



# AWS で簡単に作る生成系 AI モデル

デモンストレーション

伊藤 芳幸 Yoshiyuki Ito

機械学習ソリューションアーキテクト

# 自己紹介

伊藤 芳幸 (いとう よしゆき)

機械学習ソリューションアーキテクト

- AWS の AI/ML を活用してビジネス価値を創出するお客様にアーキテクティング面の技術支援を提供
- Sler でエンジニアとコンサルタント、事業会社でデータストラテジストを経験

好きな AWS サービス

Amazon SageMaker



# 基盤モデルを活用する3通りの方法

再掲



自ら基盤モデルを  
スクラッチ開発

コスト、開発期間要  
業務・ML専門性必須

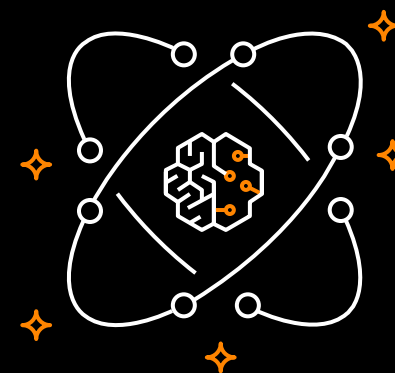
モデルプロバイダー



公開済み基盤モデルの  
活用・チューニング

運用に辿り着くまでに  
相当な分類作業要

モデルチューナー



生成系AI、API、基盤モデル  
提供ベンダの活用

データ管理・コスト管理要  
自社カスタマイズ余地小

モデルコンシューマー

# このセッションで得られること

## 生成系 AI の活用アイデアとシステムとして考慮すべき点を知る

### このセッションでやること

- 様々な公開済み基盤モデルを簡単に試す
- GUIからワンクリックで AWS 環境に 生成系 AI モデルをデプロイする
- 生成系 AI を活用したアプリケーションの紹介
  - 社内ドキュメントからの回答文生成
  - コールセンターでの通話要約文生成 **(日本語)**
- 開発者向けコード生成ソリューションの紹介

### このセッションではやらないこと

- Amazon Bedrock のデモ
- Amazon Titan のデモ

# アジェンダ

- 様々な公開済み基盤モデルを簡単に試す
- GUI からワンクリックで AWS 環境に 生成系 AI モデルをデプロイする
- 生成系 AI を活用したアプリケーションの紹介
  - 社内ドキュメントからの回答文生成
  - コールセンターでの通話要約文生成
- 開発者向けコード生成ソリューションの紹介

# アジェンダ

- 様々な公開済み基盤モデルを簡単に試す
- GUI からワンクリックで AWS 環境に 生成系 AI モデルをデプロイする
- 生成系 AI を活用したアプリケーションの紹介
  - 社内ドキュメントからの回答文生成
  - コールセンターでの通話要約文生成
- 開発者向けコード生成ソリューションの紹介

# Amazon SageMaker JumpStart

既存の基盤モデルに  
アクセスして開発・  
運用を始める



Products / Machine Learning / Amazon SageMaker JumpStart

## Getting started with Amazon SageMaker JumpStart

Amazon SageMaker JumpStart is a machine learning (ML) hub that can help you accelerate your ML journey. Explore how you can get started with built-in algorithms with pretrained models from model hubs, pretrained foundation models, and prebuilt solutions to solve common use cases. To get started, see documentation or example notebooks that you can quickly execute.

Reset Filters

Search: foundation models

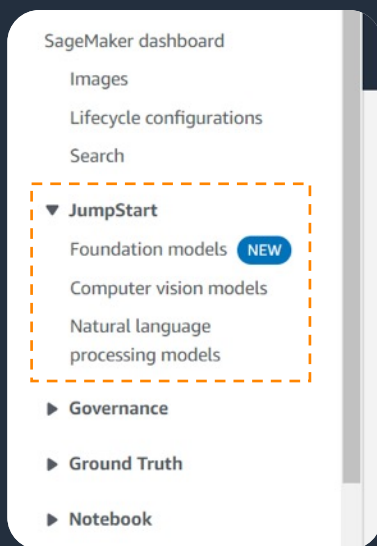
Sort By: Popularity

- Product Type
- Text Tasks
  - End-to-end Solution
  - Text Classification
  - Text Embedding
  - Text Generation
  - Text
  - Summarization
  - Named Entity Recognition
  - Question Answering

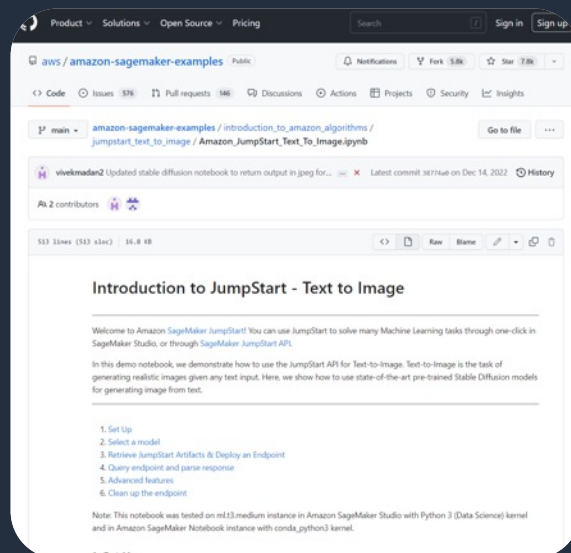
FOUNDATION MODEL	PREVIEW	FOUNDATION MODEL	FEATURED	FOUNDATION MODEL	FEATURED	FOUNDATION MODEL	FEATURED
<b>Proprietary Models</b> Various Providers Models from <b>AI21 Labs</b> , <b>Cohere</b> , and <b>LightOn</b> in preview. Sign-up for preview with JumpStart in us-east-1 or eu-west-1 SageMaker Console.		<b>Text to Image</b> stability.ai <b>Stable Diffusion 2</b> Stabilityai Model ID: model-bxt2img-stabilityai-stable-diffusion-v2. This is a text-to-image model from Stability AI and downloaded from HuggingFace. It takes a textual description as	<b>Text Generation</b> alexa <b>AlexaTM (20b)</b> Pytorch Model ID: pytorch-textgeneration1-alexa20b. AlexaTM 20B is a multitask, multilingual, large-scale sequence-to-sequence (seq2seq) model, trained on a mixture of Common Crawl	<b>Text Generation</b> 😊 <b>Bloom 1b7</b> Huggingface Model ID: huggingface-textgeneration-bloom-1b7. This is a Text Generation model built upon a Transformer model from Hugging Face. It takes a text string as input and predicts next words in the sequence. This model has BigScience Responsible AI License v1.0. Please read the [terms] ( <a href="https://huggingface.co/spaces/b">https://huggingface.co/spaces/b</a> )			
		Fine-tunable	Deploy Only	Deploy Only			

# SageMaker JumpStart: 基盤モデルを利用する 3 つの方法

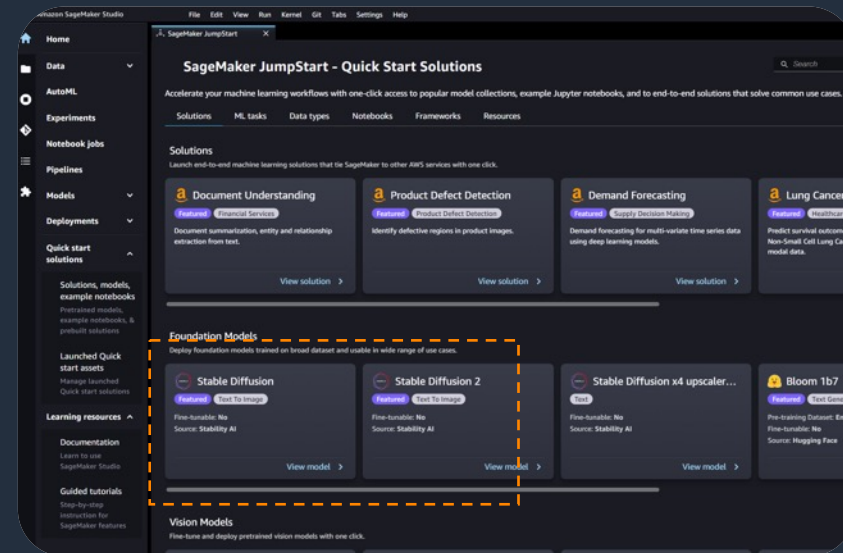
## SageMaker コンソール (プレビュー)



## SageMaker ノートブック



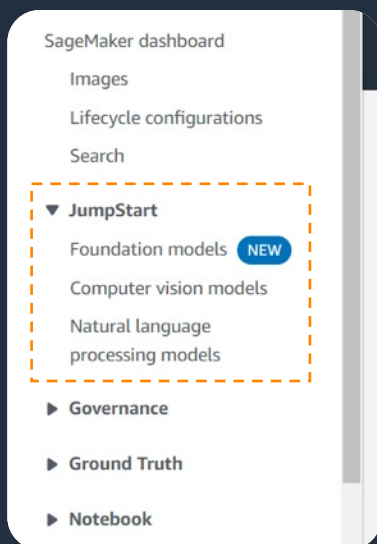
## SageMaker Studio ワンクリックデプロイ



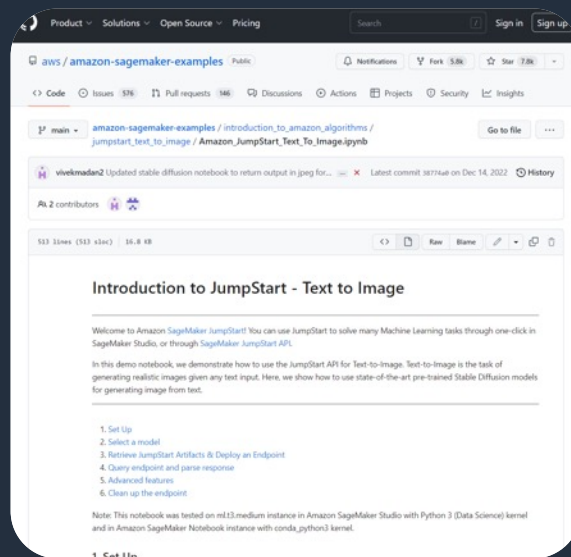


# SageMaker JumpStart: 基盤モデルを利用する 3 つの方法

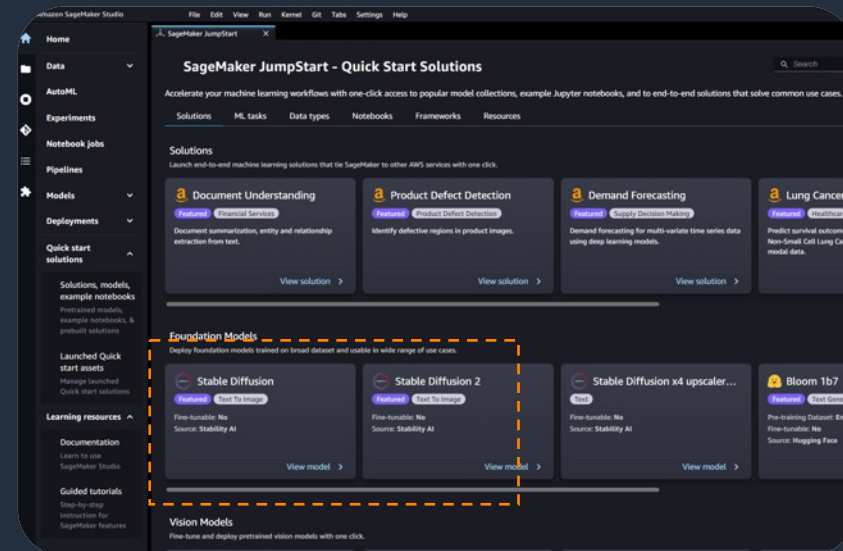
## SageMaker コンソール (プレビュー)



## SageMaker ノートブック



## SageMaker Studio ワンクリックデプロイ

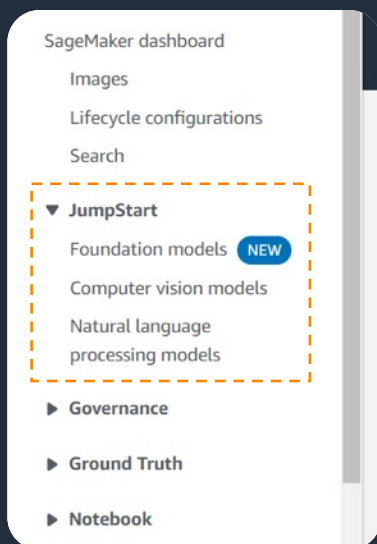


# 【デモ】 SageMaker JumpStart で基盤モデルを簡単に試す

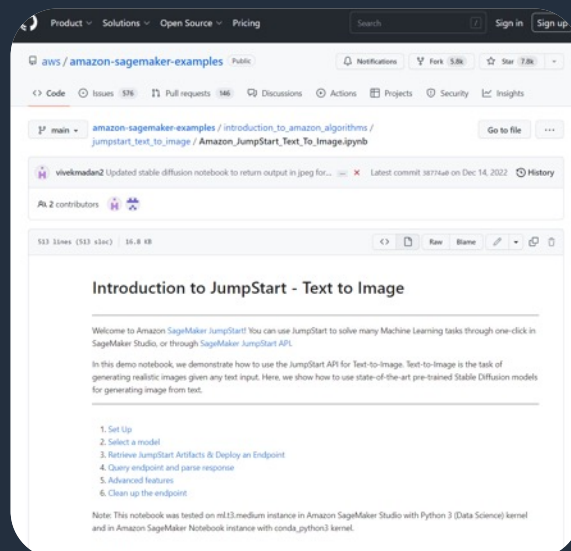
Live デモ実施

# SageMaker JumpStart: 基盤モデルを利用する 3 つの方法

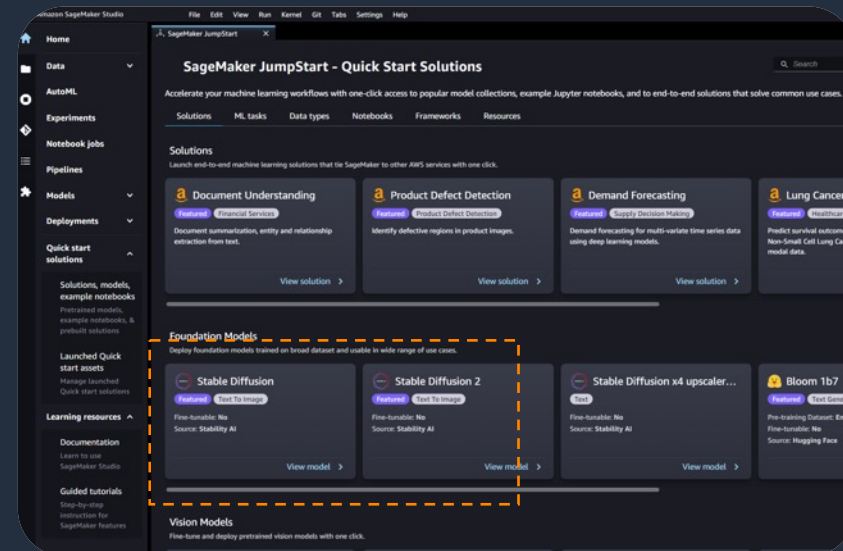
## SageMaker コンソール (プレビュー)



## SageMaker ノートブック



## SageMaker Studio ワンクリックデプロイ

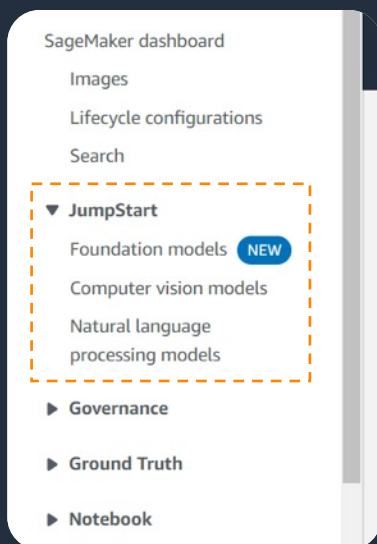


# 【デモ】 SageMaker JumpStart のノートブックを確認する

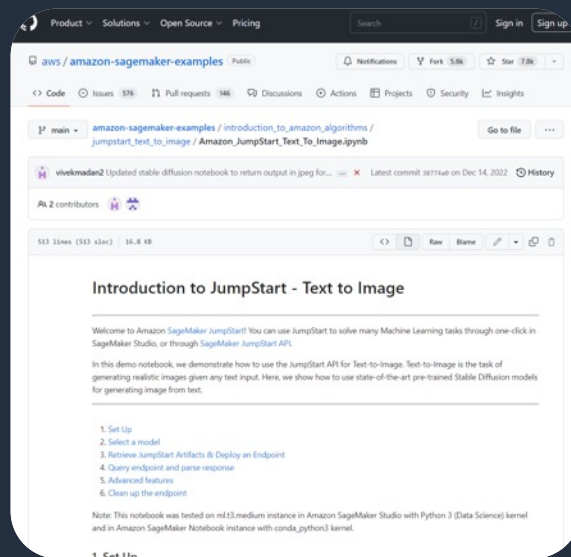
Live デモ実施

# SageMaker JumpStart: 基盤モデルを利用する 3 つの方法

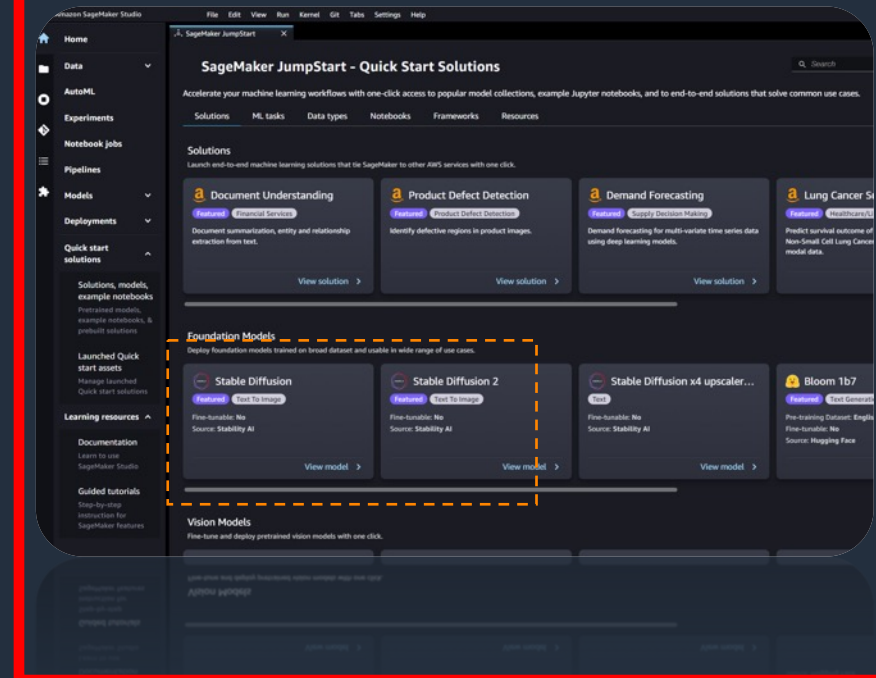
## SageMaker コンソール (プレビュー)



## SageMaker ノートブック



## SageMaker Studio ワンクリックデプロイ



【デモ】 SageMaker Studio からワンクリックデプロイする

Live デモ実施

基盤モデルを簡単に試したり、デプロイできるのはわかったけど…

**実際の業務やアプリケーションに  
どうやって使うの？**

# アジェンダ

- 様々な公開済み基盤モデルを簡単に試す
- GUI からワンクリックで AWS 環境に 生成系 AI モデルをデプロイする
- 生成系 AI を活用したアプリケーションの紹介
  - 社内ドキュメントからの回答文生成
  - コールセンターでの通話要約文生成
- 開発者向けコード生成ソリューションの紹介

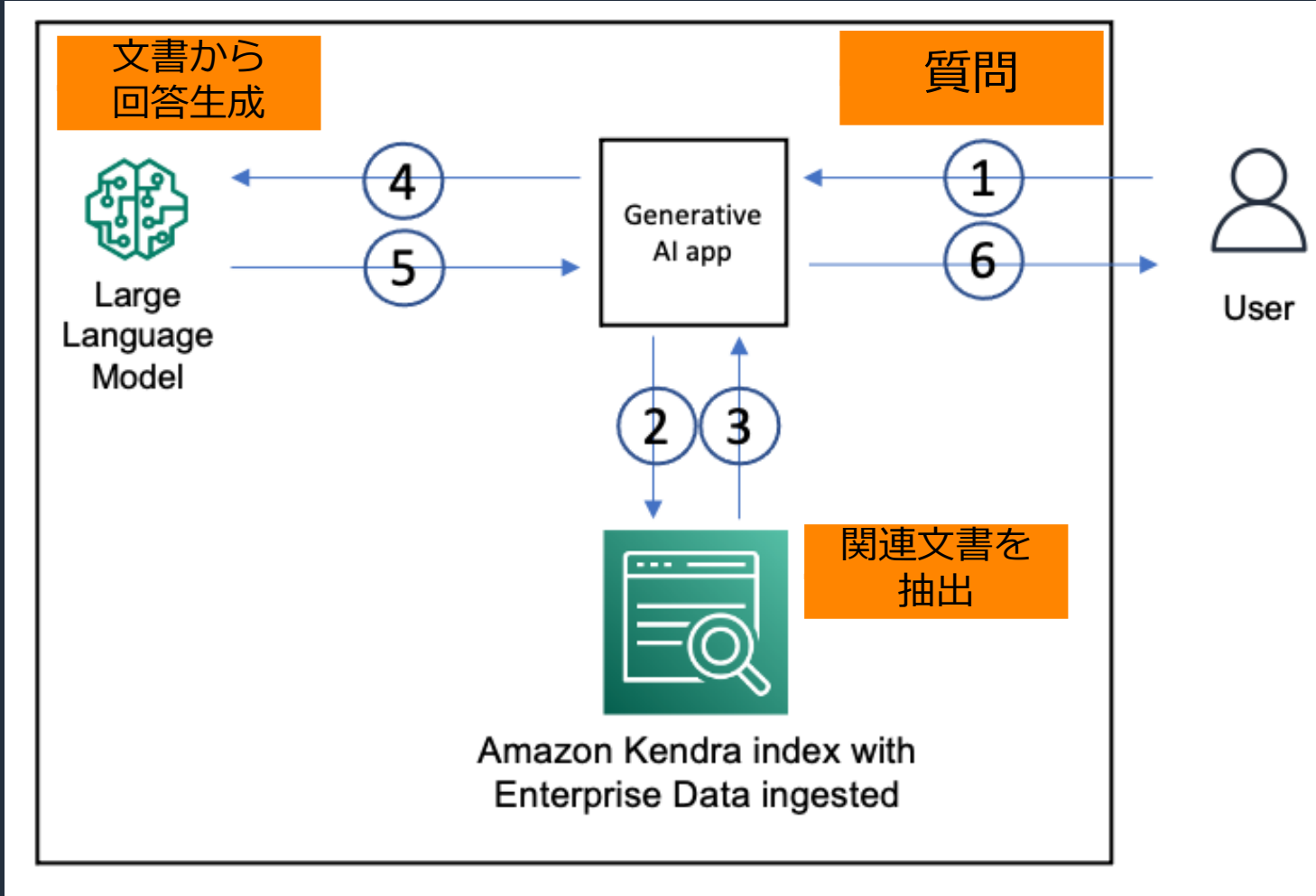


生成系 AI を活用したアプリケーション1

# 社内ドキュメントからの回答文生成

# アプリケーション概要 Retrieval Augmented Generation (RAG)

ユーザーからの質問に対して、社内ドキュメントを基にした回答を生成する

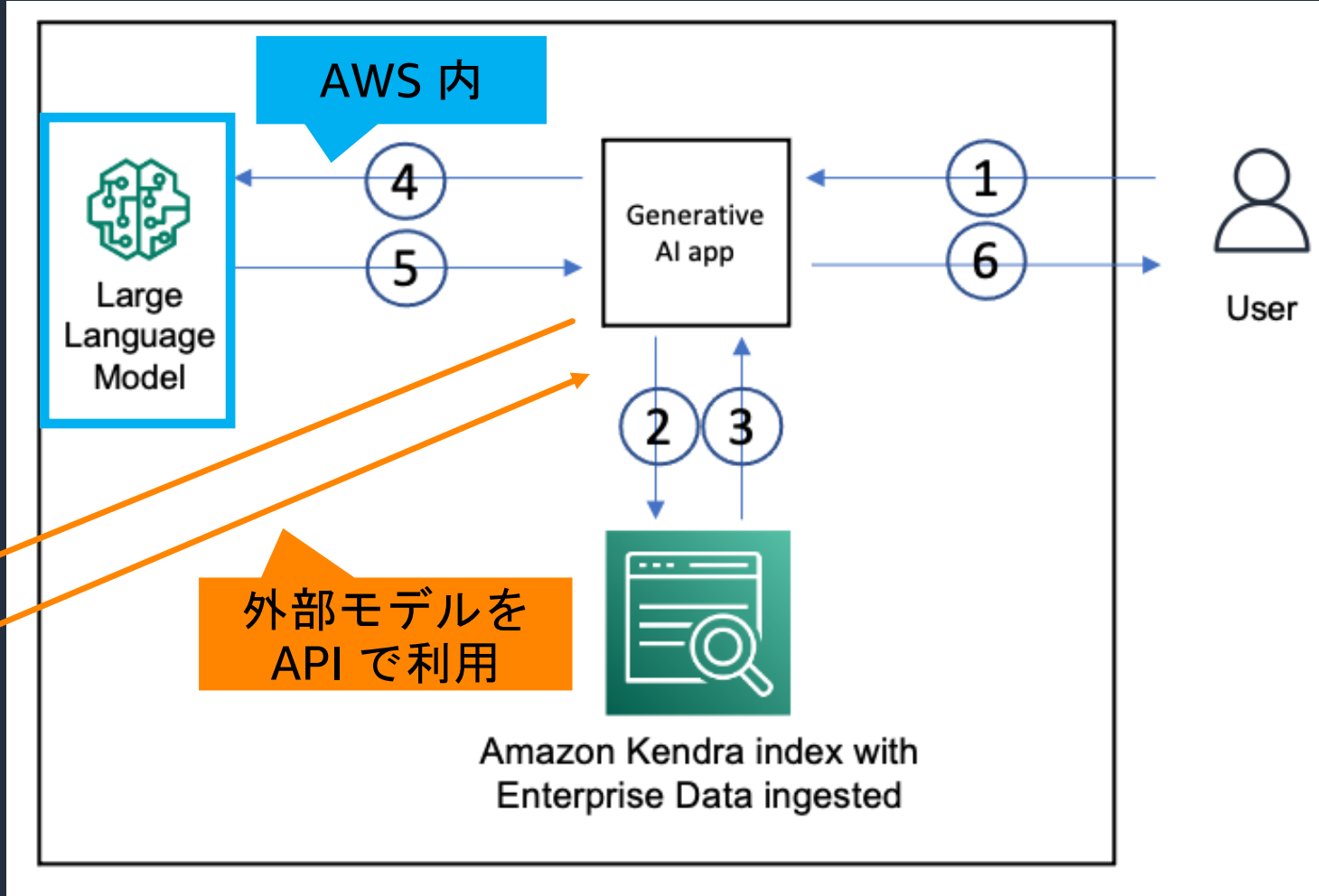


1. ユーザーが社内向け質問文をアプリケーションに入力
2. 3. アプリケーションは Amazon Kendra を利用して、関連するドキュメントを抽出
4. 5. アプリケーションは抽出されたドキュメントを生成系 AI にインプットし、生成系 AI が生成した回答を受領する
6. ユーザーはアプリケーションから生成された回答と、参考ドキュメントリンクを得る

<https://aws.amazon.com/blogs/machine-learning/quickly-build-high-accuracy-generative-ai-applications-on-enterprise-data-using-amazon-kendra-langchain-and-large-language-models/>

# 生成系 AI の利用パターン

デプロイされた自社環境のモデルを利用したり、外部モデルを API 経由で利用する



## 利用パターン1

自社 AWS 環境内の SageMaker エンドポイントにデプロイした生成系 AI を使う

## 利用パターン2

自社 AWS 環境にはない外部の生成系 AI を、API 経由で使う

# 【デモ】社内ドキュメントからの回答文生成

Live デモ実施

# アプリケーションのポイント

- 生成系 AI をコンポーネントとして利用できるようにアプリケーションを構成し、柔軟な試行錯誤を実現する
- 生成系 AI を他のサービスと組み合わせることで、機能を広げる
- Amazon SageMaker にモデルをデプロイすることで、自社 AWS 環境内でデータの流れを作る

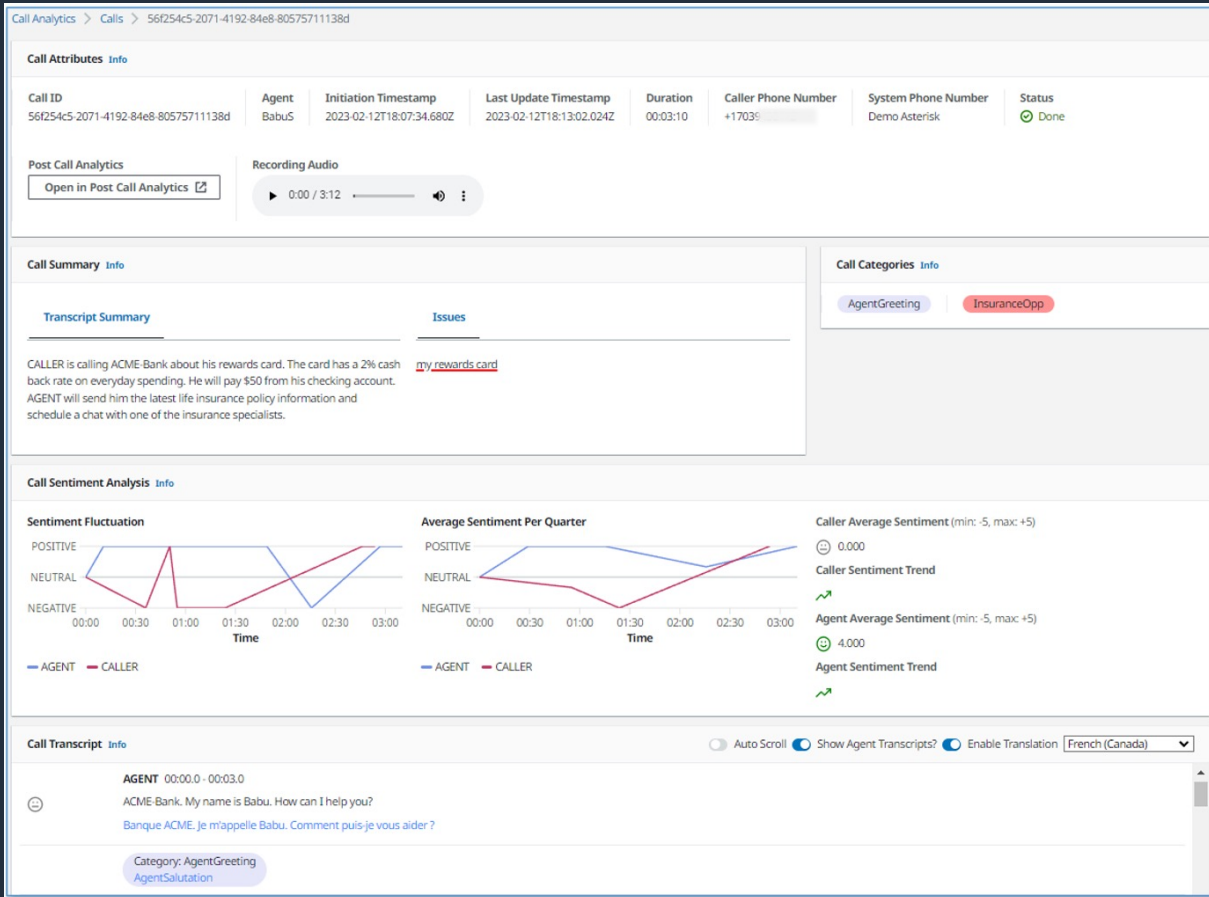
生成系 AI を活用したアプリケーション2

# コールセンターでの通話要約文生成

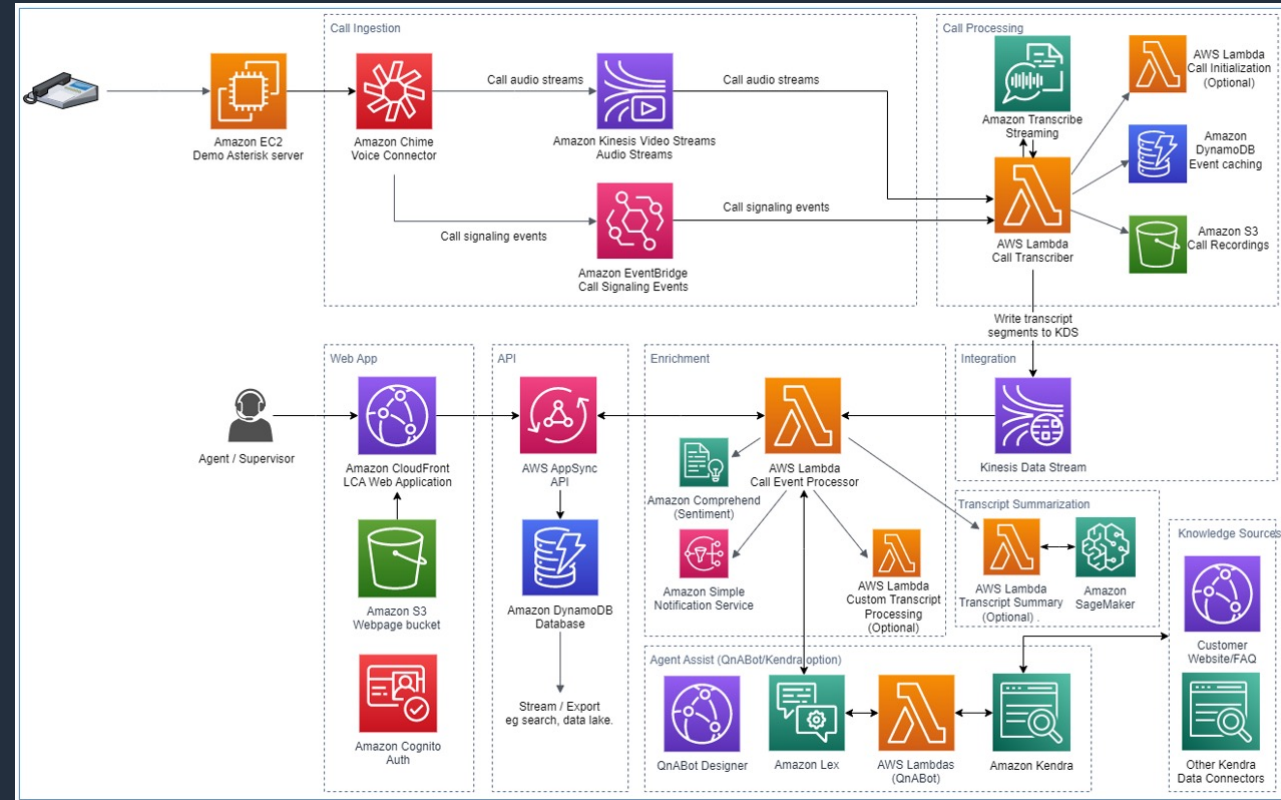
# アプリケーション概要

通話のリアルタイム書き起こし、感情分析、参考ドキュメント提示、**通話内容要約**などの機能を提供

## 分析ダッシュボード



## アーキテクチャ



<https://aws.amazon.com/blogs/machine-learning/live-call-analytics-and-agent-assist-for-your-contact-center-with-amazon-language-ai-services/>



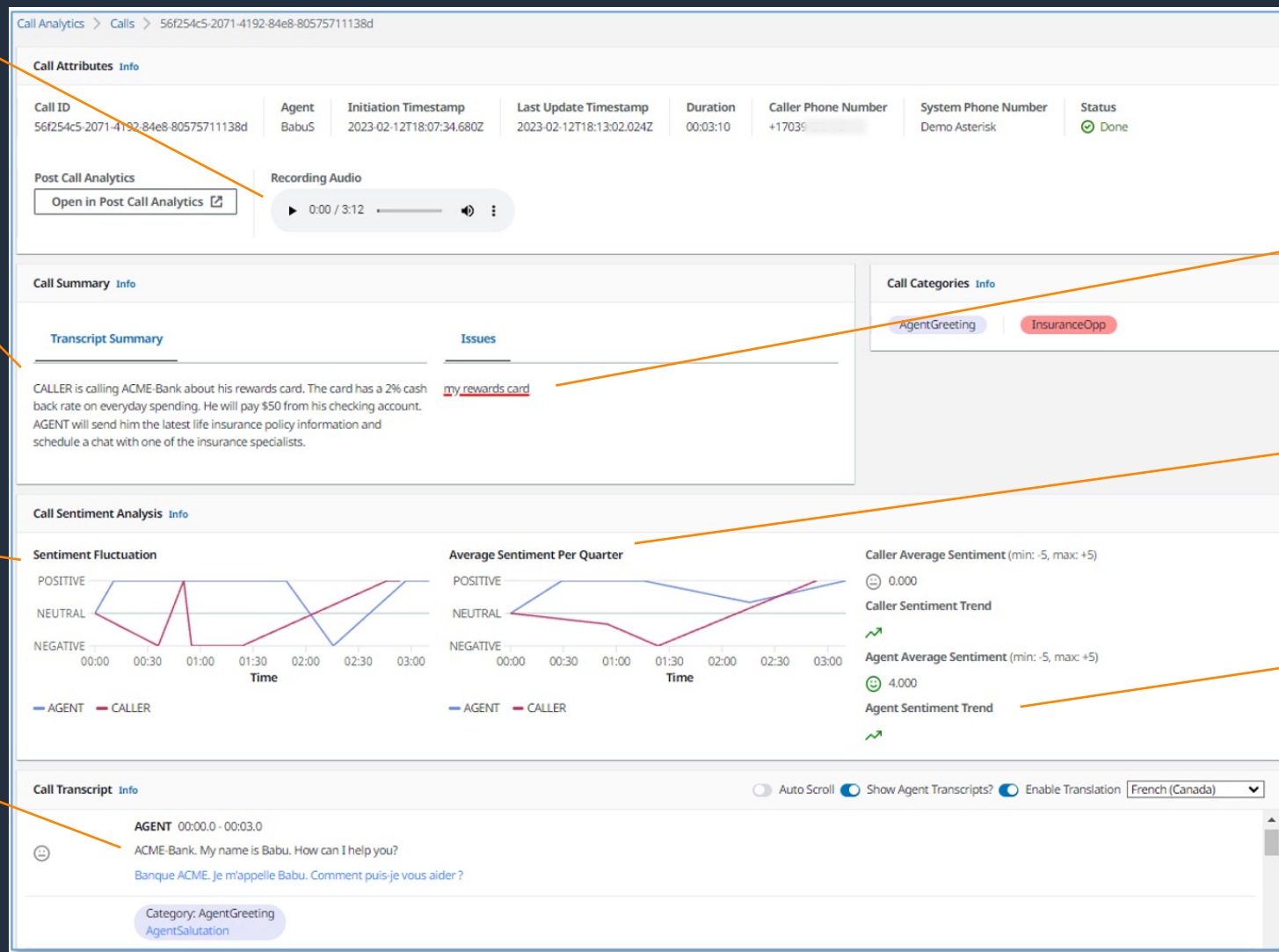
# 分析ダッシュボードの説明

実際の通話音声を確認

生成系 AI による通話の要約文

発話ごとの感情推移

リアルタイムでの音声書き起こし



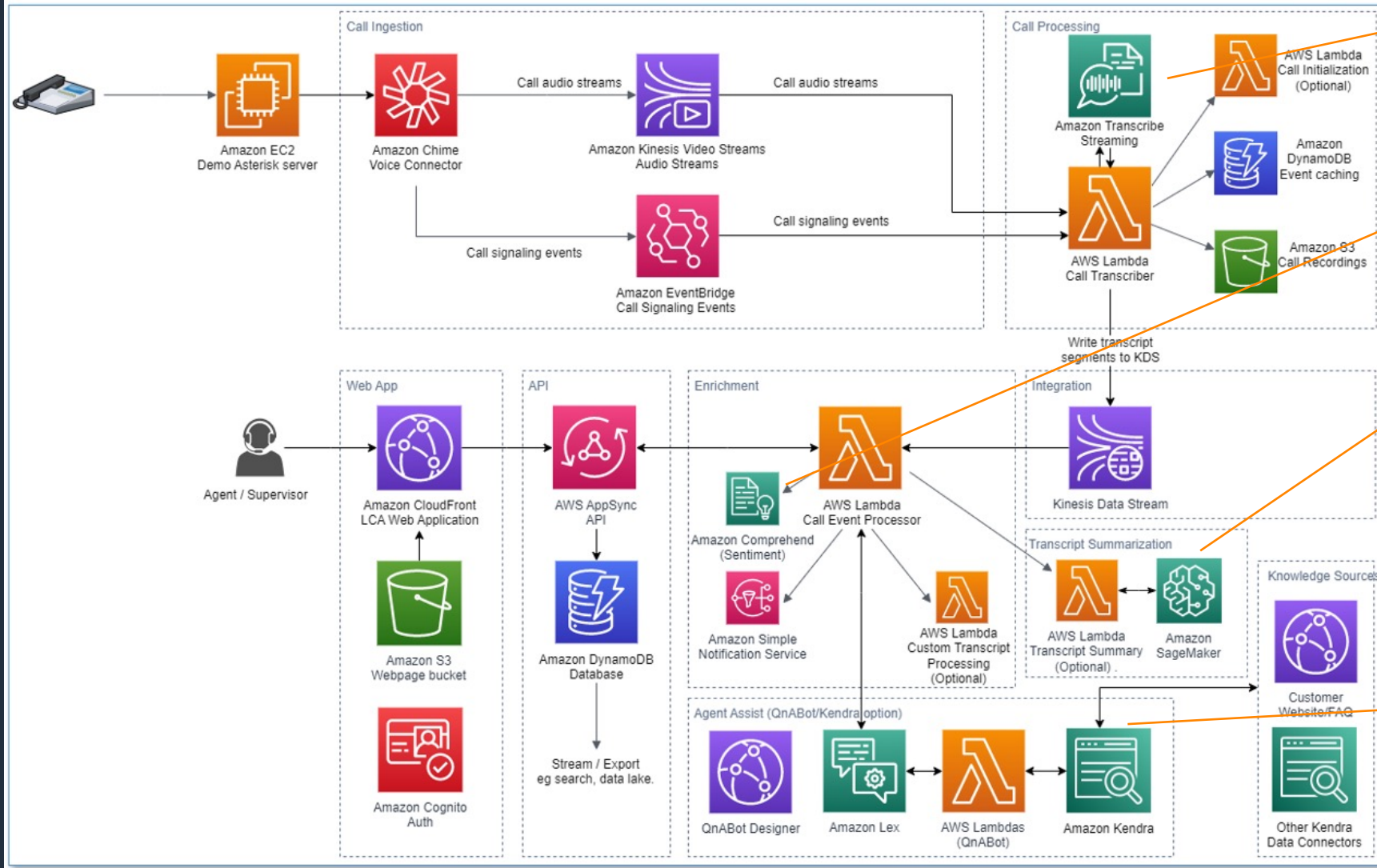
問題を検知して提示  
(英語のみ対応)

感情の15秒平均推移

感情のセッション平均スコアと推移トレンド



# アーキテクチャの説明



リアルタイム音声  
書き起こし  
(Amazon Transcribe)

感情分析  
(Amazon Comprehend)

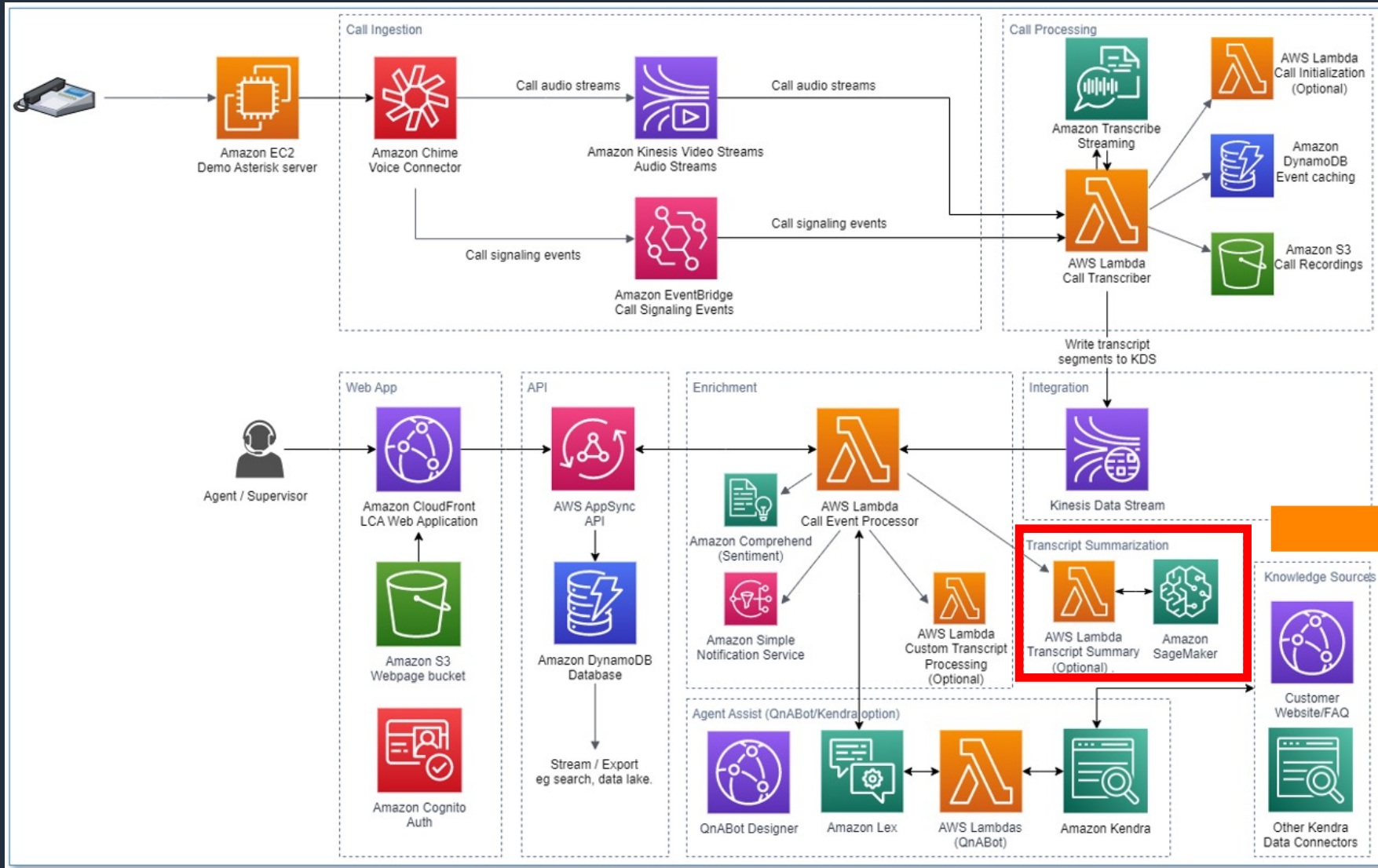
通話要約文生成モデル  
(SageMaker エンドポイント)

参考ドキュメント提示  
(Amazon Lex  
& Amazon Kendra)

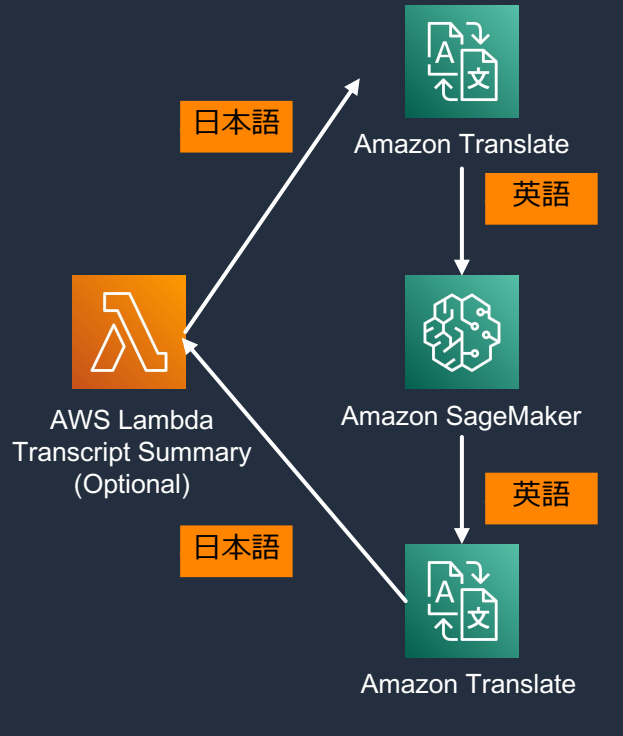
# 【デモ】 コールセンターでの通話要約文生成

録画デモ実施

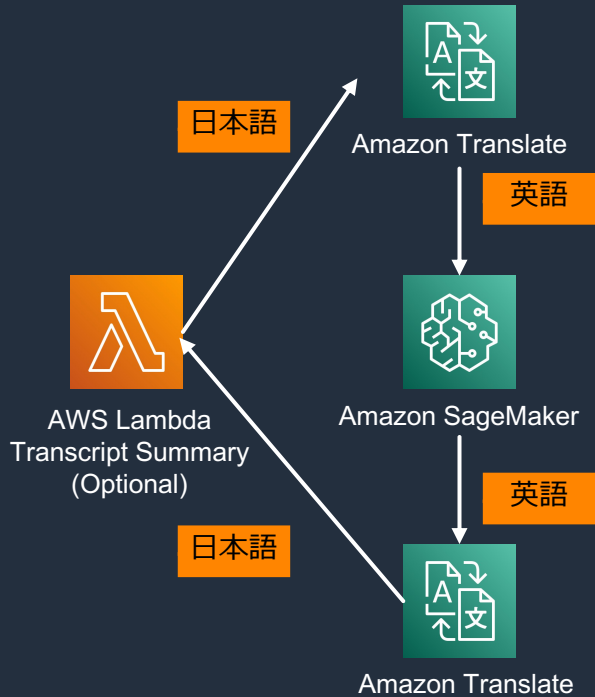
# 生成系 AI モデルの出力を日本語に変換して利用する



翻訳機能を入力・出力に加える



# 参考：日本語出力変換 – 修正した Lambda 関数



```

def lambda_handler(event, context):
    print("Received event: " + json.dumps(event, indent=2))

    # Setup model input data using text (utterances) received from LCA
    data = json.loads(json.dumps(event))
    callId = data['CallId']
    transcript_response = get_transcripts(callId)
    transcript_data = transcript_response['Payload'].read().decode()
    transcript_json = json.loads(transcript_data)
    # print(transcript_json)

    translate = boto3.client('translate')
    response = translate.translate_text(
        Text=transcript_json['transcript'],
        SourceLanguageCode='ja',
        TargetLanguageCode='en'
    )

    #payload = {'inputs': transcript_json['transcript']}
    payload = {'inputs': response['TranslatedText']}

    summaryText = ""
    try:
        data = json.dumps(event)
        response = runtime.invoke_endpoint(EndpointName=ENDPOINT_NAME,
            ContentType='application/json',
            Body=bytes(json.dumps(payload), 'utf-8'))

        # print(response)
        result = json.loads(response['Body'].read().decode())

        if len(result) > 0:
            summary = result[0]
            # print("Summary: " + summary['generated_text'])
            summaryText = summary['generated_text']
        else:
            print("No Summary")
            summaryText = "No summary"

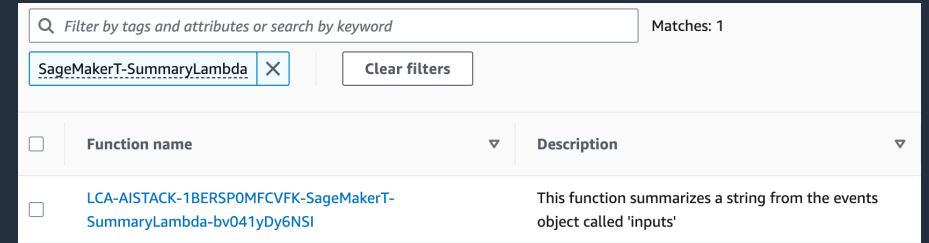
    except Exception as e:
        print("An exception occurred:", e)
        summaryText = ""

    response = translate.translate_text(
        Text=summaryText,
        SourceLanguageCode='en',
        TargetLanguageCode='ja'
    )

    #return {"summary": summaryText}
    return {"summary": response['TranslatedText']}
  
```

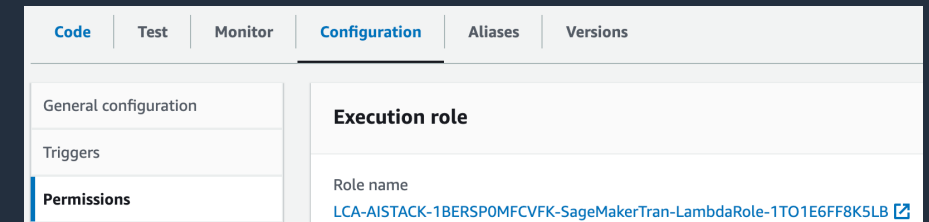
## <Lambda 関数設定手順>

1) Lambda 関数の検索窓で、“SageMakerT-Lambda”で検索



2) index.py の、lambda\_handler関数内を左の通り変更

3) Lambda の Execution role に “TranslateFullAccess” を付与



# 参考：その他アプリケーション構築のTIPS

- CloudFormation テンプレートの設定パラメータ
  - Admin Email Address : eメールアドレス
  - Authorized Account Email Domain(s) : eメールアドレスのドメイン
  - Allowed CIDR Block for Demo Softphone : <接続元 (自宅) のIP>/32
  - Call Audio Recordings Bucket Name : S3バケットを事前に作成し、指定
  - Transcribe API mode : standard
  - Language for Transcription : ja-JP
  - End of Call Transcript Summary : SAGEMAKER
- Soft Client : Zoiper5 を利用 (オペレータ・AGENT側)
  - <https://github.com/aws-samples/amazon-transcribe-live-call-analytics/blob/develop/lca-chimevc-stack/Asterisk.md>
- Skype (カスタマー・CALLER側)
  - 設定した Zoiper5 にダイヤル

# アプリケーションのポイント

- 生成系 AI をコンポーネントとして利用できるようにアプリケーションを構成し、柔軟な試行錯誤を実現する
- 生成系 AI を他のサービスと組み合わせることで、機能を広げる
- Amazon SageMaker にモデルをデプロイすることで、自社 AWS 環境内でデータの流れを作る

NEW

- 翻訳機能と組み合わせることで、英語のみ対応のモデルから日本語出力を得ることも可能（要件を満たすかは検討）

NEW

- 要件を満たすには、どこまで大規模なモデルが必要なのか検討する



# アジェンダ

- 様々な公開済み基盤モデルを簡単に試す
- GUI からワンクリックで AWS 環境に 生成系 AI モデルをデプロイする
- 生成系 AI を活用したアプリケーションの紹介
  - 社内ドキュメントからの回答文生成
  - コールセンターでの通話要約文生成
- 開発者向けコード生成ソリューションの紹介

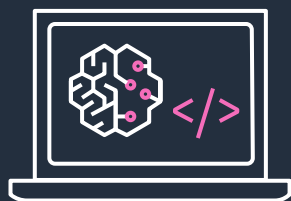
NEW

# Amazon CodeWhisperer

AI コーディング・コンパニオンで  
アプリケーションをより速く、  
より安全に構築



# Amazon CodeWhisperer: 一般公開され、個人開発者は無料で使用できるようになりました!



コードの候補を  
リアルタイムで生成



コードをスキャンして、発見の  
困難な脆弱性がないか調べ  
る



オープンソースのトレーニング  
データに似たコードにフラグを  
たて、フィルタリングする

Amazonによるプレビュー中、生産性向上課題に対し、CodeWhispererを使用した参加者は、CodeWhispererを使用しなかった参加者に比べて、タスクの正常完了率は27%高く、平均で57%も速く完了しました。

# 【デモ】 Amazon CodeWhisperer

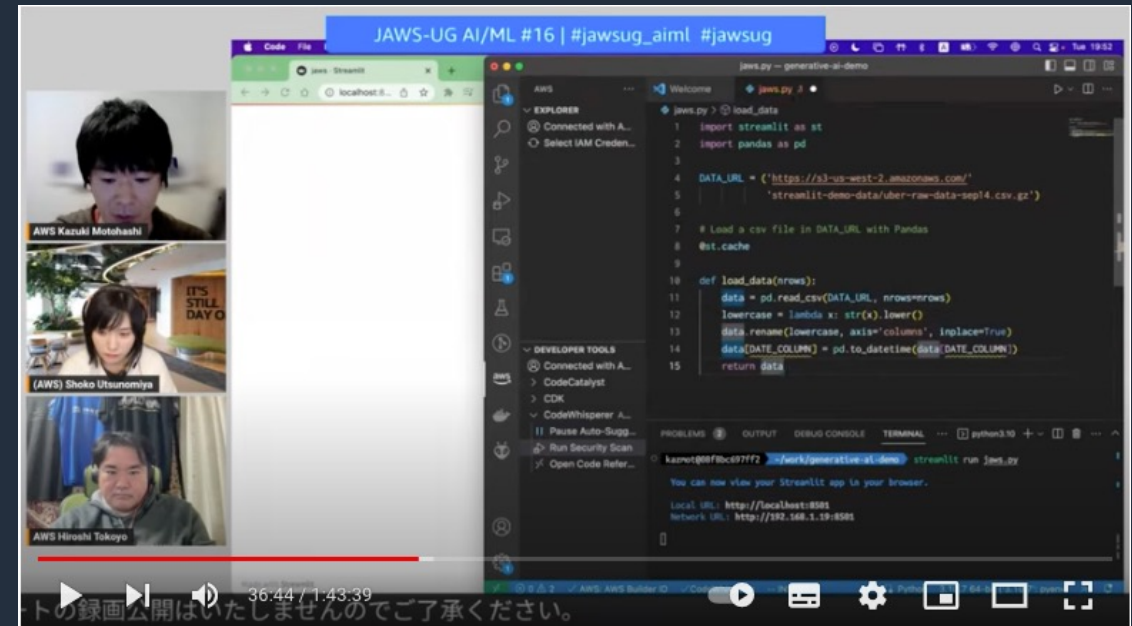
```
SQSIdentityFunction.java M src/main/java/com/amazonaws/services/sqs/SQSIdentityFunction.java
package com.amazonaws.services.sqs;

import java.util.Map.Entry;
import java.util.stream.Collectors;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.SQSEvent;
import com.amazonaws.services.lambda.runtime.events.SQSEvent.MessageAttribute;
import com.amazonaws.services.lambda.runtime.events.SQSEvent.SQSMessage;
import com.amazonaws.services.sqs.model.MessageAttributeValue;
import com.amazonaws.services.dynamodbv2.AmazonDynamoDBClientBuilder;
import com.amazonaws.regions.Regions;

// */
```

参考: YouTube でのLiveデモもぜひご覧ください



JAWS-UG AI/ML #16 Generative AI

<https://www.youtube.com/watch?v=PkJenNAXtYs&t=2170s>

<https://aws.amazon.com/blogs/aws/amazon-codewhisperer-free-for-individual-use-is-now-generally-available/>



# まとめ

## 生成系 AI の活用アイデアとシステムとして考慮すべき点を知る

- 要件を満たす生成系 AI モデルを**目利き**する  
→ 手軽に試すためなら、SageMaker JumpStart
- 他機能と組み合わせてアプリケーションでできることを拡張するために、**柔軟性**の高いシステム構成を考慮する  
→ ビルディングブロックで構成できる AWS を活用
- SageMaker エンドポイントに生成系 AI モデルをデプロイすることで、データの流れを**自社 AWS 環境内で完結**することが可能  
→ Amazon SageMaker はデプロイ後の運用(MLOps)機能もサポート



# Thank you!