの設定を移行する
- ステップ 3: ユースケースをアップデートする

> **Note**
>
> 1. v2.0.0 では、Anthropic および Hugging Face との統合は廃止され、Amazon Bedrock および Amazon SageMaker AI との統合が推奨されます。Hugging Face で利用可能なモデルは、SageMaker JumpStart を介してデプロイできます。詳細については、「Amazon SageMaker AI での Hugging Face の使用」を参照してください。
> 2. 以下のステップを実行する前に、必ず本番環境以外の環境で更新プロセスをテストしてください。

ステップ 1: デプロイダッシュボードを更新する

1. CloudFormation コンソールにサインインし、既存の CloudFormation スタックを選択して、[更新] を選択します。
2. [既存テンプレートを置き換える] を選択します。
3. [テンプレートを指定] で、以下を実行します。
   a. [Amazon S3 URL] を選択します。
   b. 最新の CloudFormation テンプレートリンクをコピーします。
   c. [Amazon S3 URL] ボックスにリンクを貼り付けます。
   d. 正しいテンプレート URL が [Amazon S3 URL] テキストボックスに表示されていることを確認し、[次へ] を選択します。[次へ] をもう一度選択します。

4. [パラメータ] で、テンプレートのパラメータを確認し、必要に応じて変更します。パラメータの詳細については、「ステップ 1: デプロイダッシュボードスタックを起動する」を参照してください。
5. [次へ] を選択します。
6. [スタックオプションの設定] ページで、[次へ] を選択します。
7. [レビュー] ページで、設定を確認して確定します。テンプレートによって IAM のリソースが作成されることを確認するボックスをチェックします。
8. [変更セットの表示] を選択して、変更を確認します。
9. [スタックの更新] を選択してスタックをデプロイします。

AWS CloudFormation コンソールの [ステータス] 列でスタックのステータスを確認できます。約 10 分で UPDATE_COMPLETE ステータスが表示されます。

既存のソリューションのバージョンが v2.0.0 より前の場合、更新により、ウェブ UI スタック (ログイン画面の `amplify-ui` 実装を Cognito がホストする UI に置き換える) と新しい CloudFront URL が作成されます。これらの URL は、スタックのステータスが UPDATE_COMPLETE になると、CloudFormation コンソールの出力セクションから取得できます。

> **Note**
>
> v2.0.0 より前のバージョンを使用して作成された既存のユースケースは、以下の手順を完了するまで表示されません。

ステップ 2: ユースケース設定を移行する (2.0.0 より前のバージョンからの更新のみ)

バージョン 2.0.0 では、保存用のスキーマと、ユースケース設定を保存する AWS サービスが変更されました。gaab_v2_migration.py スクリプトを使用して、「GAAB v2 Migration User Guide」で説明されている手順に従ってください。スクリプトを実行したら、デプロイダッシュボードにアクセスして、デプロイ済みのユースケースを表示できます。

> **Note**
>
> ユースケースの移行を完了するには、以下のステップを実行する必要があります。

ステップ 3: ユースケースをアップデートする

最新バージョンの GAAB で利用可能な新機能を使用して、デプロイ済みのユースケースを編集できます。このソリューションでのこの機能の使用方法の詳細については、「ソリューションを使用する」を参照してください。

ユースケースを最新バージョンにアップデートするには、デプロイダッシュボードで「Edit」ユースケースステップを完了する必要があります (必ずしも変更を加える必要はありません)。このアクションは、最新のテンプレートバージョンで CloudFormation スタックの更新をトリガーします。

> **Note**
>
> 1.x または 2.x バージョンのソリューションで作成されたユースケースは、それ以降のバージョンでは機能しない場合があります。このため、デプロイダッシュボードを使用して、v3.0.0 より前のバージョンで作成された既存のユースケースのクローンを作成することをお勧めします。その後は、v3.0.0 以降で作成された新しいユースケースに段階的に移行して置き換えてください。

トラブルシューティング

このセクションでは、ソリューションをデプロイして使用するためのトラブルシューティングの手順を説明します。

これらの手順で問題が解決しない場合は、「サポートに問い合わせる」に、このソリューションに関するサポートケースを開く方法が記載されています。

問題: Create a VPC for me を使用して、VPC 対応設定のデプロイで VPC を作成すると失敗する

CloudFormation が VPC ネットワーキングリソースをプロビジョニングできなかったため、デプロイダッシュボードスタックまたはユースケーススタックのデプロイに失敗することがあります。

解決方法

ご使用のアカウントで VPC と Elastic IP のクォータ制限を確認してください。Elastic IP と VPC のデフォルトの制限は、AWS アカウント、AWS リージョンごとに 5 つです。

> **Note**
>
> ソリューションが VPC を作成する場合、単一の VPC 対応デプロイ (デプロイダッシュボードまたはユースケース) は、各 AZ に 1 つのパブリックサブネットと 1 つのプライベートサブネットがある 2-AZ デプロイであり、各パブリックサブネットは 1 つの NAT ゲートウェイをデプロイします。NAT ゲートウェイが 2 つある場合、デプロイではクォータ制限から 2 つのパブリック IP アドレスが使用されます。

注意すべき制限 (アカウントごと、リージョンごと):

- VPC の数 - 5
- パブリック IP アドレスの数 - 5
- ゲートウェイ VPC エンドポイントの数 - 20
- インターフェイス VPC エンドポイントの数 - 20

問題: デプロイダッシュボードスタックが削除された後、CloudFormation でユースケーススタックを削除できない

すべてのユースケーススタックが削除される前にデプロイダッシュボードスタックが CloudFormation で削除されると、ユースケースはロックされた (使用できない) 状態になる可能性があります。これは、デプロイダッシュボードスタックによって作成された IAM ロールが存在しなくなったことで、ユースケーススタックの変更が妨げられるためです。

解決方法

> **Warning**
>
> 手動で作成したロールは、使用後すぐにクリーンアップしてください。これらは昇格されたアクセス許可であり、ユーザーがロールの昇格に悪用する可能性があります。

削除した IAM ロールを再作成して、CloudFormation スタックの削除を有効にします。

1. CloudFormation コンソールを開き、ロックされたスタックに関連付けられているロールを特定します。
   a. ロール ARN は [IAM ロール] というラベルの付いたスタック情報セクションにあります。
   b. ロール名は、IAM ロール ARN の :role/ の後に続く部分です (例: arn:aws:iam::<account-id>:role/<role-name>)
2. 削除したロールと同じ名前の新しいロールを IAM に作成します。
   a. 信頼できるエンティティとして [AWS のサービス] を選択し、ドロップダウンから [CloudFormation] を選択します。
   b. 必要なアクセス許可を追加します。必要なアクセス許可が不明な場合は、AWS マネージドの AdministratorAccess ポリシーを使用できます。
   c. 手順 1 で取得したロール名を正確に入力します。
3. CloudFormation コンソールに戻り、ロックされたスタックを削除します。
4. ロックされたスタックがすべて正常に削除されたら、IAM に戻り、手順 2 で作成したロールをすべて削除します。

問題: ユースケースの UI に設定の変更が反映されない。

ユースケースが更新されると、UI は CloudFront にデプロイされます。ただし CloudFront は、デプロイとともに、一部の設定をユーザーに表示する方法を指定する設定ファイルをキャッシュするため、これらの変更がすぐには反映されない場合があります。

解決方法

CloudFront ディストリビューションを無効にして、新しい設定を強制的にフロントエンドユーザーに伝播することができます。

1. CloudFormation コンソールを開き、ユースケーススタックに関連付けられている CloudFront ディストリビューションを特定します。
   a. ユースケーススタックは、ユースケースのデプロイ時に使用したものと同じ名前で始まります。
   b. UI に対応するネストされたスタックを探します。ネストされたスタック名は、WebAppS3UINestedStackS3UINestedStackResource で始まります。
   c. [リソース] タブで、AWS::CloudFront::Distribution のリソースタイプを探し、物理 ID を選択します。これにより、CloudFront コンソールでディストリビューションが開きます。
2. [Invalidations] タブに移動し、[Create Invalidation] をクリックして、「/*」のパスを入力します。これにより、すべてのパスが無効になります。
3. お使いのブラウザで、ユースケースに関連するすべての Cookie とキャッシュファイルを削除します。

AWS サポートに問い合わせる

AWS ビジネスサポート+、AWS エンタープライズサポート、または Unified Operations をご利用の場合は、AWS サポートセンターを利用して、このソリューションに関するエキスパートのサポートを受けることができます。次のセクションで、その方法を説明します。

ケースを作成する

1. サポートセンターにサインインします。
2. [ケースを作成] を選択します。

どのようなサポートをご希望ですか?

1. [技術] を選択します。
2. [サービス] で、[ソリューション] を選択します。
3. [カテゴリ] で、[その他のソリューション] を選択します。
4. [重要度] で、ユースケースに最も適したオプションを選択します。
5. [サービス]、[カテゴリ]、[重要度] を入力すると、インターフェイスに一般的なトラブルシューティングの質問へのリンクが表示されます。これらのリンクを使用しても問題を解決できない場合は、[次のステップ: 追加情報] を選択してください。

追加情報

1. [件名] に、質問または問題を要約したテキストを入力します。
2. [説明] では、このソリューションの名前 (Generative AI Application Builder on AWS) を含め、問題を詳細に記述します。
3. [ファイルを添付] を選択します。
4. AWS サポートがリクエストを処理するために必要な情報を添付します。

ケースの迅速な解決にご協力ください

1. 必要な情報を記入します。
2. [次のステップ: 今すぐ解決またはお問い合わせ] を選択します。

今すぐ解決またはお問い合わせ

1. [今すぐ解決] で解決策を確認します。
2. これらの解決策で問題を解決できない場合は、[お問い合わせ] を選択し、必要な情報を入力して [送信] を選択します。

ソリューションをアンインストールする

> **Note**
>
> デプロイダッシュボードで作成されたデプロイは、ソリューションの外部での管理を意図したものではありません。CloudFormation でスタックを削除する前に、必ずデプロイダッシュボード内からデプロイをすべて削除してクリーンアップしてください。

AWS での生成 AI アプリケーションビルダーソリューションは、AWS マネジメントコンソールから、または AWS コマンドラインインターフェイスを使用してアンインストールできます。このソリューションで作成された Amazon S3 バケット、Amazon Kendra インデックス、または CloudWatch Logs を手動で削除する必要があります。AWS ソリューションでは、保持するデータを保存している場合でも、Amazon S3 バケット、Amazon Kendra インデックス、または CloudWatch Logs が自動的に削除されることはありません。

AWS マネジメントコンソールの使用

1. AWS CloudFormation コンソールにサインインします。
2. [スタック] ページで、このソリューションのインストールスタックを選択します。
3. [削除] を選択します。

AWS コマンドラインインターフェイスの使用

AWS コマンドラインインターフェイス (AWS CLI) が環境で使用可能かどうかを判断します。インストール手順については、「AWS CLI ユーザーガイド」の「AWS コマンドラインインターフェイスとは」を参照してください。AWS CLI が使用可能なことを確認したら、次のコマンドを実行します。

```
$ aws cloudformation delete-stack --stack-name <installation-stack-name>
```

手動アンインストールの手順

Amazon S3 バケットの削除

このソリューションでは、偶発的なデータ損失を防ぐために AWS CloudFormation スタックを削除する際に、Amazon S3 バケットを保持するように設定されています。このソリューションをアンインストールした後に、データを保持する必要がない場合は、Amazon S3 バケットを手動で削除できます。Amazon S3 バケットを削除するには、次の手順に従います。

1. Amazon S3 コンソール にサインインします。
2. ナビゲーションペインで、[バケット] を選択します。
3. <stack-name> S3 バケットを見つけます。
4. S3 バケットを選択し、続いて [削除] を選択します。

AWS CLI を使用して S3 バケットを削除するには、次のコマンドを実行してください。--force オプションを使用する場合、最初にバケットを空にする必要はありません。

```
$ aws s3 rb s3://<bucket-name> --force
```

Amazon Kendra インデックスの削除

このソリューションでは、偶発的なデータ損失を防ぐため、AWS CloudFormation スタックが削除された場合でも、ソリューションが作成した Amazon Kendra インデックスを保持するように設定されています。ソリューションをアンインストールした後、データを保持する必要がなくなった Amazon Kendra インデックスを手動で削除できます。次の手順に従って、Amazon Kendra インデックスを削除してください。

1. Amazon Kendra コンソールにサインインします。
2. ナビゲーションペインで、[インデックス] を選択します。
3. 削除するインデックスを見つけて選択します。
4. [Delete] (削除) を選択して、選択したインデックスを削除します。

AWS CLI を使用して Amazon Kendra インデックスを削除するには、次のコマンドを実行してください。

```
$ aws kendra delete-index --id<index-id>
```

CloudWatch Logs の削除

このソリューションでは、偶発的なデータ損失を防ぐため、CloudFormation スタックを削除する場合でも CloudWatch Logs を保持するように設定されています。このソリューションをアンインストールした後にデータを保持する必要がない場合は、ログを手動で削除できます。次の手順に従って、CloudWatch Logs を削除してください。

1. Amazon CloudWatch コンソールにサインインします。
2. ナビゲーションペインで、[ロググループ] を選択します。
3. このソリューションで作成されたロググループを見つけます。
4. ロググループから 1 つ選択します。
5. [アクション] を選択してから、[削除] を選択します。

すべてのソリューションのロググループを削除するまで、このステップを繰り返します。

ソリューションを使用する

UI へのアクセス

スタックのデプロイプロセス (デプロイダッシュボードとユースケースの両方) 中に、設定されたメールアドレスに E メールが送信されます。E メールには、ユーザーがサインアップしてウェブインターフェイスにアクセスするための一時的な認証情報が含まれています。

> **Note**
>
> AWS マネジメントコンソールへのアクセス権を持つ DevOps ユーザーは、スタックの完了時にデプロイダッシュボード UI の CloudFront URL を管理者ユーザーに提供する必要があります。

ユースケースの場合、デプロイダッシュボード UI にアクセスできる管理者ユーザーは、デプロイの完了時にユースケース UI の CloudFront URL をビジネスユーザーに提供する必要があります。

ログインすると、ユーザーはソリューション UI (管理者の場合はデプロイダッシュボード、ビジネスユーザーの場合はユースケース) を操作できます。

デプロイの更新方法

デプロイダッシュボードのホームページ (またはデプロイの詳細ページ) では、デプロイで使用される設定を編集できます。編集できるのは、CREATE_COMPLETE または UPDATE_COMPLETE ステータスにあるデプロイのみです。

ユースケース名を除く、デプロイの他のすべてのオプションを編集できます。編集したい値を変更して再デプロイするだけです。

再デプロイにかかる時間は、行った編集の範囲によって異なります。単純な設定 (モデルパラメータなど) が変更された場合は数秒、大規模なインフラストラクチャ関連のオプション (Text ユースケース RAG の Amazon Kendra インデックスの作成リクエストなど) が変更された場合は 30 分以上かかる場合があります。

編集が正常に完了すると、アプリケーションステータスに UPDATE_COMPLETE ステータスが表示されます。この時点で、CloudFront URL を使用してデプロイされた UI にアクセスし、変更されたデプロイを操作できます。

> **Note**
>
> さまざまな設定や LLM を比較したい場合は、複数のデプロイを並列的に実行する方が簡単な場合があります。クローン機能を使用すると、既存の設定をすばやく使用して新しいデプロイを起動できます。

デプロイのクローン作成方法

デプロイダッシュボードのホームページ (またはデプロイの詳細ページ) で、デプロイで使用される設定のクローンを作成できます。デプロイのクローンを作成すると、新規ユースケースをデプロイするウィザードが起動しますが、ほとんどのフィールドには同じ値が事前入力されています。

これは、設定を変更したデプロイのクローンをすばやく作成したり、削除したデプロイを復活させたり、他の点ではまったく同じデプロイで複数の LLM を比較したりする際に便利です。

デプロイの削除方法

デプロイダッシュボードのホームページ (またはデプロイの詳細ページ) で、不要になったデプロイを削除できます。デプロイを削除すると、CloudFormation スタックの削除オペレーションが呼び出され、デプロイのリソースのプロビジョニングが解除されます。

デフォルトでは、クローンの機能を有効にするために、削除されたデプロイは引き続きダッシュボードに残ります。デプロイをダッシュボードから完全に削除して UI で追跡されないようにするには、削除確認ウィンドウで [Permanently delete] を選択します。

> **Important**
>
> 一部のリソースはスタックの削除中に残るため、手動で削除する必要があります。保持されるリソースとそのクリーンアップ方法の詳細については、「手動アンインストール」セクションを参照してください。

大規模言語モデル (LLM) の設定

どの LLM がユースケースに適しているかは、ニーズやキュレーションしたいカスタマーエクスペリエンスのタイプに応じた多数の要因によって異なります。このソリューションは規範的なものではな

く、お客様のアプリケーションに最適なものを評価するために必要なツールを提供することを目的としています。

AI が生成する領域は急速に進化しています。そのため、顧客にとって適切なエクスペリエンスを確実に構築できるように、最新のモデル、最適化の手法、ベストプラクティスについての最新情報を常に把握しておく必要があります。

> **Note**
>
> 非公開データや機密データを扱う場合は、必ず AWS サービスを使用する LLM オプション (Amazon Bedrock または Amazon SageMaker AI など) を選択してください。これにより、サードパーティープロバイダーがホストする LLM を使用する場合と比較して、リージョン内と AWS ネットワーク上にデータが保持され、デプロイの全体的なセキュリティ体制が強化されます。

LLM プロバイダーとしての Amazon SageMaker AI の使用

v1.3.0 以降、Amazon SageMaker AI を Text ユースケースのモデルプロバイダーとして使用できるようになりました。この機能により、ソリューションで AWS アカウント内の既存の SageMaker AI 推論エンドポイントを使用できます。開始するための方法をいくつかご紹介します。

> **Important**
>
> このソリューションは SageMaker AI エンドポイントのライフサイクルは管理しません。追加料金が発生しないように、SageMaker AI エンドポイントが不要になったら削除する必要があります。

SageMaker AI エンドポイントの作成

Amazon SageMaker AI JumpStart を使用すると、エンドポイントをすばやくデプロイできます。

テキスト生成ベースの SageMaker AI エンドポイントを使用し、ベースの SageMaker AI サービスを使用してデプロイすることもできます。推論用にモデルをデプロイする方法のステップごとのガイドについては、SageMaker AI JumpStart ドキュメントを参照してください。

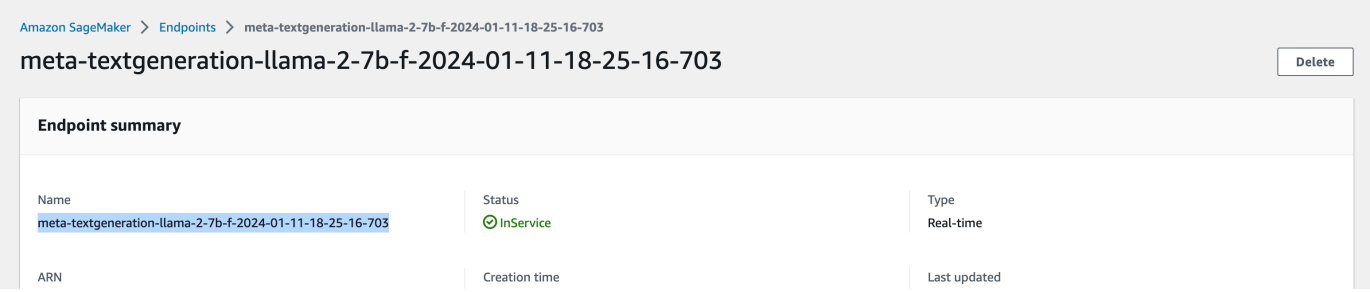
> **Note**
>
> 基盤モデル/LLM は通常かなり大きいため、多くの場合、大規模な高速コンピューティングインスタンスを使用する必要があります。これらの大規模なインスタンスの多くは、デフォルトでは AWS アカウントで使用できない場合があります。デプロイでよくある失敗を防ぐために、デフォルトの SageMaker AI クォータを参照し、デプロイ前に必ずクォータの引き上げをリクエストしてください。

SageMaker AI エンドポイントを使用して Text ユースケースのデプロイを作成する

推論用に SageMaker AI エンドポイントを使用して新しい Text ユースケースをデプロイするには:

1. デプロイウィザードを使用して新しいユースケースを作成し、モデルの選択ページが表示されるまでフォームに記入します。
2. モデルページで、モデルプロバイダーに [SageMaker AI] を選択します。これにより、次の 3 つの重要なユーザー入力を必要とするカスタムフォームが生成されます。
   - 使用する SageMaker AI エンドポイントの名前。DevOps ユーザーは AWS コンソールからこの情報を取得できます。エンドポイントは、ソリューションがデプロイされているのと同じアカウントとリージョンにある必要があることに注意してください。

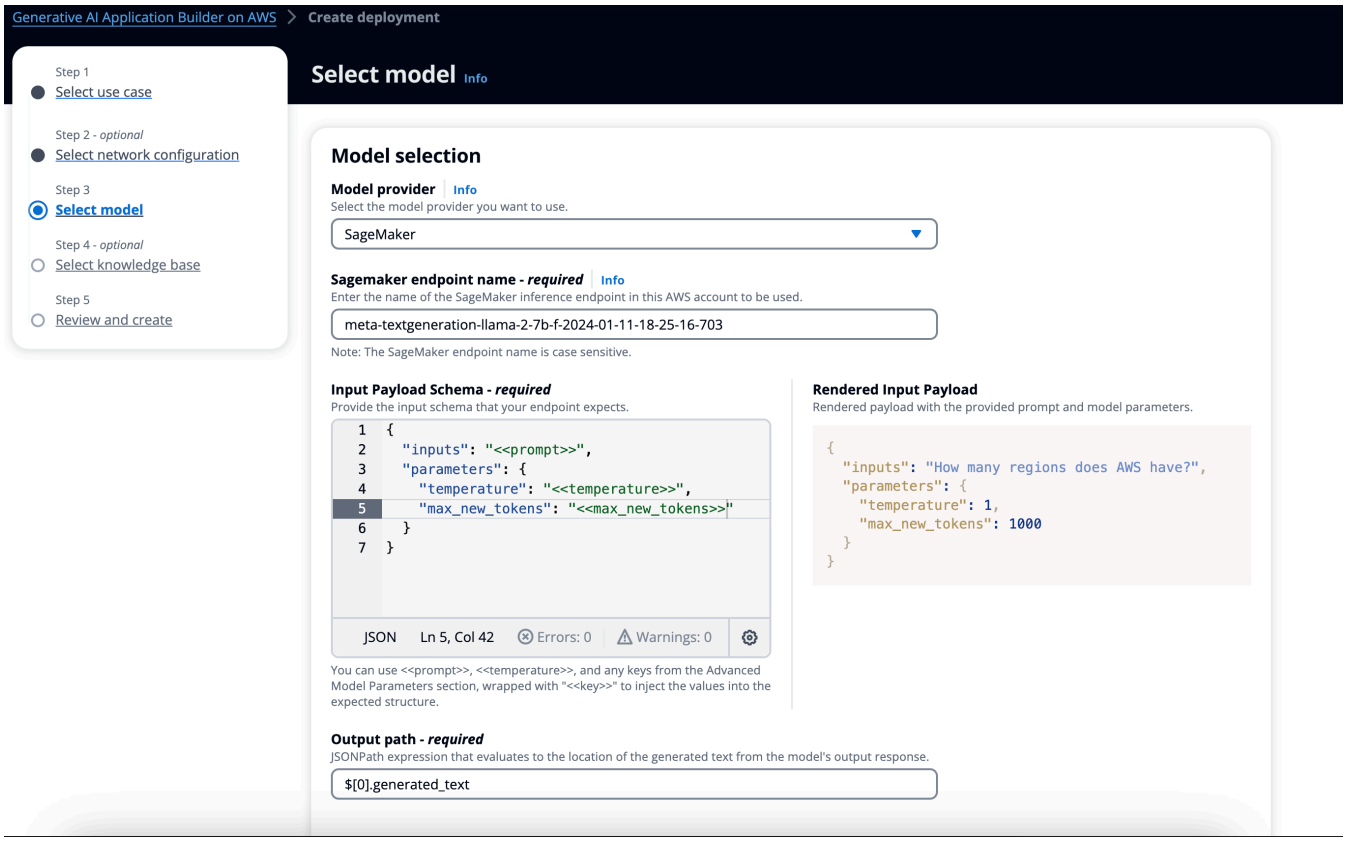
     AWS コンソール上でエンドポイント名が表示される場所

   - エンドポイントによって期待される入力ペイロードのスキーマ。最も広範なエンドポイントをサポートするには、管理者ユーザーは、エンドポイントが期待する入力の形式をソリューションに示す必要があります。モデルの選択ウィザードで、ソリューションがエンドポイントに送信する JSON スキーマを指定します。リクエストペイロードに静的値と動的値を挿入するためのプレースホルダーを追加できます。次のオプションを使用できます。
     - 必須プレースホルダー: \<\<prompt\>\> は、ランタイム時に SageMaker AI エンドポイントに送信される完全な入力 (例えば、プロンプトテンプレートからの履歴、コンテキスト、ユーザー入力など) に動的に置き換えられます。

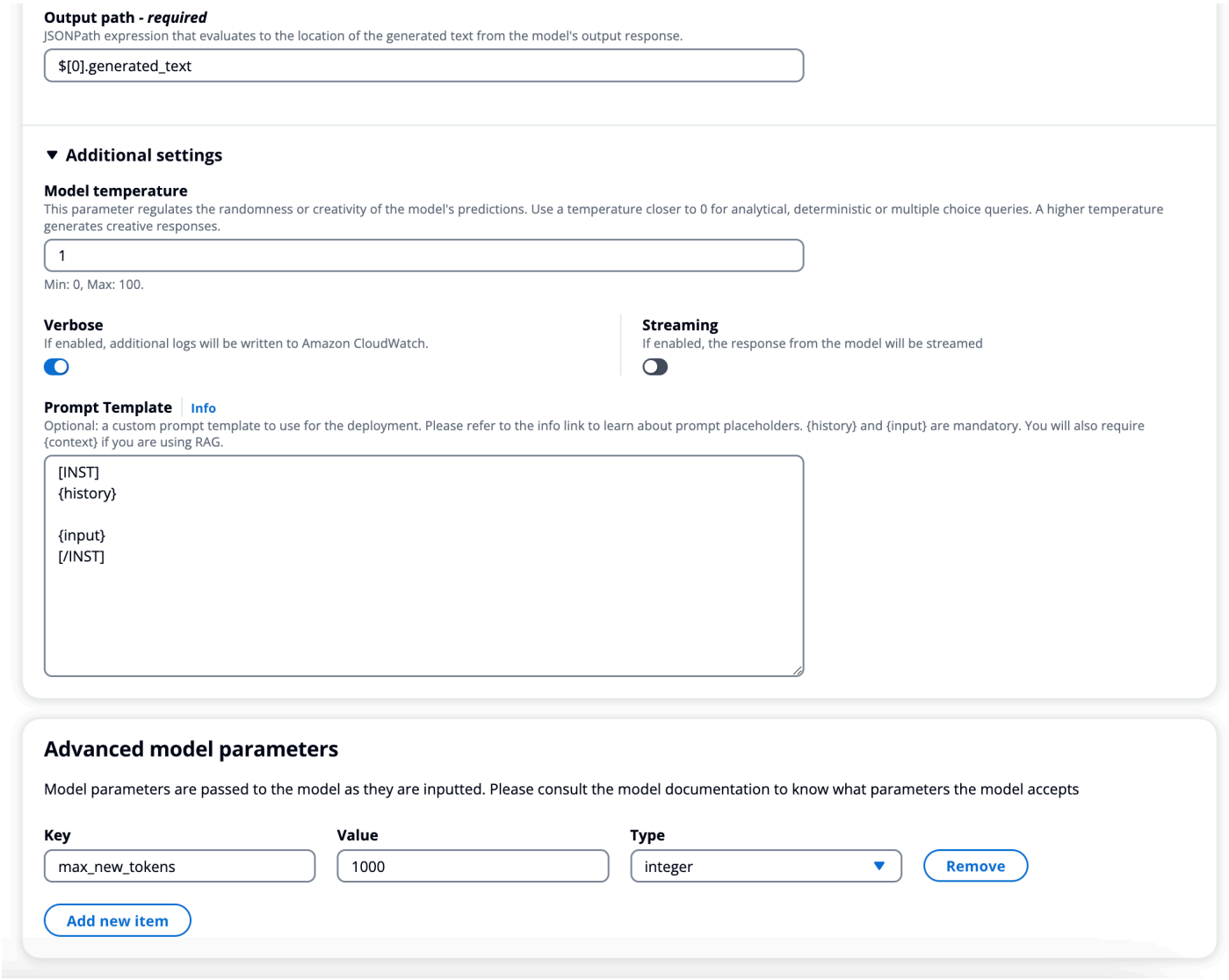
- オプションのプレースホルダー: \<\<temperature\>\>*,* 、および詳細モデルパラメータで定義された任意のパラメータをエンドポイントに提供できます。\<\< と \>\> で囲まれたプレースホルダーを含む文字列 (例: \<\<max_new_tokens\>\>) は、同じ名前の詳細モデルパラメータの値に置き換えられます。

  入力スキーマの例 - 必須フィールド、プロンプト、温度の設定、およびカスタム詳細パラメータ max_new_tokens の設定。出力パスは有効な JSONPath 文字列として指定する必要がある



3. LLM が生成した文字列応答の出力ペイロード内の場所。これを JSONPath 式として指定して、ユーザーに表示される最終的なテキスト応答へのアクセスが、エンドポイントの戻りオブジェクトと応答内のどこから期待されるかを示す必要があります。

   SageMaker AI 入力スキーマ内で使用する詳細モデルパラメータの追加例 (以前のオプション/設定については、図 2 を参照)



> **Note**
>
> SageMaker AI では、同じエンドポイントの背後で複数のモデルをホストできるようになりました。これは、現行バージョンの SageMaker AI Studio (Studio Classic ではなく) にエンドポイントをデプロイするときのデフォルト設定です。
> エンドポイントがこのように設定されている場合は、詳細モデルパラメーターセクションに InferenceComponentName を追加して、使用するモデルの名前に対応する値を指定する必要があります。

高度な LLM の設定

Amazon Bedrock を使用する場合、Amazon Bedrock ガードレール、Amazon Bedrock プロビジョンドスループット、その他のモデルパラメータなど、モデルの高度な設定を行うことができます。

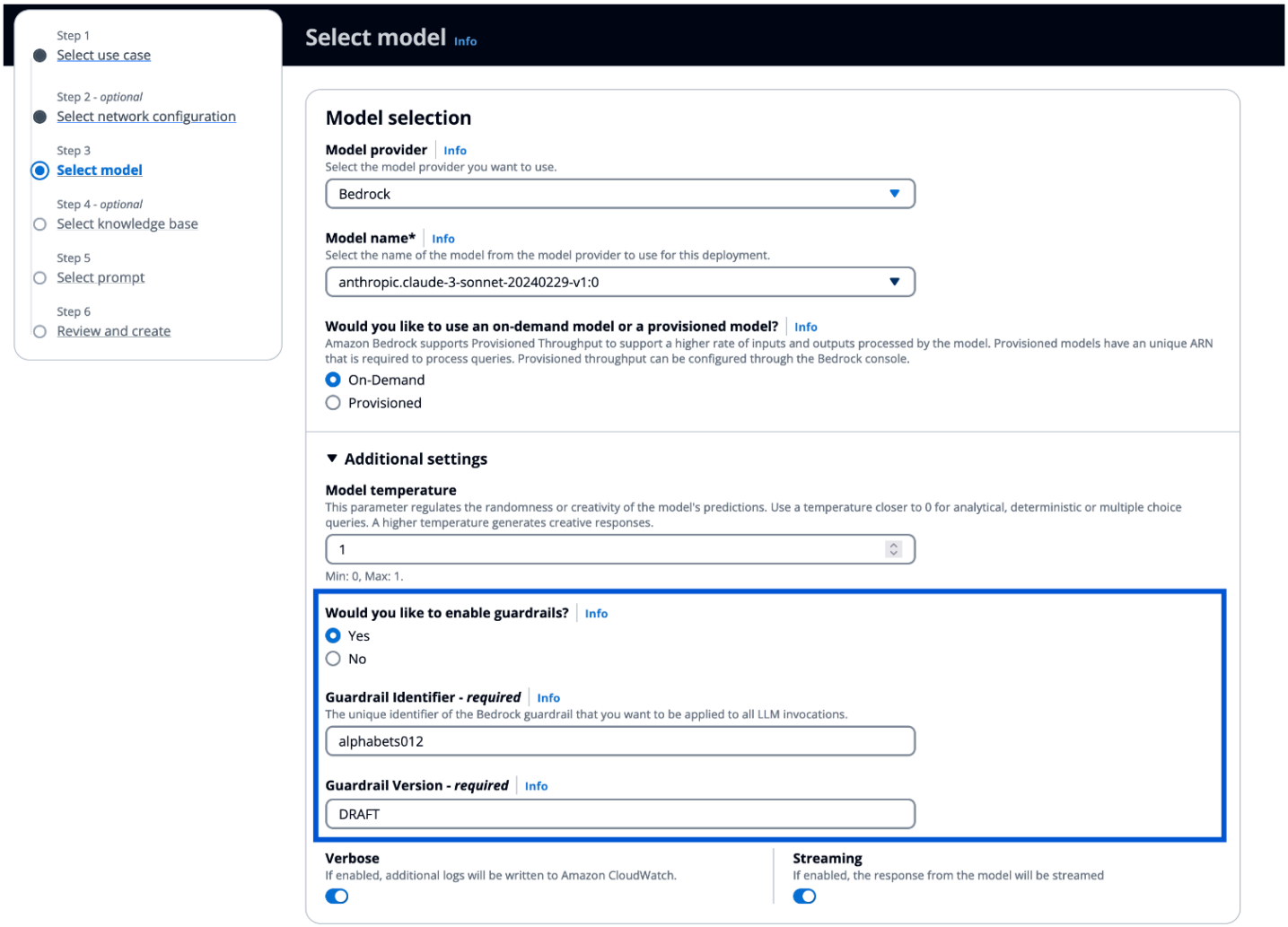
Amazon Bedrock ガードレール

Amazon Bedrock ガードレールは、Amazon Bedrock の機能の 1 つです。ユーザー設定のポリシーに基づいてユーザー入力と LLM の応答を評価し、ユーザーがユースケースに選択した基盤となる LLM の種類を問わず、追加の保護レイヤーを提供します。ガードレールは、望ましくないカテゴリまたは有害なカテゴリに分類されるコンテンツを回避するための 2 つのポリシーで構成されています。

1. 拒否されたトピックは、金融アプリケーションでの投資アドバイスなど、ユーザーのアプリケーションのコンテキストとして望ましくないトピックのセットを定義します。
2. コンテンツフィルター**** 有害なコンテンツを含む入力ユーザープロンプトまたはモデルの応答をフィルタリングできます。

生成 AI アプリケーションビルダーソリューションで使用するには、[ガードレールを作成] ウィザードを使用して、Amazon Bedrock コンソールでガードレールを設定する必要があります。作成したら、ガードレール識別子とガードレールバージョンを指定して、モデルの選択ステップの [その他の設定] で、生成 AI アプリケーションビルダーソリューションウィザードを介して作成したチャットユースケースに、このガードレールを追加できます。

デプロイウィザードの説明図 - Amazon Bedrock ガードレールを有効にする



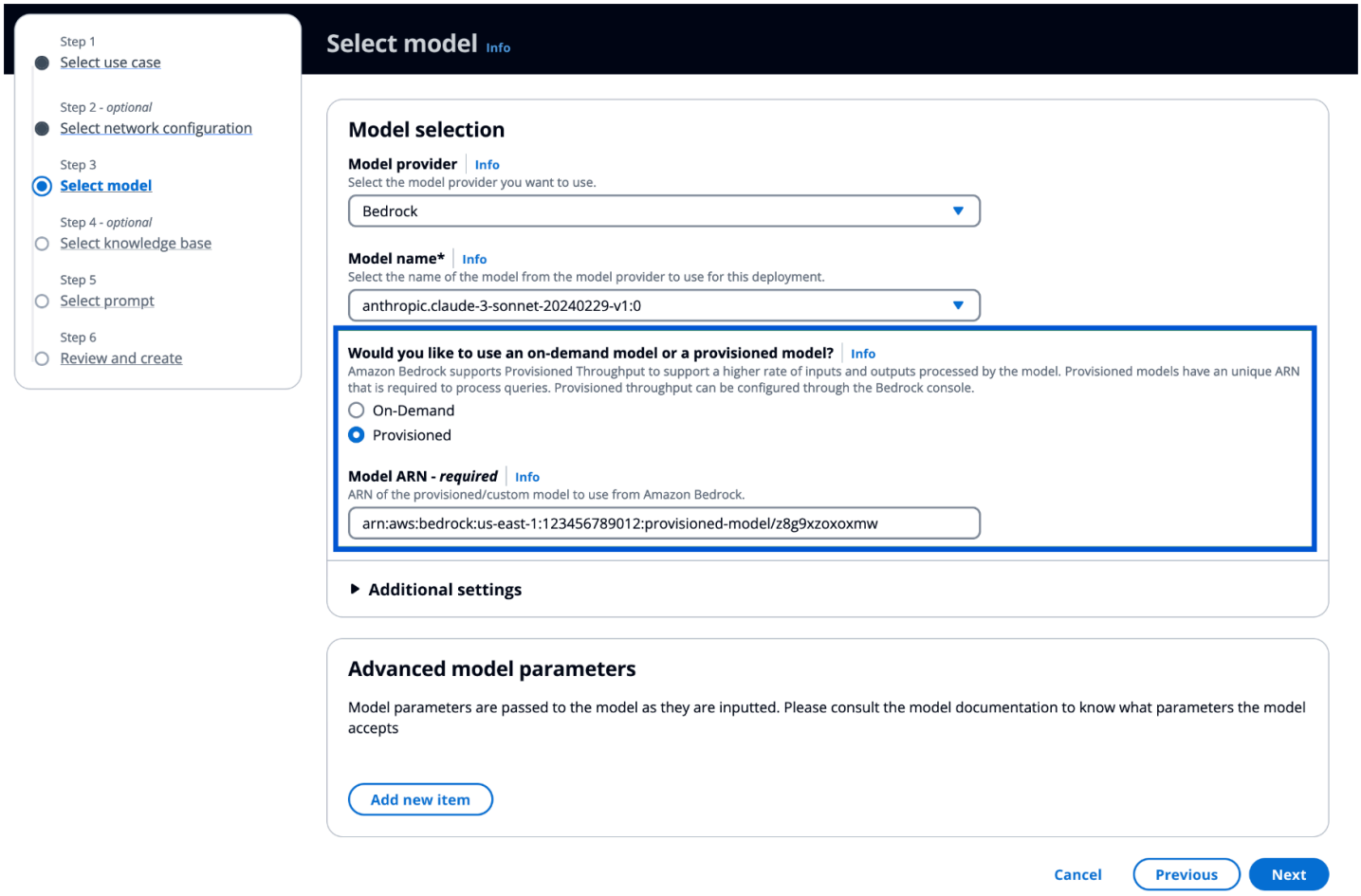
Amazon Bedrock のプロビジョンドスループット

各オンデマンド Amazon Bedrock モデルでは、モデル推論について、リージョン固有のアカウントクォータ制限に従います。例えば、Bedrock で Anthropic Claude 2.x を使用する場合、現時点では us-east-1 リージョンと us-west-2 リージョンでは、1 分あたり 500 件のリクエストと 500,000 件のトークンの処理が許可されています。このソリューションは、ファインチューニング済みモデルや継続的な事前トレーニングモデルで使用することもできます。このようなインスタンスの場合、Amazon Bedrock でプロビジョンドスループットが許可され、本番環境のアプリケーションで使用できるように、ベースモデル、ファインチューニング済みモデル、または継続的な事前トレーニングモデルに対して、大規模かつ整合性を維持した推論ワークロードを実行できます。

Amazon Bedrock コンソールでプロビジョンドスループットを購入したら、モデル ARN が生成されて使用できるようになります。このモデル ARN は、モデルの選択ステップの生成 AI アプリケーションビルダーウィザードで指定できるようになりました。これを実行するには、モデルプロバイ

ダーとして Bedrock を選択し、Amazon Bedrock コンソールでこのプロビジョンドモデル ARN を生成するために使用されたベースモデル名を選択します。次に、オンデマンドモデルとプロビジョンドモデルのどちらかを選択する際に「プロビジョンドモデル」を選択して、使用するモデル ARN を指定します。

デプロイウィザードの説明図 - Amazon Bedrock のプロビジョンドスループットを有効にする



> **Note**
>
> ガードレールとプロビジョンドスループットは、デプロイしたデプロイダッシュボードとユースケーススタックと同じリージョンに配置する必要があります。

モデルパラメータ

LLM では多くの場合、その実装に固有の幅広いパラメータを使用できます。モデルプロバイダーは一般的に、サポートされるパラメータのセットとその使用方法を概説したドキュメントを提供しています。

このソリューションはモデルパラメータを基盤モデルに直接渡すため、パラメータが正しく設定されていることを確認することが重要です。サポートされるパラメータの最新情報については、モデルプロバイダーのドキュメントを参照してください。

エージェントビルダーの設定

エージェントビルダーには、本番稼働対応の AI エージェントを作成するための包括的な設定オプションが用意されています。このセクションでは、エージェントビルダーのデプロイを設定および管理する方法を説明します。

システムプロンプトの設定

システムプロンプトは、エージェントの動作、パーソナリティ、機能を定義します。システムプロンプトを設定するには:

1. エージェントビルダーウィザードで、[エージェントを設定] ステップに移動します。
2. テキストエディタでシステムプロンプトテンプレートを編集します。
3. 以下の明確な手順を含めます。
   - エージェントの役割と目的
   - 使用可能なツールの使用方法 (MCP サーバー)
   - レスポンスのフォーマット設定
   - 動作ガイドライン
4. 必要に応じて、[デフォルトにリセット] ボタンを使用して元のテンプレートを復元します。

エージェントプロンプトのベストプラクティス:

- エージェントの機能と制限について具体的に説明する
- 望ましい動作の明確な例を提供する
- ツールの使用方法と呼び出すタイミングに関する手順を含める
- レスポンス形式の期待値を定義する

- エージェント動作の境界を設定する

MCP サーバー統合

モデルコンテキストプロトコル (MCP) サーバーは、エージェントにエンタープライズツールとデータソースへのアクセスを提供します。MCP サーバーを設定するには:

1. [エージェントを設定] ステップで、[MCP サーバー] セクションを見つけます。
2. ドロップダウンメニューから使用可能な MCP サーバーを選択します。

> **Note**
>
> エージェントをデプロイする前に、MCP サーバーを設定してアクセス可能にする必要があります。エージェントは、設定された MCP サーバーによって公開されているツールを自動的に検出して使用します。サーバーのセットアップとツールの設定については、MCP のドキュメントを参照してください。

[メモリの設定]

エージェントビルダーには、コンテキストと知識を維持するために 2 種類のメモリが用意されています。

短期メモリ

すべてのエージェントに対してデフォルトで有効:

- セッション内の会話コンテキストを維持する
- ユーザーメッセージとエージェントのレスポンスを自動的にキャプチャします
- 適切な分離のために actorId と sessionId で整理されています
- 設定は不要です

長期メモリ

セッション間でインサイトを保存するためのオプション機能:

1. [エージェントを設定] ステップで、[メモリ設定] セクションを見つけます。

2. [長期メモリを有効にする] トグルをオンにして有効化します。
3. 有効にすると、エージェントは次のことを実行できます。
   - 会話全体から重要な情報を抽出して保存する
   - 以前のセッションから関連するコンテキストを取得する
   - ユーザーの好みや履歴に関する知識を構築する

> **Note**
>
> 長期メモリでは、セマンティックメモリ戦略とデフォルトの保持設定を備えた AgentCore Memory が使用されます。

エージェントビルダーのデプロイのモニタリング

エージェントビルダーは、CloudWatch ダッシュボードとメトリクスを通じて包括的なモニタリングを提供します。

CloudWatch ダッシュボードへのアクセス

1. AWS アカウントの CloudWatch コンソールに移動します。
2. 左側のナビゲーションから [ダッシュボード] を選択します。
3. `AgentBuilder-<UseCaseId>` という名前のダッシュボードを見つけます。
4. リアルタイムのメトリクスと過去のパフォーマンスデータを表示します。

ログへのアクセスと分析

エージェントログは CloudWatch Logs で利用できます。

1. AWS コンソールで CloudWatch Logs に移動します。
2. プレフィックスが `/aws/bedrock-agentcore/runtimes/` のロググループを検索します。
3. CloudWatch Logs Insights を使用してログをクエリおよび分析します。
4. 特定のリクエスト ID またはエラーパターンを検索します。

ワークフロービルダーの設定

ワークフロービルダーは、作業を専門のエージェントビルダーのエージェントに委任するスーパーバイザーエージェントを介してマルチエージェントオーケストレーションを可能にします。

ワークフローの作成

1. デプロイダッシュボードに移動します
2. [ワークフローユースケースの作成] を選択します
3. スーパーバイザーエージェントを設定します。
   - 名前: ワークフローの説明名
   - 説明: 目的と機能
   - システムプロンプト: エージェントの委任と調整に関する指示
   - モデル: スーパーバイザーエージェントの基盤モデル

スーパーバイザープロンプトのベストプラクティス:

- 各専門エージェントをいつ使用するかを明確に記述する
- 複数のエージェントからの結果を集約するための手順を含める
- レスポンス形式の期待値を定義する
- 委任動作の境界を設定する

エージェントの選択

専門エージェントとして含めるエージェントビルダーのエージェントを選択します。

1. ワークフロー設定で [エージェントを追加] をクリックします
2. 使用可能なエージェントビルダーのエージェントを参照または検索します
3. エージェントの説明を確認します
4. ワークフローに含めるエージェントを選択します

エージェントの説明

スーパーバイザーエージェントは、エージェントの説明を使用して、委任するエージェントを決定します。説明では次の点を明確に説明します。

- エージェントの専門分野または能力
- エージェントが処理するタスクの種類
- 入出力の期待値

ワークフローのテスト

デプロイ後:

1. デプロイダッシュボードからワークフローにアクセスします
2. 複数のエージェントを必要とするクエリでテストします
3. CloudWatch Logs でエージェントの委任をモニタリングします
4. レスポンスの品質と委任パターンを確認します
5. 委任が最適でない場合はスーパーバイザーのプロンプトを調整します

モデルトークンの制限を管理するためのヒント

注記: このソリューションでは、さまざまな LLM によるトークン制限の直接的な管理は行いません。プロンプトをテストして、モデルプロバイダーによって適用される制限の範囲内であることを確認してください。

プロンプトのサイズを管理するには、次の方法を試してください。

1. 使用したいモデルでの制限をよく理解しておきます。これらの値はモデルによって大きく異なる可能性があるため、始める前に利用可能な予算を把握しておくことが重要です。
2. その予算を念頭に置いて最初のプロンプトを作成し、プロンプトの動的な要素のためにどれだけ確保すべきかを検討してください。例えば、ユーザー入力、チャット履歴、ドキュメントの抜粋などがあります。
3. プロンプト設定ページで、[Size of trailing history] の制限を設定して、プロンプトに含まれる会話ターンの数を制限します。
4. ナレッジベース設定ウィザードでドキュメントの検索結果制限を設定します。タスクの実行に十分なコンテキストを LLM に提供する一方で、トークンの制限を超えたり、レイテンシーに悪影響を及ぼしたりしないよう、適切なバランスをとる必要があります。
5. いくらかバッファーを設けておきます。一般的なケースに予算を組むのではなく、長い入力クエリ、大きなドキュメントの抜粋、長い会話などのエッジケースを考えて実験してください。

MCP サーバーの Docker イメージを構築するステップ

AWS での生成 AI アプリケーションビルダーで MCP (モデルコンテキストプロトコル) サーバーを使用するには、最初のステップとしてプライベート Amazon ECR リポジトリに構築され保存された Docker イメージが必要です。

> **Note**
>
> 現時点では、Amazon Bedrock AgentCore Runtime にデプロイされた既存の MCP サーバーを GAAB にエクスポートすることはできません。GAAB で作成されたエージェントに MCP サーバーをアタッチするには、そのエージェントを GAAB で作成する必要があります。

ステップ 1: MCP サーバーを作成する

まず、MCP サーバーの実装を準備する必要があります。MCP サーバーを作成する詳細な手順については、「Amazon Bedrock AgentCore デベロッパーガイド - MCP サーバーの作成」を参照してください。

次のようなプロジェクト構造をお勧めします。

```
.
### __init__.py
### extras/
#   ### extra_dependencies.py
#   ### Dockerfile
### requirements.txt
### server.py <-- Server Entry point
```

Dockerfile の構造については、次の例のような形式を使用することをお勧めします。

```
FROM ghcr.io/astral-sh/uv:python3.13-bookworm-slim
WORKDIR /app

# All environment variables in one layer
ENV UV_SYSTEM_PYTHON=1 \
    UV_COMPILE_BYTECODE=1 \
    UV_NO_PROGRESS=1 \
    PYTHONUNBUFFERED=1 \
```

```
    DOCKER_CONTAINER=1 \
    AWS_REGION=us-east-1 \
    AWS_DEFAULT_REGION=us-east-1

COPY requirements.txt requirements.txt
# Install from requirements file
RUN uv pip install -r requirements.txt

RUN uv pip install aws-opentelemetry-distro>=0.10.1

# Signal that this is running in Docker for host binding logic
ENV DOCKER_CONTAINER=1

# Create non-root user
RUN useradd -m -u 1000 bedrock_agentcore
USER bedrock_agentcore

EXPOSE 9000
EXPOSE 8000
EXPOSE 8080

# Copy entire project (respecting .dockerignore)
COPY . .

# Use the full module path
CMD ["opentelemetry-instrument", "python", "-m", "server"]
```

ステップ 2: MCP サーバーをローカルでテストする

AWS にデプロイする前に、ローカルで MCP サーバーをテストして、期待どおりに動作することを確認することが重要です。ローカルテストの詳細については、「Amazon Bedrock AgentCore デベロッパーガイド - MCP サーバーをローカルでテストする」を参照してください。

ステップ 3: Amazon ECR にデプロイする

MCP サーバーを作成してローカルでテストしたら、以下の手順に従って Amazon ECR にデプロイします。

1. 最新バージョンの AWS CLI と Docker がインストールされていることを確認してください。詳細については、「Amazon ECR を使い始める」を参照してください。
2. 認証トークンを取得し、Docker クライアントをレジストリに認証します。AWS CLI を使用する:

```
aws ecr get-login-password --region us-east-1 | docker login --username AWS --
password-stdin <account-id>.dkr.ecr.us-east-1.amazonaws.com
```

3. 以下のコマンドを使用して、Docker イメージを構築します。Docker ファイルをゼロから構築する方法については、「Docker のドキュメント」を参照してください。イメージが既に構築されている場合は、このステップをスキップできます。

```
docker build -t <repository-name> .
```

4. ビルドが完了したら、イメージにタグを付けて、イメージをこのリポジトリにプッシュできるようにします。

```
docker tag <repository-name>:latest <account-id>.dkr.ecr.us-east-1.amazonaws.com/
<repository-name>:latest
```

5. 次のコマンドを実行して、このイメージを新しく作成した AWS リポジトリにプッシュします。

```
docker push <account-id>.dkr.ecr.us-east-1.amazonaws.com/<repository-name>:latest
```

完全なデプロイ手順については、「Amazon Bedrock AgentCore デベロッパーガイド - MCP サーバーを AWS にデプロイする」を参照してください。

ステップ 4: GAAB で ECR URI を使用する

Docker イメージを Amazon ECR に正常にプッシュしたら、ECR コンソールからイメージ URI をコピーします。この URI は、AWS での生成 AI アプリケーションビルダーデプロイウィザードを使用して MCP サーバーをデプロイするときに使用します。

異なる MCP ゲートウェイターゲットを作成する手順

Amazon Bedrock AgentCore Gateway を使用すると、既存の AWS サービスと API を、エージェントが使用できる MCP ツールに変換できます。Gateway は複数のターゲットタイプをサポートしているため、さまざまなバックエンドサービスをシームレスに統合できます。

以下のターゲットタイプがサポートされています。

- Lambda ターゲット: AWS Lambda 関数を MCP ツールに変換します。詳細な手順については、「Amazon Bedrock AgentCore デベロッパーガイド - Lambda ターゲットの追加」を参照してください。
- OpenAPI ターゲット: OpenAPI 仕様を使用して、REST API を MCP ツールとして定義して公開します。詳細な手順については、「Amazon Bedrock AgentCore デベロッパーガイド - OpenAPI スキーマ」を参照してください。
- Smithy ターゲット: タイプセーフな API 統合のために Smithy モデル定義を使用して MCP ツールを構築します。詳細な手順については、「Amazon Bedrock AgentCore デベロッパーガイド - Smithy ターゲットの構築」を参照してください。
- MCP サーバーターゲット: URL エンドポイントを介して外部 MCP サーバーに直接接続し、既存の MCP サーバーを統合できます。詳細な手順については、「Amazon Bedrock AgentCore デベロッパーガイド - MCP サーバーターゲット」を参照してください。

MCP Gateway ターゲットの作成に関するその他の例とチュートリアルについては、Amazon Bedrock AgentCore サンプルリポジトリを参照してください。

ナレッジベースの設定

このセクションでは、ソリューションのために選択したナレッジベースにデータを取り込む方法について説明します。このソリューションは現在、RAG ベースのユースケースデプロイのナレッジベースとして Amazon Kendra と Amazon Bedrock ナレッジベースをサポートしています。

Amazon Kendra

Amazon Kendra をナレッジベースとして使用する場合は、さまざまなデータソースコネクタを使用して多様なソースからデータを取り込む方法について、「Amazon Kendra デベロッパーガイド」を参照してくさい。

重要: 偶発的なデータ損失を防ぐため、このソリューションではデプロイまたはスタックが削除されても (このソリューションが作成したかどうかにかかわらず)、Kendra インデックスが自動的に削除されることはありません。ナレッジベースを削除してコストが発生しないようにするには、「手動アンインストール」セクションで、保持されるリソースとそのクリーンアップ方法の詳細を確認してください。

Amazon Bedrock ナレッジベース

Amazon Bedrock ナレッジベースは、さまざまなベクトルストアをサポートでき、それぞれにデータのインデックス作成機能を利用できます。ナレッジベースを設定して入力するには、「Amazon Bedrock ユーザーガイド」を参照してください。具体的には、以下を実行します。

- まず、データソースをセットアップします。
- 次に、サポートされているベクトルストアでナレッジベースのベクトルインデックスを設定します。ナレッジベースの作成中に Bedrock コンソールで「新しいベクトルストアをクイック作成」オプションを使用する場合は、このステップをスキップできます。
- 最後に、ナレッジベースを作成して、設定したデータソースを同期します。

高度なナレッジベースの設定

このソリューションでは、ナレッジベースのフィルタリングやロールベースのアクセスコントロールを使用した RAG などの高度なナレッジベース設定を使用できます。ナレッジベースのフィルタリングは、どちらのナレッジベースでも適用できます。一方、ロールベースのアクセスコントロールを使用した RAG は Amazon Kendra で利用できます。

ナレッジベースのフィルタリング

このソリューションでは、ウィザードのナレッジベースステップの [Advanced RAG configurations] セクションでユースケースをデプロイする際に、[Amazon Kendra attribute filters] または [Bedrock knowledge base retrieval filters]を指定できます。これらのフィルターを使用すると、検索戦略、クエリ対象の基盤となるドキュメントの言語など、ナレッジベースのデータソースのクエリ方法を定義できます。

いずれの場合も、JSON オブジェクトを使用して、各サービスドキュメント (上記のリンク参照) で指定した形式に従ってフィルター設定を指定します。

例 1: Kendra AttributeFilter

```
{
 "EqualsTo": {
 "Key": "_language_code",
 "Value": {
 "StringValue": "es"
 }
 }
}
```

例 2: Bedrock RetrievalFilter

```
{
 "equals": {
 "key": "language",
 "value": "es"
 }
}
```

Amazon Kendra によるロールベースのアクセスコントロールを備えた RAG

ロールベースアクセスコントロール (RBAC) を使用すると、Amazon Kendra インデックス内の特定のドキュメントにアクセスしたり、検索結果で特定のドキュメントを表示したりできるユーザーまたはグループを管理できます。AWS での生成 AI アプリケーションビルダー (GAAB) のユースケースを使用して Amazon Kendra インデックス ID の RBAC を設定するには、次の手順を実行します。

1. Amazon Kendra インデックスを設定する

1. Amazon Kendra インデックスが作成済みで、少なくとも 1 つのデータソースが追加されていることを確認します。
2. ユーザーグループに基づいてデータソースのアクセスコントロールを設定します。S3 データソースの場合は、ドキュメントの手順に従って、Amazon Cognito ユーザープールで作成したグループ名と同じグループ名を使用してアクセスコントロールリスト (ACL) を設定します。これにより、ユーザーはグループメンバーシップに基づいて表示権限が付与されたドキュメントと検索結果にのみアクセスできるようになります。

> **Note**
>
> 作成した Kendra インデックスの [ユーザーアクセスコントロール] で、[トークンベースのユーザーアクセスコントロール] を [いいえ] のままにします。ステップ 2 で [ロールベースのアクセスコントロール] を有効にすると、AWS での生成 AI アプリケーションビルダーはユーザー認証トークンから適切なクレームを抽出し、属性フィルターを作成します。

2. GAAB デプロイウィザードを使用して RAG ユースケースをデプロイする

1. GAAB デプロイウィザードの画面に表示されるウィザードの指示に従ってウィザードのステップ 4 まで進み、RAG を設定します。
2. デプロイウィザードの [Select Knowledge Base] ステップで、ナレッジベースタイプとして [Amazon Kendra] を選択します。
3. 既存の Amazon Kendra インデックスを使用するか、新しい Index を作成するかを指定します。既存のインデックスがある場合は、ユーザーグループに基づいてアクセスコントロールリスト (ACL) を設定済みの Amazon Kendra インデックスの ID を指定します。
4. [Role Based Access Control] オプションを有効にします。このオプションを使用することで、Amazon Kendra インデックスから返される検索結果が、ユーザーのロールとグループのアクセス許可に基づいてフィルタリングされるようになります。
5. ユースケースを確認してデプロイします。

3. Amazon Cognito を設定する

1. GAAB デプロイで使用する Amazon Cognito ユーザープールを検索します。この Amazon Cognito ユーザープールは通常、メインデプロイダッシュボードの CloudFormation スタックが作成したものです。
2. Amazon Cognito ユーザープールに新しいユーザーを作成します。ユーザーを作成する際は、[Send an email invitation] オプションを選択して、ユーザーが E メールで一時的なログイン認証情報を受け取るようにします。これにより、新しいユーザーがサインアップして GAAB アプリケーションにアクセスできるようになります。
3. Amazon Cognito ユーザープールにユーザーグループを作成します。グループ名が Amazon Kendra インデックス ACL で設定されているグループと完全に一致していることを確認します。ユーザーがアクセスできる検索結果はグループのメンバーシップによって決まるため、これは、RBAC を有効にする上で非常に重要です。
4. ユーザーのロールとアクセス許可に基づいて、ユーザーを適切なグループに割り当てます。ユーザーは、Amazon Kendra インデックス ACL に必要なグループと、GAAB デプロイ中に作成されたユースケース固有のグループの両方に追加する必要があります。これにより、特定のユースケースと関連する検索結果にアクセスするために必要なアクセス許可がユーザーに付与されます。

このような手順を実行すると、GAAB デプロイにロールベースのアクセスコントロール (RBAC) を設定でき、ユーザーは割り当てられたユーザーグループとアクセス許可に基づいて、承認済みの情報と機能のみにアクセスして操作できるようになります。

> **Note**
>
> 現時点では、AWS での生成 AI アプリケーションビルダーでナレッジベースの RBAC をサポートしているのは Amazon Kendra のみです。Amazon Bedrock ナレッジベースでは RBAC はサポートされていませんが、メタデータフィルターを使用してある程度のフィルタリングを行うことができます。詳細については、「Amazon Bedrock ユーザーガイド」を参照してください。

プロンプトの設定

デプロイダッシュボードウィザードには、プロンプト設定ステップが提供されており、ユーザーと AI モデル間のインタラクションをガイドするプロンプトエクスペリエンスとテンプレートをカスタマイズできます。AI アシスタントから正確かつ関連性の高い応答を得るために、これらの設定を適切に指定することが不可欠です。

このセクションでは、AI プロンプトの全体的なエクスペリエンスと動作を制御します。

- Max prompt template length: この設定により、プロンプトテンプレートの最大長 (文字単位) が決まります。値を大きくすると、AI モデルに提供されるコンテキストが増大し、応答の精度向上につながる可能性があります。ただし、プロンプトが長すぎるとノイズが発生し、パフォーマンスに悪影響を及ぼす可能性があります。Amazon Bedrock モデルの場合、プロンプトテンプレートの最大長 (文字単位) のデフォルト値は、基盤となるモデルのトークン制限を使用して計算されます。Bedrock 内でモデル名を編集したり変更したりすると、[Reset to default] ボタンが強調表示されて、新しく選択したモデルのデフォルトを採用するために使用できます。Amazon SageMaker AI モデルの場合、適切なデフォルト値が提供されるとはいえ、基盤となるモデルを確認して、それに応じてプロンプトテンプレートの最大長と入力テキストの長さを選択することをお勧めします。詳細については、「モデルトークンの制限を管理するためのヒント」セクションを参照してください。
- Max input text length: この設定は、ユーザーが入力するテキストの最大長 (文字数) を制限します。入力テキストが長すぎると、無関係な情報が含まれる可能性があり、AIモデルから無関係な応答や不正確な応答が返されるリスクが増大します。
- User Prompt Editing: このオプションを使用すると、ユーザーがチャット UI を使用してプロンプトテンプレートを変更したり、無効にしたりできるようになります。この機能を無効にすると、整合性を維持して、プロンプトへの意図しない変更を回避できます。

Prompt template

このセクションでは、AI モデルで使用する実際のプロンプトテンプレートを定義します。プロンプトテンプレートは通常、ユーザーの入力、リファレンスする文章、チャット履歴など、さまざまなコンポーネントのプレースホルダーを含む構造に従います。

- Prompt template: これは、必要なプロンプトテンプレートを入力したり貼り付けたりすることができるメインテキスト領域です。テンプレートは、AI モデルに必要なコンテキストと指示を提供するように作成する必要があります。通常、以下のプレースホルダーが含まれます。
  - `{input}`: このプレースホルダーは Sagemaker AI デプロイでは必須で、ユーザーの入力またはクエリに置き換えられます。
  - `{history}`: このプレースホルダーは Sagemaker AI デプロイでは必須で、現在の会話のチャット履歴に置き換えられます。
  - `{context}`: このプレースホルダーは RAG デプロイに必須であり、設定したナレッジベースから取得したドキュメントの抜粋に置き換えられます。
- Rephrase Question?: (RAG デプロイでのみ利用可能な) このオプション を使用して、AI モデルに渡される前に、ユーザーの元の入力クエリを言い換えるか、曖昧さを解消するかを指定します。クエリを言い換えることで、モデルがユーザーの意図をよりよく理解し、より正確な応答につながる可能性があります。

プロンプトテンプレートとエクスペリエンスを設定する際は、AI モデルに十分なコンテキストと指示を提供すると同時に、ノイズやパフォーマンスの問題を引き起こす可能性のある、長すぎる情報や無関係な情報は回避するというバランスを取ることが重要です。

Advanced prompt settings

このセクションでは、会話履歴を AI モデルに提供する方法を制御します。

- Size of trailing history: この設定により、最終敵にプロンプトに含める以前のメッセージの数が決定します。この値をゼロに設定すると、プロンプトテンプレートまたは曖昧さを回避するプロンプトテンプレートに履歴は挿入されません。注意: ゼロに設定する場合でも、{history} プレースホルダーはプロンプトテンプレートに残す必要があります。これはランタイムで空の文字列に置き換えられます。
  - 注: この値には、偶数を指定することをお勧めします。奇数を指定すると、ペアになっているインタラクションの AI 応答のみが返されます。
- Human Prefix: これは、会話履歴でユーザーが送信したメッセージを識別するために使用されるプレフィックスです。

- AI Prefix: これは、会話履歴で AI モデルが返したメッセージを識別するために使用されるプレフィックスです。

Disambiguation Prompt Configuration

このセクションでは、設定したナレッジベースに送信する前に、ユーザー入力の曖昧さを解消するための動作とテンプレートを設定できます。

- Enable Disambiguation: このオプションは、ナレッジベースに送信する前にユーザー入力の曖昧さを解消するかどうかを決定します。
- Disambiguation Prompt Template: これは、ナレッジベースに接続する際にユーザー入力の曖昧さ解消に使用されるプロンプトテンプレートです。このプロンプトから生成された出力は、ナレッジベースに送信されるクエリとして使用されます。曖昧さ解消を無効にすると、ユーザーの raw クエリは変更されずにナレッジベースに送信されます。

例えば、曖昧さの解消を有効にすると、「コストはどの程度ですか」というフォローアップのユーザークエリは、「ナンバープレートを更新するにはコストはどの程度になりますか」というように曖昧さが解消され、検索クエリが改善されます。

デプロイされた Text ユースケースを使用する

Text ユースケース用の組み込み UI は、管理者ユーザーが作成したデプロイをビジネスユーザーがすばやく調べて実験できるようにすることを目的としています。ビジネスユーザーが行った設定変更は、そのビジネスユーザーのセッションでのみ有効になります。ビジネスユーザーは、これらの変更を管理者ユーザーと共有する必要があります。管理者ユーザーは、これらの変更を使用して基本デプロイを更新し、すべてのユーザーが使用できるようにします。

チャット UI のコンポーネントは次のとおりです。

- チャットウィンドウ
- チャット入力ボックス
- 設定
- 会話をクリア

チャットウィンドウ

会話のさまざまなターンを保持します。右側で始まるメッセージはビジネスユーザーからのもので、左側で始まるメッセージは設定された LLM からのものです。すべての LLM の回答には小さなクリップボードアイコンがあり、回答を簡単にコピーできます。

チャット入力ボックス

チャット入力ボックスは、チャットウィンドウの下部に固定されています。これはビジネスユーザーが LLM に送信するメッセージを入力できる場所です。入力ボックスのすぐ上には接続ステータスが表示されます。接続が失われた場合は (操作がない場合など)、次回チャットメッセージが送信されるときに、新しい接続が自動的に作成されます。追加の WebSocket 接続時間が発生するため、このリクエストにはもう少し時間がかかると予想されます。

特定の設定によっては、入力に最大長が適用される場合があります。この制限を超えると、ユーザーは警告を受け取り、メッセージは送信されません。

Amazon Kendra で RAG を使用する場合、Retrieve API はクエリを 30 トークンワードに切り捨てます。ユーザー入力がそれよりも長くなることが予想される場合は、これが検索パフォーマンスにどのように影響するか評価してください。

設定

ビジネスユーザーがさまざまな設定をすばやく実験できるように、特定のデプロイ設定オプション (プロンプトテンプレートなど) を即座に編集できる設定パネルが

用意されています。これらの変更は、新しいセッションの開始時にのみ行うことができます。会話の開始後は、会話をクリアすると再び構成設定を編集できるようになります。

注: 管理者ユーザーは、デプロイの設定をロックできます。プロンプトステップ中にウィザードを使用して、デプロイ時のライブ編集を回避できます。

会話をクリア

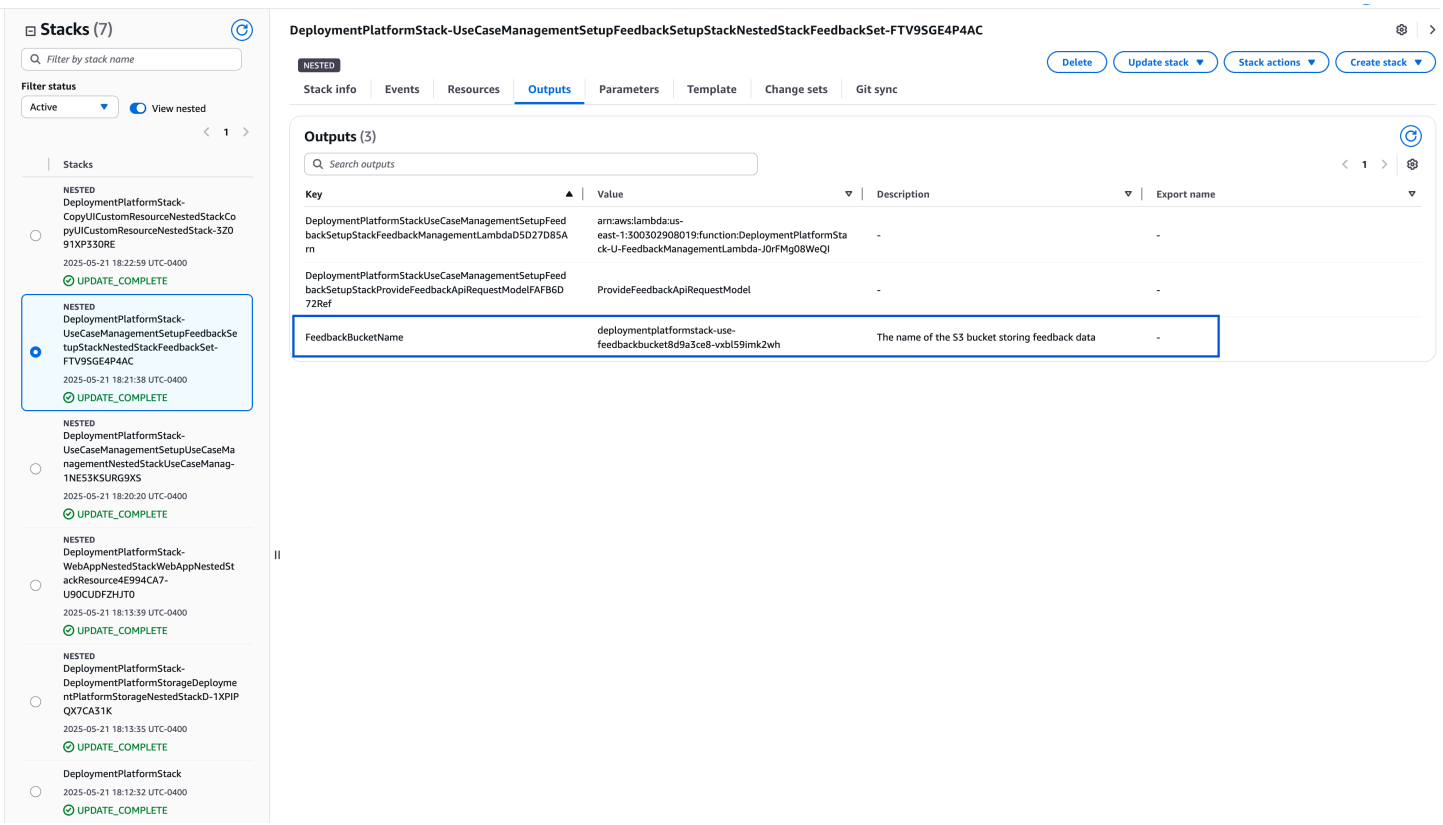
会話の間、ソリューションはチャット履歴を保持し、会話形式のエクスペリエンスを可能にします。これにより、クエリの曖昧さ回避とフォローアップの質問が可能になります。会話をリセットし、このインタラクションのチャット履歴をすべて削除するには、チャットウィンドウの上部にある [Clear conversation] を選択します。会話がクリアされると、新しいセッションが作成され、再び設定を編集できるようになります。

ユーザーが収集したフィードバックへのアクセスと分析

v3.0.0 以降、デプロイダッシュボードはネストされたフィードバックスタックをデプロイします。これにより、ダッシュボードでデプロイされた Text および Bedrock エージェントユースケースは、LLM/エージェントが生成するレスポンスのフィードバック収集機能を利用できるようになります。特に、ユーザーはオプションのコメントとともに肯定的または否定的なフィードバックを提供できます。ユーザーが否定的なフィードバックを提供する場合、「不正確」、「不完全または不十分」、「有害」、および/または「その他」のいずれかの否定的なカテゴリをさらに選択できます。

ユーザーがフィードバックを提供すると、フィードバックはユースケース ID、年、月ごとにパーティション化された S3 バケットに保存されます。ユースケース ID はデプロイダッシュボードで確認でき、フィードバック S3 バケットはデプロイダッシュボードスタックのフィードバックネストスタックの出力で確認できます。

デプロイスタックの図解 - フィードバックバケット名の検索



ユーザーフィードバックは、最小限の情報を含む API リクエストとして送信されます。

```
{
  "useCaseRecordKey": "a1b2c3d4-e5f6g7h8",
```

```
  "conversationId": "12345678-1234-1234-1234-123456789012",
  "messageId": "87654321-4321-4321-4321-210987654321",
  "rephrasedQuery": "What are the key features of the Generative AI Application Builder
 on AWS?",
  "sourceDocuments": [
    "s3://bucket-name/document1.pdf",
    "s3://bucket-name/document2.pdf"
  ],
  "feedback": "positive",
  "feedbackReason": [
    "Incomplete or insufficient"
  ],
  "comment": "The response was helpful but could include more details about important
 features."
}
```

このペイロードは、デプロイ時にユースケースの正しい設定を識別する useCaseRecordKey を使用して Lambda によって処理されます。この設定は、ConversationTable の名前 (すべての会話と人間と AI メッセージシーケンスを含む) などのフィードバックの具体的な詳細を取得するために使用され、さらに実際の userInput と llmResponse を取得するために使用されます。このフィードバックレコードには、Bedrock エージェントユースケースの agentId や agentAliasId、この設定を使用する Text ユースケースの modelProvider、bedrockModelId などの追加の詳細もアタッチされます。この設定にアクセスする方法の詳細については、以下の「カスタムフィードバックマッピング」セクションを参照してください。各受信フィードバックリクエストは JSON オブジェクトとして保存され、Text ユースケースのサンプルフィードバックレコードは次のようになります。

```
{
   "useCaseId": "12345678-1234-1234-1234-123456789012",
   "useCaseRecordKey": "c07a2e3b-2f31b1e0",
   "userId": "22345678-1234-1234-1234-123456789012",
   "conversationId": "dd51de5d-5af1-4ec6-91d2-aadf14352109",
   "messageId": "32345678-1234-1234-1234-123456789012",
   "userInput": "What are its key features?",
   "rephrasedQuery": "What are the key features of the Generative AI Application
 Builder on AWS?",
   "llmResponse": "Generative AI Application Builder on AWS can help you build
 production ready enterprise chatbots rapidly.",
   "feedback": "negative",
   "feedbackReason": [
      "Incomplete or insufficient"
   ],
```

```
   "comment": "The response was helpful but could include more details about important
 features.",
   "timestamp": "2025-05-22T18:48:08.340Z",
   "feedbackId": "42345678-1234-1234-1234-123456789012",
   "useCaseType": "Text",
   "modelProvider": "Bedrock",
   "bedrockModelId": "amazon.nova-lite-v1:0",
   "ragEnabled": "false"
}
```

または、Bedrock エージェントユースケースの場合は次のようになります。

```
{
   "useCaseId": "12345678-1234-1234-1234-123456789012",
   "useCaseRecordKey": "c07a2e3b-2f31b1e0",
   "userId": "22345678-1234-1234-1234-123456789012",
   "conversationId": "dd51de5d-5af1-4ec6-91d2-aadf14352109",
   "messageId": "32345678-1234-1234-1234-123456789012",
   "userInput": "What are its key features?",
   "llmResponse": "Generative AI Application Builder on AWS can help you build
 production ready enterprise chatbots rapidly.",
   "feedback": "negative",
   "feedbackReason": [
      "Incomplete or insufficient"
   ],
   "comment": "The response was helpful but could include more details about important
 features.",
   "timestamp": "2025-05-22T18:48:08.340Z",
   "feedbackId": "42345678-1234-1234-1234-123456789012",
   "useCaseType": "Agent",
   "agentId": "AHFXUJCAK1",
   "agentAliasId": "KSEDKOS0BL"
}
```

このフィードバックは、さらなる処理、分析、モデルの再トレーニング/フィードバックループに使用できます。また、カスタムマッピングを追加して、フィードバック Lambda に保存されるフィードバックレコードを強化することもできます。

カスタムフィードバックマッピング

デプロイダッシュボードには、LLMConfigTable が含まれており、これは、キー LLMConfigTableName を持つデプロイダッシュボードスタックのスタック出力で見つかりま

す。LLMConfigTable には、デプロイダッシュボードウィザードを使用してユースケースをデプロイするときに管理者が選択した設定に基づいて、各ユースケースの設定が含まれています。各ユースケース設定は、useCaseRecordKey によって識別されます。LLMConfigTable のユースケース設定レコードの例を次に示します。

```
{
   "key": "2dd76cfa-bc1a14da",
   "config": {
      "ConversationMemoryParams": {
        ...
      },
      "FeedbackParams": {
         "CustomMappings": {
            "NumberOfDocs": "$.KnowledgeBaseParams.NumberOfDocs",
            "ScoreThreshold": "$.KnowledgeBaseParams.ScoreThreshold"
         },
         "FeedbackEnabled": true
      },
      "IsInternalUser": "true",
      "KnowledgeBaseParams": {
         "KendraKnowledgeBaseParams": {
            "ExistingKendraIndexId": "d2831033-667f-4539-ab28-e6c7c7c5988b",
            "RoleBasedAccessControlEnabled": false
         },
         "KnowledgeBaseType": "Kendra",
         "NumberOfDocs": 5,
         "ReturnSourceDocs": false,
         "ScoreThreshold": 0.3
      },
      "LlmParams": {
         "BedrockLlmParams": {
            "BedrockInferenceType": "QUICK_START",
            "ModelId": "amazon.nova-lite-v1:0"
         },
         "ModelParams": {},
         "ModelProvider": "Bedrock",
         "PromptParams": {
            ...
         },
         "RAGEnabled": true,
         "Streaming": false,
         "Temperature": 0.1,
         "Verbose": false
```

```
        },
        "UseCaseName": "test-rag-usecase",
        "UseCaseType": "Text"
    }
}
```

ユースケースに対してフィードバックが有効になっている場合、この設定には FeedbackParams オブジェクトが含まれます。このオブジェクトには、フィードバック S3 バケットに保存されるフィードバック JSON レコードに追加されるすべての追加フィールドの JSONPath を指定できる CustomMappings オブジェクトが含まれます。例えば、上記のサンプルユースケース設定の場合、CustomMappings には、JSONPath のルートとして config で始まる CustomMappings オブジェクトに NumberOfDocs と ScoreThreshold JSONPath が追加で含まれています。この設定では、フィードバック S3 バケットに保存されている各 JSON レコードは、既に提供されているフィールドとは別に、これら 2 つの追加値の取得を開始します。

フィードバックデータの分析

フィードバックデータは JSON オブジェクトとして S3 に保存されます。このフィードバックデータをよりアクセスしやすく実用的なものにするためのアプローチをいくつか紹介します。

AWS Glue と Amazon Athena を使用する

AWS Glue と Amazon Athena は、フィードバックデータをサーバーレスでカタログ化し、クエリし、分析する方法を提供します。

AWS Glue を使用すると、S3 バケット内のデータを検査してスキーマを推測し、関連するすべてのメタデータをカタログに記録する AWS Glue クローラーを作成できます。その後、Amazon Athena などのサービスを使用してデータをクエリできます。

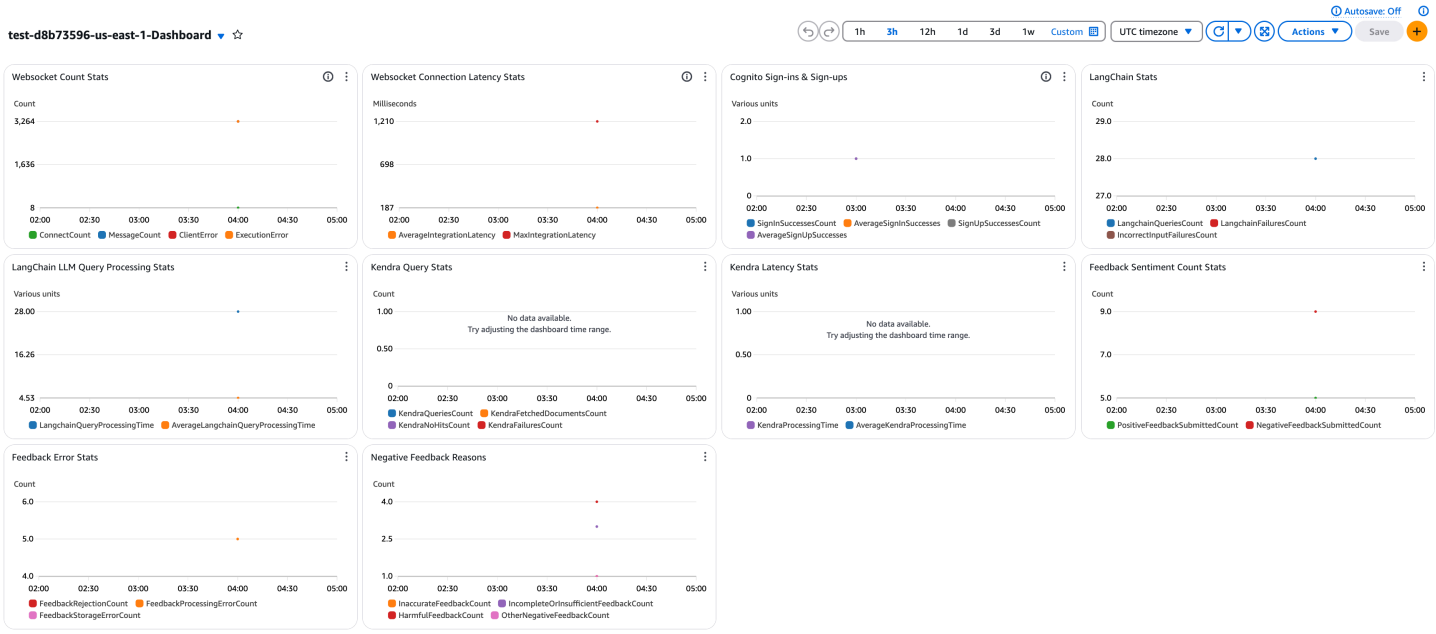
AWS Glue データカタログを使用してフィードバック S3 バケットを Amazon Athena に接続する手順については、「AWS Amazon Athena ドキュメント」を参照してください。Glue のより強力な機能の一部を使用して、このデータに対して抽出、変換、ロード (ETL) ジョブを実行し、分析やモデルの再トレーニングのユースケースに適した形式に変換することもできます。Glue を使用すると、特定のフィードバックタイプのレコードのフィルタリング、不足している情報の入力などの操作を実行できます。また、このデータを別の S3 バケットや別の AWS データストアなどの別のストレージロケーションにロードすることもできます。

> **Note**
>
> ユースケースによっては、フィードバックデータがまばらである可能性があるため、コストを最適化するために、Glue クローラーを毎晩ではなく定期的に (例えば、毎週) 実行するようにスケジュールすることを検討してください。

ソリューションの CloudWatch ダッシュボードの使用

また、ソリューションにパッケージ化された CloudWatch ダッシュボードにアクセスして、肯定的なフィードバックと否定的なフィードバックの傾向、否定的なフィードバックの理由のカテゴリなどをユースケースごとに確認することもできます。このダッシュボードは、AWS CloudWatch コンソール内の [ダッシュボード] でユースケース名を使用して見つけることができます。

CloudWatch ダッシュボードのユースケースの図解



このダッシュボードに追加のウィジェットを構築したり、Amazon Quick Sight ダッシュボードを作成したりすることもできます。

フィードバックデータ分析のベストプラクティス

- S3 バケットにデータライフサイクルポリシーを実装して、古いフィードバックデータを低コストのストレージ階層にアーカイブする
- ユースケースごとに個別の分析を作成して、モデル固有の改善の機会を特定する

- 否定的なフィードバックが許容レベルを超えたときにアラートをトリガーするフィードバックしきい値を設定する
- ステークホルダーやモデル改善チームと共有するための重要なインサイトを定期的に工クスポートする

デプロイの運用メトリクスを表示する

デプロイダッシュボードとユースケーススタックにはそれぞれ、ソリューションのさまざまな運用メトリクスを追跡する独自の CloudWatch ダッシュボードが付属しています。これらの CloudWatch ダッシュボードを使用して、さまざまなデプロイを比較できます。ダッシュボードにアクセスするには:

1. [CloudWatch console] (CloudWatch のコンソール) に移動する。
2. スタック名、または Universally Unique Identifier (UUID) を検索して、事前構築済みのダッシュボードを検索します。

例えば、Text ユースケースには、WebSocket 接続の数、ユーザーのサインインとサインアップの数、LLM が実行の処理にかかった時間などを追跡するグラフが付属しています。お客様はこれらのグラフを使用して、デプロイのさまざまな定量的メトリクスを比較できます。

Example

さまざまなユースケースに適用されるさまざまなモデルの定性的結果を比較することは困難です。クローン機能を使用すると、複数のデプロイをすばやく起動して、出力を並べて比較できます。

CloudWatch Logs Insights にアクセスする

このソリューションは、Lambda 関数のエラー、警告、情報、デバッグの各メッセージをログに記録します。ログ記録するメッセージのタイプを選択するには:

1. AWS Lambda コンソールで該当する関数を探します。
2. POWERTOOLS_LOG_LEVEL 環境変数を追加します。
3. この変数を該当するメッセージタイプに設定します。

詳細な手順については、「AWS Lambda デベロッパーガイド」の「Lambda 環境変数の作成」を参照してください。

選択できるログレベルのタイプは、次の表のとおりです。

レベル	説明
エラー	ログには、オペレーションの失敗の原因となるすべての情報が含まれます。
警告	ログには、関数の不整合を引き起こす可能性はあるが、必ずしもオペレーションの失敗の原因となるとは限らないすべての情報が含まれます。ログには、ERROR メッセージも含まれます。
情報	ログには、関数の動作に関するハイレベル情報が含まれます。ログには、ERROR および WARNING メッセージも含まれます。
DEBUG	ログには、関数の問題をデバッグする際に役立つ可能性のある情報が含まれます。ログには、ERROR、WARNING、INFO メッセージも含まれます。

次の手順に従って、このソリューションに CloudWatch Logs Insights を追加します。

1. 以下のとおり、関連するロググループを特定します。
   a. AWS CloudFormation コンソールにサインインします。
   b. ターゲットスタックを選択します。
   c. [Resources] タブを選択して、ターゲットの Lambda 関数を検索します。
   d. AWS Lambda コンソールにサインインして、各ターゲットの Lambda 関数を選択します。
   e. ターゲット Lambda 関数ごとに、[Monitor] タブを選択して、[View CloudWatch Logs] をクリックします。
   f. インサイトを抽出するロググループの名前をコピーします。
2. Amazon CloudWatch コンソールに移動します。
3. ナビゲーションメニューの [ログ] で、[ログのインサイト] を選択します。
4. [ログのインサイト] ページで、[ログ] タブを選択します。

5. 手順 1 のロググループ名を検索します。
6. 次のサンプルクエリのいずれかをコピーし、クエリフィールドに貼り付けます。

   a. すべてのクライアント例外を識別するには:

   ```
   fields @message
   |filter @message like /(?i)Exception/|stats count(*) as exceptionCount by @message
   ```

   b. 関数名別に呼び出し回数を取得するには:

   ```
   stats count(*) by function_name
   ```

   c. 5 分間隔で呼び出された回数を取得するには:

   ```
   stats count(*) as invocations by bin(5m)
   ```

   d. すべての AWS X-Ray のトレース ID を取得するには:

   ```
   filter @message like "XRAY TraceId"
   |parse @message "XRAY TraceId: * " as traceId|stats count(*) by traceId
   ```

   e. 特定の X-Ray のトレース ID に関するログを取得するには:

   ```
   filter @message like "your-traceid-here"
   ```

   f. 不正な WebSocket エラーを取得するには:

   ```
   fields
   @ingestionTime,
   @log,
   @logStream,
   @message,
   @requestId,
   @timestamp,
   errorMessage,
   errorType
   |filter @message like /Unauthorized/ and @message like /websocket/|sort @timestamp
    desc
   ```

   g. 公開されるメトリクスの数を取得するには:

   ```
   filter @message like "CloudWatchMetrics"
   ```

```
|parse @message /"Metrics":\s*\[(?<metrics>.*?)\]/|stats count(*) as metric_count
 by metrics
```

デベロッパーガイド

このセクションでは、ソリューションのソースコード、統合ガイド、カスタマイズガイド、API リファレンスを提供します。

ソースコード

GitHub リポジトリにアクセスして、このソリューションのソースファイルをダウンロードし、カスタマイズを他のユーザーと共有できます。

AWS での生成 AI アプリケーションビルダーテンプレートは、AWS Cloud Development Kit (AWS CDK) を使用して生成されます。詳細については、README.md ファイルを参照してください。

統合ガイド

このソリューションは、簡単に拡張できるように設計されています。このソリューションのオーケストレーションレイヤーは、LangChain を使用して構築されています。任意のモデルプロバイダー、ナレッジベース、または会話メモリタイプ (LangChain またはサードパーティー製で、LangChain コネクタを通じてコンポーネントが提供されているもの) を追加できます。

サポートされている LLM の拡張

カスタム LLM プロバイダーなどの別のモデルプロバイダーを追加するには、ソリューションの次の 3 つのコンポーネントを更新する必要があります。

1. カスタム LLM プロバイダーで設定されたチャットアプリケーションをデプロイする新しい `TextUseCase` CDK スタックを作成します。
   a. このソリューションの GitHub リポジトリのクローンを作成し、README.md ファイルの手順に従ってビルド環境をセットアップします。
   b. `source/infrastructure/lib/bedrock-chat-stack.ts` ファイルをコピー (または新規作成) して同じディレクトリに貼り付け、名前を `custom-chat-stack.ts` に変更します。
   c. ファイル内のクラスの名前を、`CustomLLMChat` などの適切な名前に変更します。
   d. このスタックに Secrets Manager シークレットを追加して、カスタム LLM の認証情報を保存することもできます。これらの認証情報は、次の段落で説明するチャット Lambda レイヤーでモデルを呼び出す際に取得できます。

2. 追加するモデルプロバイダーの Python ライブラリを含む Lambda レイヤーを構築してアタッチします。Amazon Bedrock ユースケースのチャットアプリケーションの場合、`langchain-aws` の Python ライブラリには、LangChain パッケージの上に構築されたカスタムコネクタが含まれており、AWS モデルプロバイダー (Amazon Bedrock および SageMaker AI)、ナレッジベース (Amazon Kendra および Amazon Bedrock ナレッジベース)、メモリタイプ (DynamoDB など) との接続に使用されます。同様に、他のモデルプロバイダーにも独自のコネクタがあります。このレイヤーは、このモデルプロバイダーの Python ライブラリをアタッチすることを目的としており、これにより、LLM を呼び出すチャット Lambda レイヤーでこれらのコネクタを使用できるようになります (ステップ 3)。このソリューションでは、カスタムアセットバンドラーを使用して Lambda レイヤーを構築し、CDK のアスペクトを使用してアタッチします。カスタムモデルプロバイダーライブラリの新しいレイヤーを作成するには:

   a. `LambdaAspects` ファイルの `source/infrastructure/lib/utils/lambda-aspects.ts` クラスに移動します。

   b. ファイル内で提供されている Lambda アスペクトクラスの機能を拡張する方法 (`getOrCreateLangchainLayer` メソッドの追加など) についての手順に従います。この新しいメソッド (`getOrCreateCustomLLMLayer` など) を使用するには、`source/infrastructure/lib/utils/constants.ts` ファイル内の `LLM_LIBRARY_LAYER_TYPES` 列挙型も更新します。

3. `chat` Lambda 関数を拡張して、新しいプロバイダーのビルダー、クライアント、ハンドラーを実装します。

   `source/lambda/chat` には、さまざまな LLM の LangChain 接続と、これらの LLM を構築するためのサポートクラスが含まれています。これらのサポートクラスは、ビルダーとオブジェクト指向の設計パターンに従って LLM を作成します。

   各ハンドラー (`bedrock_handler.py` など) は、まず client を作成し、必要な環境変数について環境をチェックしてから、`get_model` メソッドを呼び出して LangChain LLM クラスを取得します。その後、生成メソッドが呼び出されて LLM が起動し、その応答を取得します。LangChain は現在 Amazon Bedrock のストリーミング機能をサポートしていますが、SageMaker AI はサポートしていません。ストリーミング機能または非ストリーミング機能に基づいて、適切な WebSocket ハンドラー (`WebsocketStreamingCallbackHandler` または `WebsocketHandler`) が呼び出され、 `post_to_connection` メソッドを使用して応答が WebSocket 接続に送り返されます。

   `clients/builder` フォルダには、ビルダーパターンを使用して LLM ビルダーを構築するのに役立つクラスが含まれています。まず、DynamoDB の設定ストアから `use_case_config` が取得されます。このストアには、構築するナレッジベース、会話メモリ、モデルのタイプに関する詳細が格納されています。また、モデルパラメータやプロンプトなど、関連するモデルの詳細も

含まれています。ビルダーは、ナレッジベースの作成、会話コンテキストを維持するための LLM 用会話メモリの作成、ストリーミングケースと非ストリーミングケースに応じた LangChain コールバックの設定、提供されたモデル設定に基づく LLM モデルの作成の手順を支援します。この DynamoDB 設定は、デプロイダッシュボードからユースケースをデプロイするとき (またはデプロイダッシュボードなしでスタンドアロンのユースケーススタックデプロイでユーザーによって提供されるとき) に、ユースケースの作成時に保存されます。

clients/factories サブフォルダには、LLM の設定に基づいて適切な会話メモリとナレッジベースクラスを設定するのに役立ちます。これにより、実装でサポートする他のナレッジベースやメモリタイプへの拡張が容易になります。

shared サブフォルダには、ビルダーがファクトリー内でインスタンス化するナレッジベースと会話メモリの具体的な実装が含まれています。また、RAG ユースケースでのドキュメント取得のために、LangChain 内で呼び出される Amazon Kendra および Amazon Bedrock ナレッジベース用のリトリーバーのほか、LangChain LLM モデルで使用されるコールバックも含まれています。

LangChain の実装では、会話チェーンを構成するために LangChain 式言語 (LCEL) を使用してします。RunnableWithMessageHistory クラスは、カスタム LCEL チェーンを使用して会話履歴を維持するために使われます。これにより、例えばソースドキュメントを返したり、ナレッジベースに送信されたリフレーズされた (または曖昧性の解消された) 質問を LLM も送信したりできます。

カスタムプロバイダーの独自の実装を作成するには、次の方法があります。

a. bedrock_handler.py ファイルをコピーして独自のカスタムハンドラー (custom_handler.py など) を作成します。これにより、カスタムクライアント (CustomProviderClient など。次のステップで指定します) が作成されます。
b. クライアントフォルダの bedrock_client.py をコピーし、名前を custom_provider_client.py (または CustomProvider など、特定のモデルプロバイダー名に応じた名前) に変更します。その中のクラスにも適切な名前を付けます (LLMChatClient を継承する CustomProviderClient など)。

   LLMChatClient が提供するメソッドを使用することも、独自の実装を作成してこれらをオーバーライドすることもできます。

   get_model メソッドは CustomProviderBuilder をビルドし (次のステップを参照)、ビルダーステップを使用してチャットモデルを構築する construct_chat_model メソッドを呼び出します。このメソッドは、ビルダーパターンの Director として機能します。

c. clients/builders/bedrock_builder.py をコピーして名前を custom_provider_builder.py に変更し、その中のクラスの名前を LLMBuilder (llm_builder.py) を継承する CustomProviderBuilder に変更します。LLMBuilder が提供するメソッドを使用することも、独自の実装を作成してこれらをオーバーライドすることもできます。ビルダーの各ステップは、クライアントの construct_chat_model メソッド内で順番に呼び出されます (例: set_model_defaults、set_knowledge_base、set_conversation_memory など)。

   set_llm_model メソッドは、それ以前に呼び出されたメソッドによって設定されたすべての値を使用して、実際の LLM モデルを作成します。具体的には、RAG あり (CustomProviderRetrievalLLM) または RAG なし (CustomProviderLLM) の LLM を、DynamoDB に保存された LLM 設定から取得した rag_enabled variable に基づいて作成します。

   この設定は、LLMChatClient クラスの retrieve_use_case_config メソッドで取得されます。

d. RAG ありと RAG なしのユースケースのどちらが必要かに基づいて、CustomProviderLLM または CustomProviderRetrievalLLM を llm_models サブフォルダに実装します。これらのモデルを実装するために必要な機能の大部分は、RAG なしのユースケースでは BaseLangChainModel クラスで、RAG ありのユースケースでは RetrievalLLM クラスで提供されています。

   llm_models/bedrock.py ファイルをコピーし、独自のカスタムプロバイダーを参照する LangChain モデルを呼び出すために必要な変更を加えることができます。例えば、Amazon Bedrock では、ChatBedrock クラスを使用して LangChain を通じてチャットモデルを作成します。

   generate メソッドは、LangChain の LCEL チェーンを使用して LLM の応答を生成します。

   また、get_clean_model_params メソッドを使用して、LangChain やモデルの要件に合わせてモデルパラメータをサニタイズすることもできます。

サポートされている Strands ツールの拡張

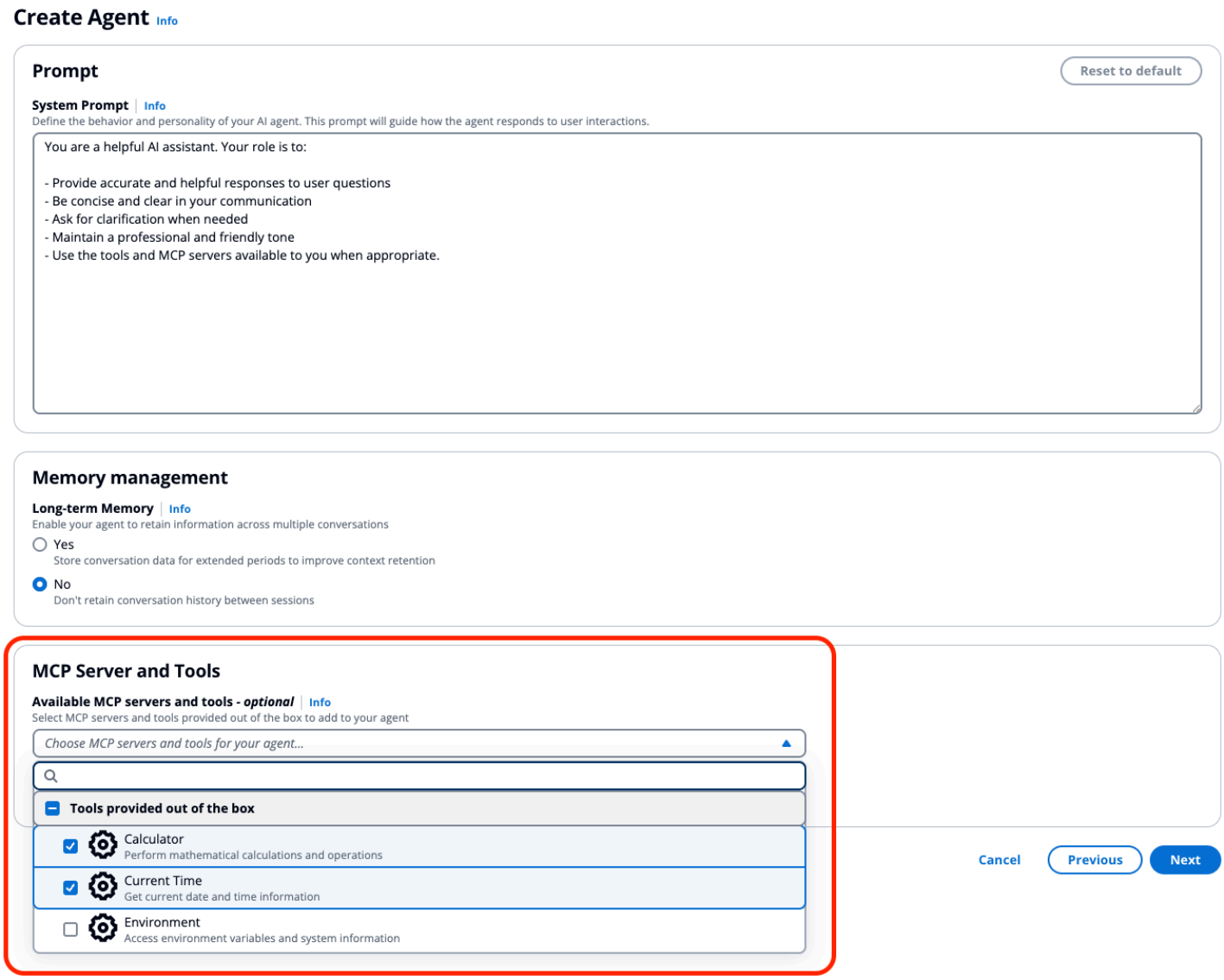
このソリューションを使用すると、MCP サーバー、AI エージェント、およびマルチエージェントワークフローを構築およびデプロイできます。エージェントビルダーエクスペリエンスの中で、MCP サーバーをアタッチして、エージェントに追加の機能を提供できます。MCP サーバーに加

えて、Strands が提供する組み込みツール (ソリューションで使用される基盤となるフレームワーク) を活用できます。

このソリューションには、すぐに使用できる以下の Strands ツールが事前設定されています。

- 現在の時刻 (デフォルトで有効)
- 計算ツール (デフォルトで有効)
- 環境

組み込みの Strands ツールが表示されている、エージェントビルダーウィザードの MCP サーバーとツールの選択



追加の Strands ツールを使用してエージェントを拡張するには、このセクションで説明されている 4 ステップのプロセスに従います。

ステップ 1: Strands ツールを検索する

使用可能な Strands ツールを参照して、使用するツールを特定します。各ツールには特定の機能と設定要件があります。

例えば、Amazon Bedrock ナレッジベースの取得機能を追加するには、取得ツールを使用します。

ステップ 2: SSM パラメータを更新する

エージェントビルダーデプロイ UI でツールを使用できるようにするには、サポートされている Strands ツールを定義する AWS Systems Manager Parameter Store のパラメータを更新します。

1. AWS アカウントの AWS Systems Manager Parameter Store に移動します。
2. `/gaab/<stack-name>/strands-tools` パラメータを見つけます。
3. 次の JSON 構造を使用して、ツール設定を既存のリストの末尾に追加します。

```
{
  "name": "Bedrock KB Retrieve",
  "description": "Retrieve information from Bedrock Knowledge Base",
  "value": "retrieve",
  "category": "AI",
  "isDefault": false
}
```

フィールド	説明
.name	エージェントビルダー UI に表示される表示名
説明	ツールの機能の簡単な説明
値	Strands ツールパッケージで定義されている正確なツール名
category	UI 内でツールをグループ化するための組織カテゴリ

フィールド	説明
isDefault	新しいエージェントに対してツールをデフォルトで有効にするかどうか

ステップ 3: 環境変数を設定する

多くの Strands ツールでは、設定に環境変数が必要です。変数は次の 2 つの方法で定義できます。

オプション 1: AgentCore Runtime での直接設定

必要な環境変数を使用して、Amazon Bedrock AgentCore Runtime でデプロイされたエージェントを直接更新します。

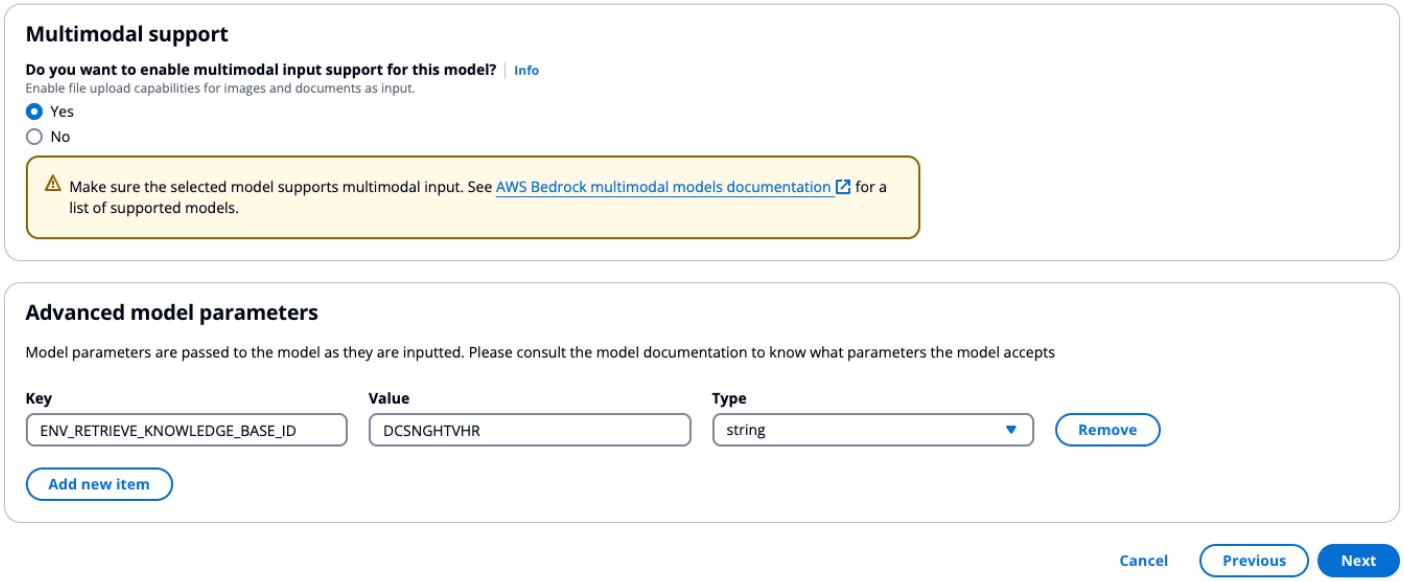
オプション 2: デプロイウィザードのモデルパラメータ

モデルパラメータセクションを使用して、エージェントビルダーウィザードのモデル選択ステップ中に環境変数を追加します。命名規則 ENV_<ALL_CAPS_TOOL_NAME>_<env_variable_name> に従う環境変数は、実行時にエージェントの実行環境に <env_variable_name> として自動的にロードされます。

例えば、次のようになります。

- ENV_RETRIEVE_KNOWLEDGE_BASE_ID が になりますKNOWLEDGE_BASE_ID
- ENV_RETRIEVE_MIN_SCORE が になりますMIN_SCORE

ENV_RETRIEVE_KNOWLEDGE_BASE_ID 設定を示す高度なモデルパラメータセクション



必要な環境変数を特定するには、特定のツールのドキュメントまたはソースコードを参照してください。取得ツールについては、ソースコードに設定オプションがあります。

ステップ 4: IAM アクセス許可を追加する

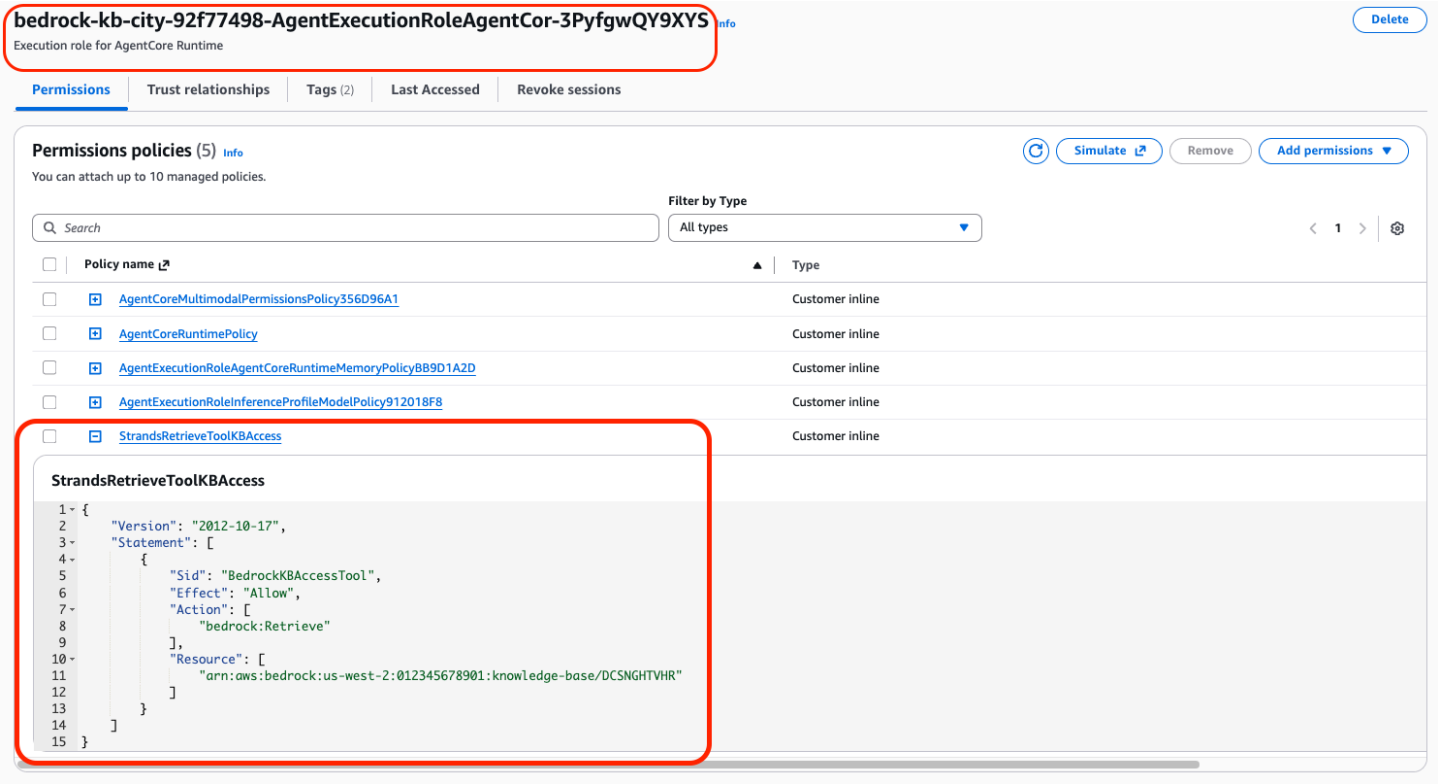
AgentCore Runtime 実行ロールに必要な IAM アクセス許可を手動で追加して、エージェントがツールを使用できるようにします。

例えば、Amazon Bedrock ナレッジベースで取得ツールを使用するには、以下を行います。

1. 使用中の AWS アカウントで IAM コンソールに移動します。
2. エージェントの AgentCore Runtime 実行ロールを見つけます。
3. 以下のアクセス許可を追加します。

```
{
  "Effect": "Allow",
  "Action": "bedrock:Retrieve",
  "Resource": "arn:aws:bedrock:region:account-id:knowledge-base/knowledge-base-id"
}
```

AgentCore Runtime 実行ロールにアタッチされた StrandsRetrieveToolKBAccess ポリシーを示す IAM コンソール

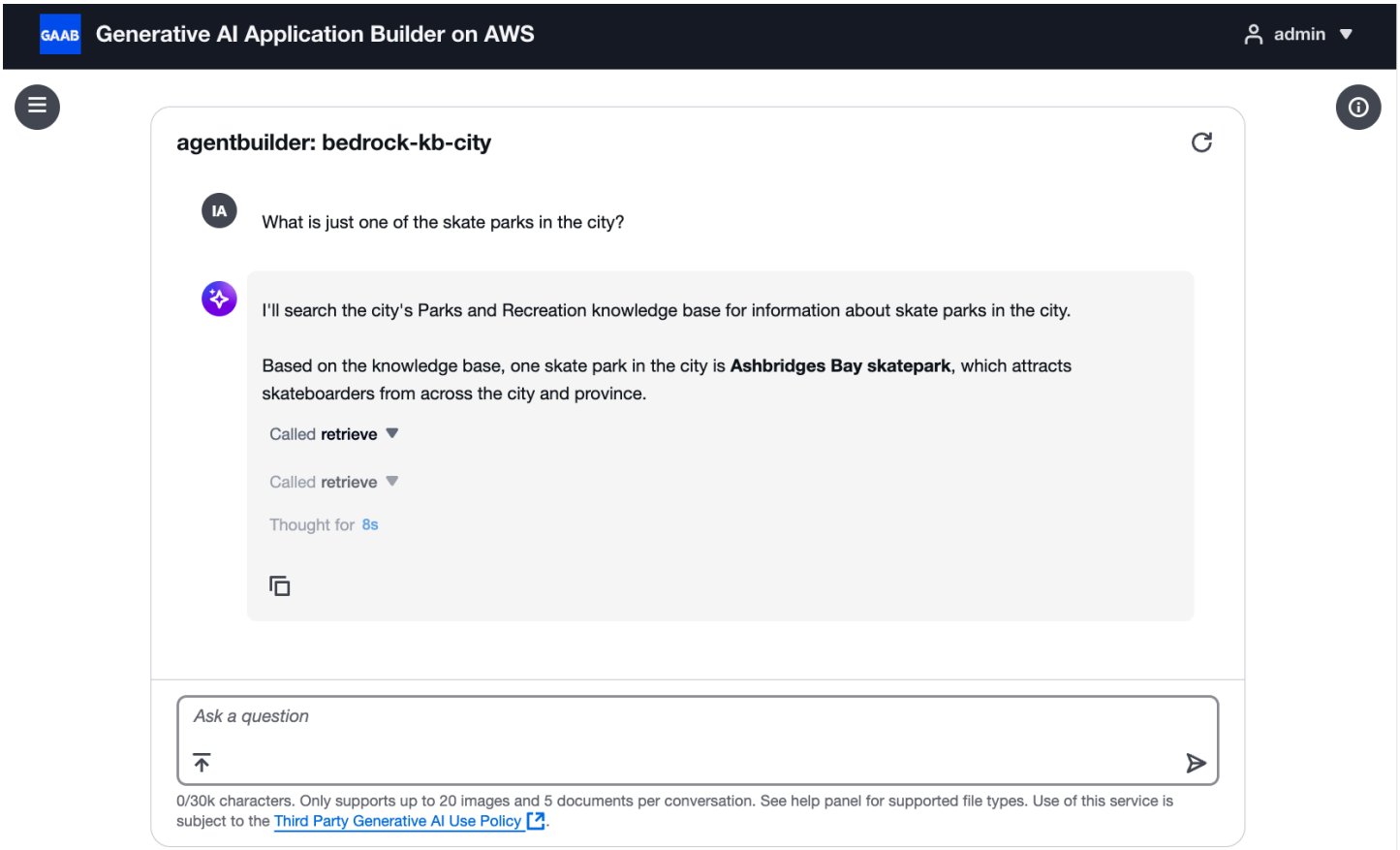


どんな特定のアクセス許可が必要になるかは、ツールによって異なります。ツールのドキュメントと AWS のサービスドキュメントを参照して、適切な IAM アクセス許可を決定します。

ステップ 5: エージェントをテストする

設定ステップを完了したら、エージェントをテストしてツールが正しく動作していることを確認します。エージェントの実行ログとレスポンスにツール呼び出しが表示されます。

エージェントが取得ツールを使用してスケートパークに関する質問に回答する



> **Note**
>
> 利用可能な Strands ツールとその機能の完全なリストについては、「Strands コミュニティツールのドキュメント」を参照してください。

サポートされているナレッジベースと会話メモリタイプの拡張

会話メモリまたはナレッジベースの実装を追加するには、shared フォルダに必要な実装を追加し、ファクトリーと適切な列挙を編集して、これらのクラスのインスタンスを作成します。

Parameter Store 内に保存されている LLM 設定を指定すると、LLM 用の適切な会話メモリとナレッジベースが作成されます。例えば、ConversationMemoryType を DynamoDB として指定すると、DynamoDBChatMessageHistory (shared_components/memory/ddb_enhanced_message_history.py 内で利用可能) のインスタンスが作成されます。KnowledgeBaseType が Amazon Kendra として指定されている場合、KendraKnowledgeBase (shared_components/knowledge/kendra_knowledge_base.py 内で利用可能) のインスタンスが作成されます。

コード変更のビルドとデプロイ

`npm run build` コマンドを使用してプログラムをビルドします。エラーが解決したら、`cdk synth` を実行してテンプレートファイルとすべての Lambda アセットを生成します。

1. `0/stage-assets.sh` スクリプトを使用すると、生成されたアセットをアカウントのステージングバケットに手動でステージングできます。
2. 次のコマンドを使用して、プラットフォームをデプロイまたは更新します。

```
cdk deploy DeploymentPlatformStack --parameters AdminUserEmail='admin-
email@amazon.com'
```

追加の AWS CloudFormation パラメータも AdminUserEmail パラメータとともに指定する必要があります。

カスタマイズガイド

Cognito ユーザープールの管理

デプロイダッシュボードがデプロイされると、アプリケーションの認証を行うための Amazon Cognito ユーザープールと管理者ユーザーが作成されます。このユーザープールは、デプロイダッシュボードとすべてのユースケースで共有されます。ダッシュボードのデプロイ時に作成された管理者ユーザーには、ダッシュボードを使用してデプロイされるすべてのユースケースへのアクセス権が自動的に付与されます。このメカニズムは、Amazon Cognito ユーザープールグループを介して提供されます。

ユースケースをダッシュボードからデプロイする際に E メールを指定すると、共有ユーザープールにユーザーが作成され、そのユースケース用に名前がつけられたユーザーグループも同時に作成されます。その後、新しく作成されたユーザーがそのグループに追加され、ユースケースへのアクセス権がユーザーに付与されます。

特定のユースケースにユーザーを追加する場合は、Cognito ユーザープールにユーザーを作成し、アクセスを許可したいユースケースに対応するグループに追加します。ステップバイステップガイドについては、「AWS Management Consoleでの新しいユーザーの作成」を参照してください。

同様に、追加の管理者ユーザーを作成する場合は、新しいユーザーを作成し、ユーザープールの管理者グループに追加する必要があります。

ユーザー名は、指定された E メールの @ の前の部分に、生成されたユースケースの UUID (管理者ユーザーの場合は `-admin`) を追加することによって作成されます。

[グループ] タブで、ユースケースの名前 (ウィザードで指定したもの) とユースケースの UUID を使用して、[管理者] グループと各ユースケースのグループが自動的に作成されたことを確認できます。

API リファレンス

このセクションでは、ソリューションの API リファレンスを提供します。

デプロイダッシュボード

REST API	HTTP メソッド	機能	認可された呼び出し元
`/deployments`	`GET`	すべてのデプロイを取得します。	Amazon Cognito 認証済み JWT トークン
`/deployments`	`POST`	新しいユースケースのデプロイを作成します。	Amazon Cognito 認証済み JWT トークン
`/deployments/{useCaseId}`	`GET`	1 つのデプロイのデプロイ詳細を取得します。	Amazon Cognito 認証済み JWT トークン
`/deployments/{useCaseId}`	`PATCH`	指定されたデプロイを更新します。	Amazon Cognito 認証済み JWT トークン
`/deployments/{useCaseId}`	`DELETE`	指定されたデプロイを削除します。	Amazon Cognito 認証済み JWT トークン
`/model-info/use-case-types`	`GET`	デプロイで使用できるユースケースタイプを取得します。	Amazon Cognito 認証済み JWT トークン
`/model-info/{useCaseType}/providers`	`GET`	指定されたユースケースタイプで使用可	Amazon Cognito 認証済み JWT トークン

REST API	HTTP メソッド	機能	認可された呼び出し元
		能なモデルプロバイダーを取得します。	
/model-info/{useCaseType}/{providerName}	GET	指定されたプロバイダーとユースケースタイプで使用可能なモデルの ID を取得します。	Amazon Cognito 認証済み JWT トークン
/model-info/{useCaseType}/{providerName}/{modelId}	GET	指定されたモデルに関する情報 (デフォルトのパラメータを含む) を取得します。	Amazon Cognito 認証済み JWT トークン

> **Note**
>
> API との統合を容易にするため、OpenAPI ファイルと Swagger ファイルを API Gateway からエクスポートすることもできます。「API Gateway から REST API をエクスポートする」を参照してください。

POST ペイロードと PATCH ペイロード

新しいユースケースを作成する/deployments エンドポイントへの POST ペイロードの例については、以下を参照してください。

```
{
 "UseCaseName": "usecase1",
 "UseCaseDescription": "Description of the use case to be deployed. For display
 purposes", // optional
 "DefaultUserEmail": "placeholder@example.com", // optional, if not provided, the
 Cognito Group and User will not be created
 "DeployUI": true, // optional
 "VpcParams": {
 "VpcEnabled": true,
 "CreateNewVpc": false,
```

```
// provide these if not creating new vpc
"ExistingVpcId": "vpc-id",
"ExistingPrivateSubnetIds": ["subnet-1", "subnet-2"],
"ExistingSecurityGroupIds": ["sg-1", "sg-2"]
},
"ConversationMemoryParams": {
"ConversationMemoryType": "DynamoDB",
"HumanPrefix": "user", // optional
"AiPrefix": "ai", // optional
"ChatHistoryLength": 10 // optional
},
"KnowledgeBaseParams": {
"KnowledgeBaseType": "Bedrock",
// one of the following based on selected provider
"BedrockKnowledgeBaseParams": {
"BedrockKnowledgeBaseId": "my-bedrock-kb",
"RetrievalFilter": {}, // optional
"OverrideSearchType": "HYBRID" // optional
},
"KendraKnowledgeBaseParams": {
"AttributeFilter": {}, // optional
"RoleBasedAccessControlEnabled": true, // optional
"ExistingKendraIndexId": "12345678-abcd-1234-abcd-1234567890ab",
// provide the following in place of ExistingKendraIndexId if you want the solution to
deploy an index for you
"KendraIndexName": "index",
"QueryCapacityUnits": 1, // optional
"StorageCapacityUnits": 1, // optional
"KendraIndexEdition": "DEVELOPER" // optional
},
"NoDocsFoundResponse": "Sorry, I couldn't find any relevant information for your
query.", // optional
"NumberOfDocs": 3, // optional
"ScoreThreshold": 0.7, // optional
"ReturnSourceDocs": true // optional
},
"LlmParams": {
"ModelProvider": "Bedrock | SAGEMAKER",
// one of the following based on selected provider
"BedrockLlmParams": {
"ModelId": "model-id", // use this for on demand models. Can't use with ModelArn
"ModelArn": "model-arn", // use this for provisioned/custom models. Can't use with
ModelId,
"InferenceProfileId": "profile-id"
```

```
 "GuardrailIdentifier": "arn:aws:bedrock:us-east-1:123456789012:guardrail/my-
guardrail", // optional
 "GuardrailVersion": "1" // optional. Required if GuardrailIdentifier provided.
 },
 "SageMakerLlmParams": {
 "EndpointName": "some-endpoint",
 "ModelInputPayloadSchema": {},
 "ModelOutputJSONPath": "$."
 },
 // optional. Passes on arbitrary params to the underlying LLM.
 "ModelParams": {
 "param1": {
 "Value": "value1",
 "Type": "string"
 },
 "param2": {
 "Value": 1,
 "Type": "integer"
 }
 },
 // optional
 "PromptParams": {
 "PromptTemplate": "some template",
 "UserPromptEditingEnabled": true,
 "MaxPromptTemplateLength": 1000,
 "MaxInputTextLength": 1000,
 "DisambiguationPromptTemplate": "some disambiguation template",
 "DisambiguationEnabled": true
 },
 "Temperature": 1.0, // optional
 "Streaming": true, // optional
 "RAGEnabled": true, // optional. Must be true if providing KnowledgeBaseParams above.
 "Verbose": false // optional
 },
 "AgentParams": {
 "AgentType": "Bedrock",
 "BedrockAgentParams": {
 "AgentId": "agent-id",
 "AgentAliasId": "alias-id",
 "EnableTrace": true
 }
 },
 // optional
 "AuthenticationParams": {
```

```
 "AuthenticationProvider": "Cognito",
 "CognitoParams": {
 "ExistingUserPoolId": "user-pool-id",
 "ExistingUserPoolClientId": "client-id" // optional. If not provided, the solution
 will create a client for you in the provided pool
 }
 }

}
```

更新の場合、構造は上記と同じですが、いくつかの注意点があります。

- ユースケース名は変更できません
- ユースケースが VPC にデプロイされた後は、セキュリティグループとサブネットのみ変更できます。VPC 自体は変更できません。
- ナレッジベースとして Kendra インデックスが作成された場合、そのインデックスの設定 (KendraIndexName、QueryCapacityUnits など) を変更することはできません。

共有ユースケース API

Text と Bedrock エージェントの両方のユースケースで、次の REST API エンドポイントを使用できます。

REST API	HTTP メソッド	機能	認可された呼び出し元
`/details/{useCaseConfigKey}`	`GET`	特定のユースケースの設定の詳細を取得します。	Amazon Cognito 認証済み JWT トークン

WebSocket API	機能	認可された呼び出し元
`/$connect`	WebSocket 接続を開始し、ユーザーを認証します。	Amazon Cognito 認証済み JWT トークン

WebSocket API	機能	認可された呼び出し元
`/$disconnect`	WebSocket 接続が切断されたときに呼び出されるエンドポイント。	Amazon Cognito 認証済み JWT トークン

ユースケースの詳細 API

詳細 API エンドポイントは、特定のユースケースに関する情報を取得します。

```
GET /details/{useCaseConfigKey}
```

このエンドポイントは、モデルパラメータ、ナレッジベース設定、その他のデプロイ情報など、特定のユースケースの設定の詳細を返します。承認には Amazon Cognito 認証 JWT トークンが必要です。

Text ユースケース

WebSocket API	機能	認可された呼び出し元
`/sendMessage`	ユーザーのチャットメッセージを WebSocket に送信し、設定済みの LLM エクスペリエンスで処理します。	Amazon Cognito 認証済み JWT トークン

REST API	HTTP メソッド	機能	認可された呼び出し元
`/feedback/{useCaseId}`	`POST`	特定のユースケースに関するユーザーフィードバックを送信します。	Amazon Cognito 認証済み JWT トークン

sendMessage ペイロード

/sendMessage API と直接統合する場合は、次のリクエストおよびレスポンスペイロード形式に従う必要があります。

リクエストペイロード

```
{
 "action": "sendMessage",
 "question": "the message to send to the api",
 "conversationId": "", // If not provided, a new conversation will be created, with the
 conversationId returned in the response. All subsequent messages in that conversation
 (where history is retained), should provide the conversationId there.
 "promptTemplate": "", // Optional. Overrides the configured prompt
 "authToken": "XXXX" // Optional. accessToken from cognito flow. Required for RAG with
 RBAC
}
```

Parameter Name	型	説明
action	String	現在、WebSocket では "sendMessage" アクションのみをサポートしています。
question	String	LLM に送信するユーザー入力。
conversationId	String	会話を識別する UUID。指定しない場合は新しい会話が作成され、その conversationId が応答で返されます。その会話の後続のすべてのメッセージで履歴やコンテキストを保持する場合は、そこに conversationId が提供されます。
promptTemplate	String [オプション]	このメッセージ用のプロンプトテンプレートを上書きします。空または指定されていない場合、デプロイ時に設定さ

Parameter Name	型	説明
		れたデフォルトのプロンプトが使用されます。指定する場合は、設定に応じて適切なプレースホルダーを含める必要があります (例: RAG なしの Sagemaker AI デプロイの場合は {history} と {input}、すべてのデプロイに RAG ありの場合は {context} を追加する)。
authToken	`String` [オプション]	Cognito 認証フローから取得された accessToken。ロールベースのアクセスコントロール (RBAC) を使用して RAG 用に設定されたチャット WebSocket エンドポイントを呼び出すときに必要です。この JWT トークンの cognito:groups クレームリストは、Kendra インデックス内のドキュメントへのアクセス制御に使用されます。このパラメータは、RAG なしのユースケースには必要ありません。また、RAG ありのユースケースでも、RBAC が無効になっている場合はは必要ありません。

レスポンスペイロード

質問に対する応答

WebSocket API は、各クエリに対して次のように構造化された JSON オブジェクトで応答します。ストリーミングが無効になっている場合は 1 件、ストリーミングが有効になっている場合は複数件のオブジェクトが返されます。

```
{
 "data": "some data",
 "conversationId": "id",
}
```

Parameter Name	型	説明
data	String	ストリーミングが有効な場合は LLM からの応答の一部、無効な場合はまたは応答全体が含まれます。ストリーミングを使用している場合、データの内容が END_CONVERSATION となっているものこの形式の応答で送信され、1 つの質問に対する応答の終了を示します。
conversationId	String	この sourceDocument の応答が属する会話の ID。

ソースドキュメントの応答

ソースドキュメントを返すように RAG ユースケースを設定している場合、応答の生成に使用された各ソースドキュメントについて、次のペイロードがすべての応答の末尾に返されます。

```
{
 "sourceDocument": {
 "excerpt": "some excerpt from the",
 "location": "s3://fake-bucket/test.txt",
 "score": 0.500,
 "document_title": null,
 "document_id": null,
 "additional_attributes": null
 },
```

```
 "conversationId": "some-id"
}
```

Parameter Name	型	説明
excerpt	`String`	ソースドキュメントからの抜粋。
location	`String`	ソースドキュメントの場所。使用されるデータソースとナレッジベースのタイプによって異なりますが、S3 の URI やウェブサイトなどです。
score	`Number` | `String`	質問に対する関連度スコア。Bedrock の場合は 0 ~ 1 の浮動小数点数、Kendra の場合は HIGH、LOW などの文字列になります。
document_title	`String`	返されたソースドキュメントのタイトル。Kendra を使用する場合にのみ返されます。
document_id	`String`	返されたソースドキュメントの ID。Kendra を使用する場合にのみ返されます。
additional_attributes	`String`	このフィールドには、取り込み時にナレッジベースでカスタマイズされたドキュメント上のすべての追加属性が含まれます。
conversationId	`String`	この sourceDocument の応答が属する会話の ID。

フィードバック API ペイロード

以下は、特定のユースケースに関するユーザーフィードバックを送信する /feedback/{useCaseId} エンドポイントへの POST ペイロードの例です。

```
{
  "useCaseRecordKey": "12345678-12345678",
  "conversationId": "12345678-1234-1234-1234-123456789012",
  "messageId": "12345678-1234-1234-1234-123456789012",
  "feedback": "positive",
  "feedbackReason": ["accurate", "helpful"],
  "comment": "This response was very helpful.",
  "rephrasedQuery": "What are the key features of Amazon Bedrock?",
  "sourceDocuments": [
    "s3://bucket-name/document1.pdf",
    "s3://bucket-name/document2.pdf"
  ]
}
```

Bedrock エージェントユースケース

WebSocket API	機能	認可された呼び出し元
/invokeAgent	ユーザーのメッセージを WebSocket に送信し、設定されたエージェントで処理します。	Amazon Cognito 認証済み JWT トークン

invokeAgent ペイロード

/invokeAgent API と直接統合する場合は、次のリクエストおよびレスポンスペイロード形式に従う必要があります。

リクエストペイロード

```
{
"action": "invokeAgent",
"inputText": "User query to the agent",
"conversationId": "", // Optional. Empty conversationId implies a new conversation.
 When not provided, a new conversationId will be created and returned with the
 response. All subsequent messages in the same conversation should provide the same
 conversationId (i.e. chat memory/history is maintained).
```

```
"authToken": "XXXX" // Optional. accessToken from cognito flow. If provided, it needs
 to be a valid JWT token associated with the user
}
```

パラメータ名	型	説明
action	`String`	WebSocket では `invokeAgent` アクションのみをサポートしています。
inputText	`String`	LLM に送信するユーザー入力。
conversationId	`String[Optional]`	会話を一意に識別する UUID。この値を指定しない場合、ソリューションは新しい会話を作成し、レスポンスに conversationId が返されます。その会話の後続のすべてのメッセージで履歴やコンテキストを保持する場合は、そこに conversationId が提供されます。
authToken	`String[Optional]`	Amazon Cognito 認証フローから取得された accessToken。このパラメータは必須ではありません。これを指定すると、JWT トークンが検証されます。これにより、このソリューションの拡張が容易になります。

レスポンスペイロード

質問に対する応答

WebSocket API は、各クエリに対して次のように構造化された JSON オブジェクトで応答します。ストリーミングが無効になっている場合は 1 件、ストリーミングが有効になっている場合は複数件のオブジェクトが返されます。

```
{
 "data" "some data",
 "conversationId": "id",
 }
```

パラメータ名	型	説明
data	String	エージェント呼び出しからの応答。
conversationId	String	会話の ID。

参照資料

このセクションには、このソリューションのデータ収集に関する情報、関連リソースへのポインタ、このソリューションに貢献したビルダーのリストが含まれています。

サポートされている LLM プロバイダー

このソリューションは、以下の LLM プロバイダーと統合できます。

1. Amazon Bedrock
   - ドキュメント: https://aws.amazon.com/bedrock/
   - サポートされているモデル
     - Amazon
       - Nova Lite
       - Nova Micro
       - Nova Pro
     - AI21 Labs
       - Jamba 1.5 Mini
       - Jamba 1.5 Large
     - Anthropic
       - Claude v3 Haiku
       - Claude v3.5 Sonnet
       - Claude v3.7 Sonnet (推論プロファイルを使用)
     - Cohere
       - コマンド R
       - コマンド R+
     - DeepSeek
       - DeepSeek-R1 (推論プロファイルを使用)
     - Meta
       - Llama 3
       - Llama 3.2 (推論プロファイルを使用)
     - Mistral AI

    - Mistral 7B Instruct
    - Mistral 8x7B Instruct
  - クロスリージョン推論
    - デプロイダッシュボードと同じリージョンで定義された推論プロファイルを使用する機能

2. Amazon SageMaker AI
   - ドキュメント: https://aws.amazon.com/sagemaker/
   - サポートされているモデル: Text to Text モデル

最新のモデルパラメータ、ベストプラクティス、推奨される使用法については、モデルプロバイダーのドキュメントを参照してください。

データ収集

このソリューションは、このソリューションの使用に関するオペレーションメトリクスを AWS (「データ」) に送信します。AWS ではこのデータを使用して、ユーザーがこのソリューション、関連サービスおよび製品をどのように使用しているかをよりよく理解し、提供するサービスや製品の改善に役立てます。AWS によるこのデータの収集には、AWS プライバシー通知が適用されます。

寄稿者

- Tarek Abdunabi
- Majd Arbash
- George Bearden
- Mukit Bin Momin
- Michael Connor
- Johny Duval
- Nihit Kasabwala
- Ahern Knox
- Simon Krol
- Michael Lin
- Tim Mekari
- Ibrahim Mohamed

- Omar Radwan Mohsen
- James Nixon
- Dekshitha Ravikumar
- Jae Shim
- Ajay Swamy
- Mohammed Taha
- Reet Takkar
- Dimitri Tchikatilov
- Jason Wreath
- Kamyar Ziabari

リビジョン

公開日: 2023 年 10 月 (最終更新日: 2025 年 1 月)

ソフトウェアの主な変更点と更新点を確認するには、GitHub リポジトリ内の CHANGELOG.md ファイルを参照してください。この改訂履歴には、各バージョンの改良点と修正点が明確に記録されています。

注意

お客様は、本書に記載されている情報を独自に評価する責任を負うものとします。本書は、(a) 情報提供のみを目的とし、(b) AWS の現行製品と慣行について説明しており、これらは予告なしに変更されることがあり、(c) AWS およびその関連会社、サプライヤー、またはライセンサーからの契約上の義務や保証をもたらすものではありません。AWS の製品やサービスは、明示または黙示を問わず、一切の保証、表明、条件なしに「現状のまま」提供されます。お客様に対する AWS の責任は AWS 契約によって規定されています。また、本文書は、AWS とお客様との間の契約に属するものではなく、また、当該契約が本文書によって修正されることもありません。

AWS での生成 AI アプリケーションビルダーは、Apache ライセンスバージョン 2.0 の条件に基づいてライセンスされます。

Important

AWS での生成 AI アプリケーションビルダーでは、任意の生成 AI モデルを利用して、AWS で生成 AI アプリケーションを構築し、デプロイできます。選択可能なモデルには、AWS が所有していない、または制御もできないサードパーティーの生成 AI モデル (「サードパーティーの生成 AI モデル」) も含まれます。
サードパーティーの生成 AI モデルの使用には、モデルの使用ライセンスを取得したときにサードパーティーの生成 AI モデルプロバイダーが提示した条件 (サービス規約、ライセンス契約、利用規約、プライバシーポリシーなど) が適用されます。
ユーザーは、サードパーティーの生成 AI モデルの使用が、それらに適用される条件、および適用されるあらゆる法律、規則、規制、ポリシー、または基準に準拠していることを確認する責任を負います。
また、使用するサードパーティーの生成 AI モデルについて、その出力や、サードパーティーの生成 AI モデルプロバイダーがデプロイに基づいて受信する可能性のあるデータをどのように使用するかなど、独自に評価する責任もユーザー側にあります。AWS は、ユーザーと AWS との契約に基づく「サードパーティーコンテンツ」であるサードパーティーの生成 AI モデルについて、いかなる表明も保証も行いません。AWS での生成 AI アプリケーションビルダーは、ユーザーと AWS との契約に基づき「AWS コンテンツ」として提供されます。