



# Amazon Bedrock

## Workshop

Arnaud Lauer (he/him)

Sr. Partner Solutions Architect, EMEA PS  
IBM Nordcloud – Strategic Alliance  
Amazon Web Services

# Agenda

- Amazon Bedrock
- Security
- Architecture patterns
- Hands-on labs
- Extras/next step

# Amazon Bedrock

THE EASIEST WAY TO BUILD AND SCALE GENERATIVE AI APPLICATION WITH FMS



Access a range of leading FMs via a single API



Privately customize FMs with your own data



Enable data security and compliance



Build agents that execute complex business tasks by dynamically invoking APIs

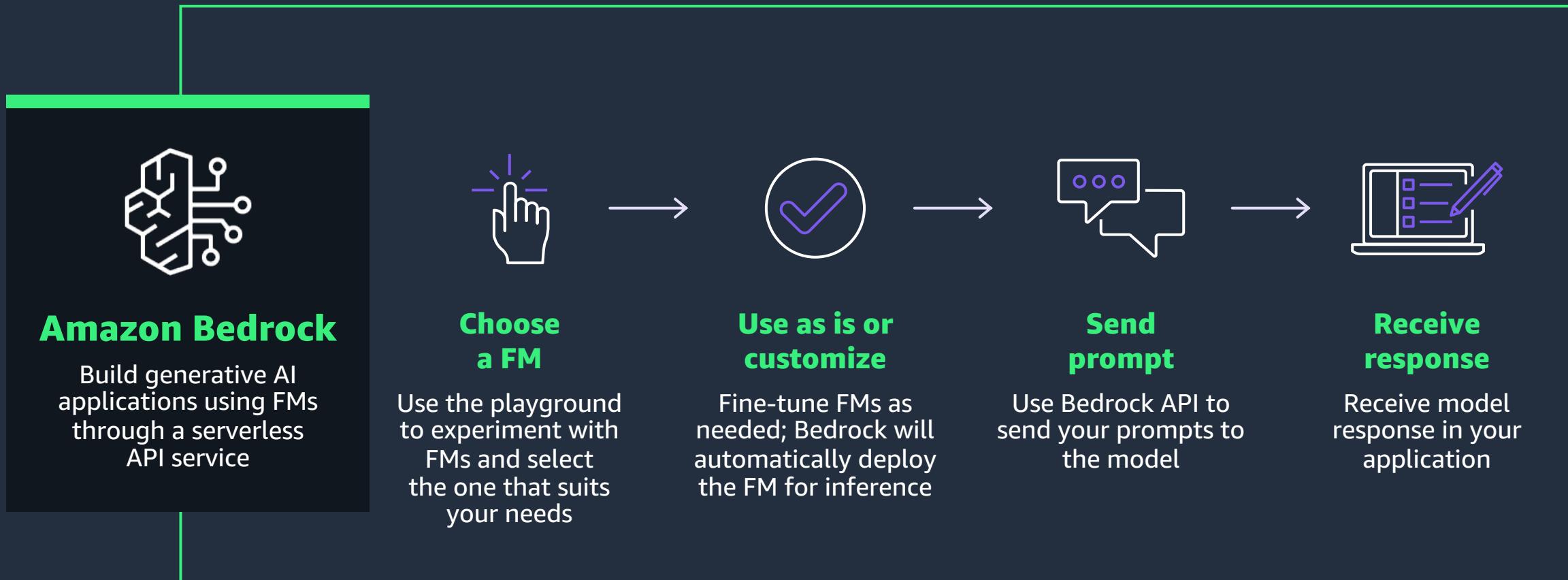


Extend the power of FMs with your data using retrieval augmented generation (RAG)



Get the best price performance without managing infrastructure

# How Amazon Bedrock works



# Amazon Bedrock is now generally available!



All customers with an AWS Account now have access to Amazon Bedrock



HIPAA eligible & GDPR compliant



Pre-announcing Llama 2 – 13B and 70B



# Foundation models

# Amazon Bedrock supports leading foundation models

---



## Amazon Titan

Text summarization, generation, classification, open-ended Q&A, information extraction, embeddings and search

## AI21labs

### Jurassic-2

Multilingual LLMs for text generation in Spanish, French, German, Portuguese, Italian, and Dutch

## ANTHROP\IC

### Claude 2

LLM for conversations, question answering, and workflow automation based on research into training honest and responsible AI systems



## Command and embed

Text generation model for business applications and embeddings model for search, clustering, or classification in 100+ languages

## Meta

### Llama 2

Fine-tuned models ideal for dialogue use-cases and language tasks

COMING SOON

## stability.ai

### Stable Diffusion

Generation of unique, realistic, high-quality images, art, logos, and designs

# Titan Text Embeddings



## Titan Text Embeddings

NEW

v2.0

Translates text inputs (words, phrases) into numerical representations (embeddings). Comparing embeddings produces more relevant and contextual responses than word matching.

Max Tokens: **8,000**

Output Vectors: **1,536**

Language: **Multilingual** (25 languages)

modelID: *amazon.titan-embed-text-v1*



## Highlights

- Titan Text Embeddings offers fast, cost effective, high-performance, accurate embeddings in 25 languages.
- Optimized for text retrieval tasks, semantic similarity and clustering.
- Applications of this model includes semantic search and personalization.



## Titan Text Express

v1.0

Preview

Generative LLM for tasks such as summarization, text generation, classification, open-ended Q&A, and information extraction. Trained on many different programming languages as well as rich text format like tables, JSON, and csv among others.

Max Tokens: **4,000**

modelID: *amazon.titan-text-express-v1*



## Other news!

- Titan Text Lite and Titan Text Agile are coming soon!

# Jurassic 2



## Jurassic-2 Mid

Mid-sized model, carefully designed to strike the right balance between exceptional quality and affordability. Jurassic-2 Mid can be applied to any language comprehension or generation task including question answering, summarization, long-form copy generation, advanced information extraction and many others.

Max Tokens: 8,191

modelID: ai21.j2-mid-v1



## Jurassic – 2 Ultra

AI21's most powerful model offering exceptional quality. Apply Jurassic-2 Ultra to complex tasks that require advanced text generation and comprehension. Popular use cases include question answering, summarization, long-form copy generation, advanced information extraction, and more.

Max Tokens: 8,191

modelID: ai21.j2-ultra-v1



## Claude 2

Anthropic's most powerful model, which excels at a wide range of tasks from sophisticated dialogue and creative content generation to detailed instruction following.

Max Tokens: 100,000<sup>+</sup>

modelID: anthropic.claude-v2



## Claude Instant

A faster and cheaper yet still very capable model, which can handle a range of tasks including casual dialogue, text analysis, summarization, and document question-answering.

Max Tokens: 100,000<sup>+</sup>

modelID: anthropic.claude-instant-v1

+ Claude v2, and Claude Instant models supporting 100K tokens size are only available in N. Virginia, and Oregon regions at this time. Claude end points in Tokyo, and Frankfurt regions will cap at 44K tokens.

# Cohere is now available on Bedrock



## Cohere Command v14.6

NEW

Command is a text generation model for business use cases. Command is trained on data that supports reliable business applications, like text generation, summarization, copywriting, dialogue, extraction, and question answering.

Max Tokens: **4,096**

Model ID: *cohere.command-text-v14*

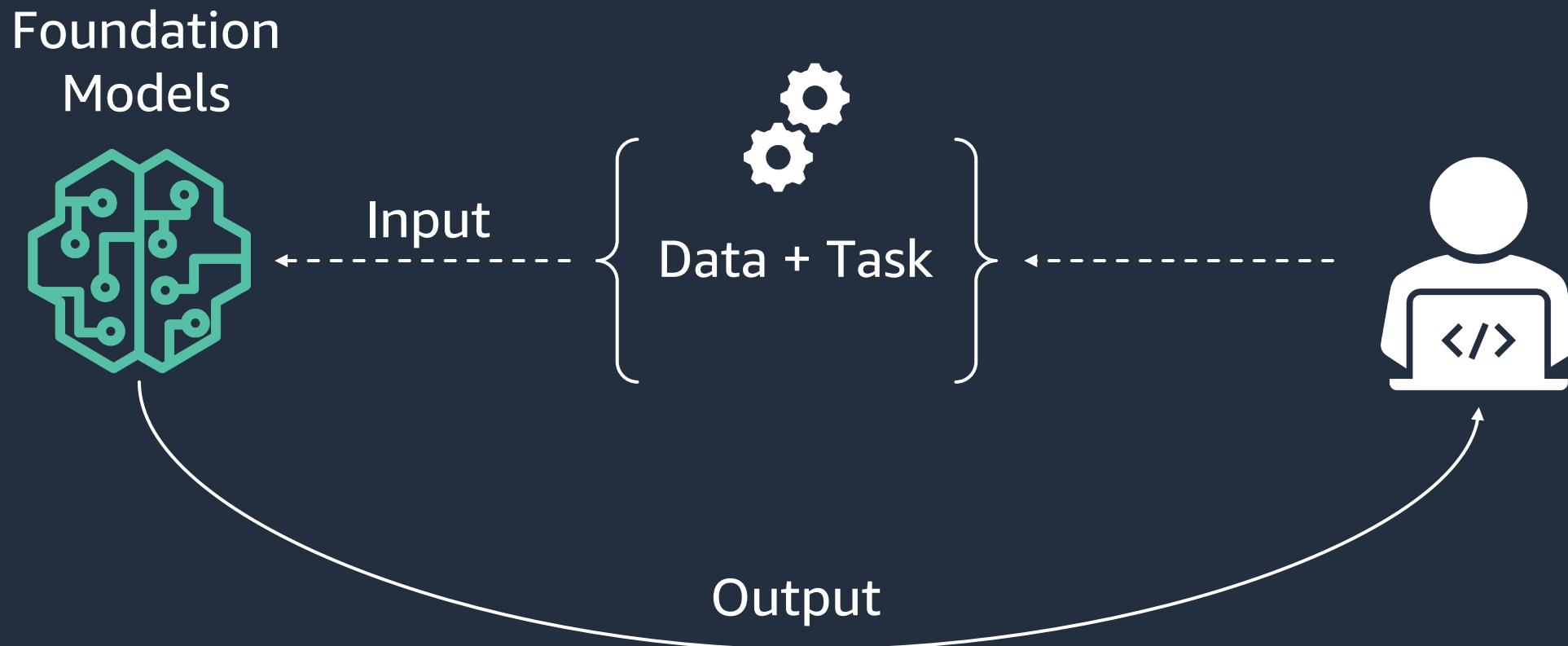


## Other news!

- Cohere Command Lite and Cohere Embed coming soon!

# Prompt engineering

# Prompt engineering, new way of using ML!



# Elements of a Prompt

## Instruction

Task description or instruction on how the model should perform

## Context

Additional/external information to steer the model performance

## Input Data

The input/question that the model needs to provide output for

## Output Indicator

The indicator/format the model needs to provide output with

Instructions

Context

User Input

Output Indicator

The screenshot shows a user interface for generating AI responses. At the top, there's a logo for 'Amazon' and dropdown menus for 'Amazon' and 'Titan XL v1.01'. Below that, a descriptive text states: 'Powerful, general-purpose models pretrained on large datasets. Titan FMs are powerful, general-purpose models that can be used as-is or customized to perform specific tasks.' On the left side, four green boxes with white text labels identify the interface elements: 'Instructions', 'Context', 'User Input', and 'Output Indicator'. Red arrows point from each of these labels to their corresponding sections in the main content area. The 'Instructions' section contains the text: 'Act as an IT technical expert providing customer service. Consider the Context below to answer the user's questions with a friendly tone. Answer in English in 2 sentences or less providing instructions.' The 'Context' section contains: 'Context: You work in the Support line of a technology company that commercializes Android smartphones. The user is calling because the phone is not charging'. The 'User Input' section contains: 'User: Hi, how can I fix my phone?'. The 'Output Indicator' section contains: 'Assistant: I will try my best to assist you with this. Can you please tell me the model of your phone and what version of the Android operating system it is running?'. At the bottom of the interface are several small circular icons for interacting with the response.

# Experiment with models



## Playground experience

- You can choose from multiple models and providers
- Easy to use. Just enter text in the Prompt field, then choose Run. In the Response panel, the console displays the response from the model.
- You can adjust inference configuration parameters and then re-run your prompt.

# Examples



## Try out real-life use cases

- The Amazon Bedrock console displays up to 20 examples for each model provider.
- You can filter the list of examples by one or more of the following attributes: modality, provider name, model name, or example category.
- You can easily run the example by opening it in playground.

# Experiment with models to try out real-life use cases

Examples (15) Info

Search in examples

**Titan Text Large v1.01**

**Action items from a meeting transcript**

This prompt creates a list of action items from a meeting transcript.

Summarization

**Claude Instant v1.3**

**Content Generation**

An example prompt to generate a paragraph from instructions

Text generation

**Stable Diffusion XL v2.2.2**

**Create an image**

This prompt creates an image based on a short description

Image gen

**Titan Text Large v1.01**

**Creating a table of product descriptions**

This is a prompt for generating product descriptions that incorporate keywords

Open ended text generation

**Creating a table of product descriptions**

This is a prompt for generating product descriptions that incorporate keywords

**Open in Playground**

 Titan Text Large v1.01  
By Amazon

**Prompt**

Product: Sunglasses.  
Keywords: polarized, designer, comfortable, UV protection, aviators.

Create a table that contains five variations of a detailed product description for the product listed above, each variation of the product description must use all the keywords listed.

**Inference configuration**

Temperature	0
Top P	1
Response length	4096
Stop sequences	N/A

**API request**

**Copy**

```
1 {
2   "modelId": "amazon.titan-tg1-large",
3   "contentType": "application/json",
4   "accept": "*/*",
5   "body": {
6     "inputText": "Product: Sunglasses. Keywords: polarized, de",
7     "textGenerationConfig": {
8       "maxTokenCount": 4096,
9       "stopSequences": [],
10      "temperature": 0,
11      "topP": 1
12    }
13 }
```

# Access Bedrock via the AWS console

Amazon ▾ Titan Large v1.01 ▾

Powerful, general-purpose models pretrained on large datasets, Titan FMs are powerful, general-purpose models that can be used as-is or customized to perform specific tasks.  
[View details](#) ⓘ

**What is generative AI?**

Generative AI is a form of artificial intelligence that can create text, images, or videos based on user prompts. It uses machine learning algorithms to analyze data and generate new content that is similar to the input it has been given.

One of the main applications of generative AI is in natural language processing (NLP). Generative AI can be used to create text that is similar to the way a human writes, and can also be used to generate translations between different languages. It can also be used to create social media posts, reviews, and other types of content that require human-like writing skills.

Another application of generative AI is in image and video creation. Generative AI can be used to create new images or videos that are similar to those that have been seen before. It can also be used to create animations or to modify existing images or videos in creative ways.

There are also generative AI applications in music and sound. Generative AI can be used to create new music or to modify existing music files in creative ways. It can also be used to create sound effects or to modify existing sound files.

Overall, generative AI is a powerful tool that has the potential to revolutionize the way we create and consume content. It can help to automate many of the tasks that were previously done by humans, and can also help to create new forms of content that were not possible before.

**Inference configuration** [Reset](#)

**Randomness and diversity** [Info](#)

Temperature

Top P

**Length** [Info](#)

Response length

Stop sequences  
  
Only | allowed (maximum 20 characters)

**Run** ⏪ ⏴ [View API request](#) [Save](#)

Stability AI ▾

Stable Diffusion XL v2.2.2 ▾

Deep learning, text-to-image model used to generate detailed images conditioned on text descriptions, inpainting, outpainting, and generating image-to-image translations.

[View details](#)

**Prompt**

photo taken from above of an italian landscape. cloud is clear with few clouds. Green hills and few villages, a lake

**Inference configuration** [Info](#)

Reset

Prompt strength

Generation step

Seed

**Response**



# Chat Playground

Quickly evaluate Foundation models using the Chat playground

Provide instructions to assume a role throughout the conversation in its entirety.

Prompts and responses are erased once the browser session is complete, or by clicking on 'Start new chat'

The screenshot shows the 'Chat playground' interface with the 'Streaming' checkbox checked. At the top, there are dropdown menus for 'AI' (set to Anthropic), 'Anthropic' (selected), 'Claude V2 v2' (available), and a 'Start new chat' button. Below this, a note states: 'Next-gen AI assistant trained on helpful, honest, and harmless AI systems. Claude can help with summarization, search, creative writing, Q&A, coding, as well as take direction. Note: The terms "Human:" and "Assistant:" will automatically be added to all prompts using Anthropic models to improve your results.' A link to 'View details' is provided. The main conversation area has an instruction: 'Instructions update: Use a tone that is technical and scientific.' It shows a Human message: 'Hello, who are you? Can you tell me about the creation of blackholes?' followed by an Assistant response: 'Here is some information about the creation of black holes:'. The Assistant continues: 'Black holes are formed when massive stars collapse at the end of their life cycles. When a star with a mass greater than about 3 times the mass of the sun runs out of fuel to burn, it can no longer produce enough heat and radiation pressure to counteract the gravitational force inward. This causes the star to collapse rapidly under its own weight. As it collapses, the gravitational force increases further, speeding up the collapse. If the remaining core of the star after collapse has a mass greater than about 3 solar masses, it will form a black hole. The collapse continues until the core reaches an extremely dense state called a singularity, where the laws of physics as we know them break down. At the singularity, the gravitational field becomes infinite. Surrounding the singularity is the event horizon, the boundary past which nothing, not even light, can escape the black hole's gravitational pull.' Below this, another Human message reads: 'Some ways black holes can form:'.

# Amazon Bedrock API

## Single API to build with generative AI



# Single API to build with generative AI

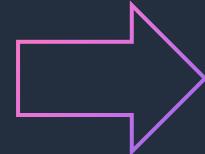


## Bedrock core API: InvokeModel

- Pass the model ID, type of content, and body of the request
  - Body includes the prompt and execution parameters
  - Returns model response and metadata
- Handles text-to-text, text-to-image, image-to-image, and more
- Supports current and future Amazon Titan models, third-party models, and even fine-tuned models

# Bedrock core API: InvokeModel

```
bedrock.invoke_model(  
    modelId = model_id,  
    contentType = "...",  
    accept = "...",  
    body = body)
```

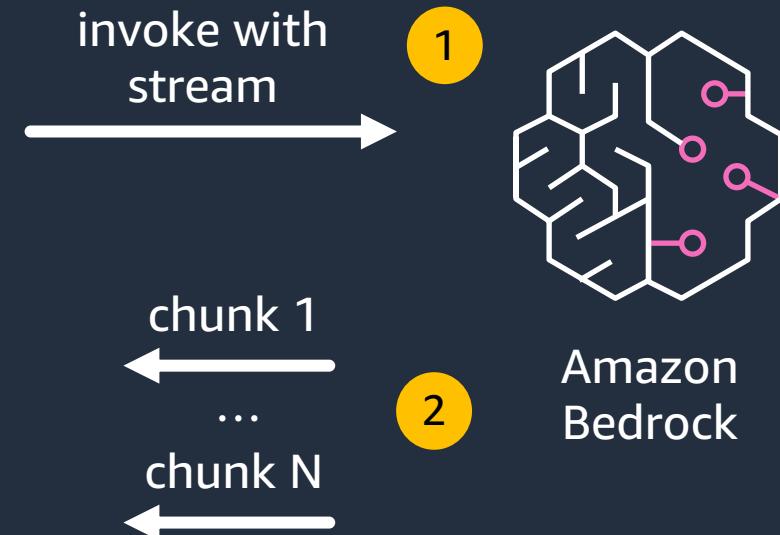


Access  
foundation  
models

- Amazon Titan models
- Third-party models
- Fine-tuned models

# Bedrock core API: Streaming responses

```
response = bedrock.invoke_model_with_response_stream(  
    modelId = model_id, body = body)  
stream = response.get('body')  
if stream:  
    for event in stream:  
        chunk = event.get('chunk')  
        if chunk:  
            print(json.loads(chunk.get('bytes')).decode())
```



- Users can start reading the response as soon as the first chunk is available
- Initially supported for Amazon Titan models; Claude and J2 models coming soon

# Access Bedrock via Boto3: API operations

- list foundation models()

Use the ListFoundationModels operation to retrieve information about the foundation models.

- invoke model()

Use this call to invoke the desired model. The API parameters and result depend on which model you are invoking.

\*You can access the Amazon Bedrock API using the AWS CLI and the AWS SDK for Python (Boto3)

# API operations: `invoke_model()`

## PARAMETERS

Parameters	Models				
	Titan Text Large	Claude	Command	Jurassic	Stable Diffusion
Temperature	✓	✓	✓	✓	
TopP	✓	✓	✓	✓	
StopSequences	✓	✓	✓	✓	
MaxTokens	✓	✓	✓	✓	
TopK		✓	✓		
CountPenalty				✓	
PresencePenalty				✓	
FrequencePenalty				✓	
Return Likelihoods			✓		
stream			✓		
Number of generations			✓		
Prompt strength (cfg_scale)					✓
Generation step					✓
Seed					✓

# invoke\_model() – Titan Text

```
prompt_data = """Command: Write me a blog about making strong business decisions as a leader."""

config = {"maxTokenCount":512,"StopSequences":[],"temperature":0.5,"topP":0.9}
body = json.dumps({"inputText": prompt_data}, "textGenerationConfig": config)
modelId = "amazon.titan-tg1-large"
accept = "*/*"
contentType = "application/json"
response = bedrock_runtime.invoke_model(
    body=body, modelId=modelId, accept=accept, contentType=contentType
)
response_body = json.loads(response.get("body").read())
print(response_body.get("results")[0].get("outputText"))
```

# invoke\_model() – Titan Embeddings

```
prompt_data = """This is just some input I want to get embeddings for"""

body = json.dumps({"inputText": prompt_data})
modelId = "amazon.titan-embed-text-v1"
accept = "*/*"
contentType = "application/json"
response = bedrock_runtime.invoke_model(
    body=body, modelId=modelId, accept=accept, contentType=contentType
)
response_body = json.loads(response.get("body").read())
embedding = response_body.get("embedding")
print(embedding)
```

# invoke\_model() - Anthropic Claude

```
prompt_data = """Command: Write me a blog about making strong business decisions as a leader."""

body = json.dumps({"prompt": prompt_data,
                   "max_tokens_to_sample": 500,
                   "temperature": 0.5,
                   "top_k": 250,
                   "top_p": 1,
                   "stop_sequences": ["\n\nHuman:"]
})
modelId = "anthropic.claude-instant-v1"
accept = "*/*"
contentType = "application/json"
response = bedrock_runtime.invoke_model(
    body=body, modelId=modelId, accept=accept, contentType=contentType
)
response_body = json.loads(response.get("body").read())
print(response_body.get("completion"))
```

# invoke\_model () - AI21 Labs Jurassic-2

```
prompt_data = """Command: Write me a blog about making strong business decisions as a leader."""

body = json.dumps({"prompt": prompt_data,
                    "maxTokens": 500,
                    "temperature": 0.5,
                    "topP": 0.5,
                    "stop_sequences": [],
                    "countPenalty": {"scale": 0},
                    "presencePenalty": {"scale": 0},
                    "frequencyPenalty": {"scale": 0}
                  })

modelId = "ai21.j2-mid-v1"
accept = "application/json"
contentType = "application/json"
response = bedrock_runtime.invoke_model(
    body=body, modelId=modelId, accept=accept, contentType=contentType
)
response_body = json.loads(response.get("body").read())
print(response_body.get("completions")[0].get("data").get("text"))
```

# invoke\_model() – Stability.ai Stable Diffusion

```
body = json.dumps({
    "text_prompts": [
        {"text": "a pizza", "weight": 1},
        {"text": "an unrealistic pizza shape", "weight": -1}
    ],
    "cfg_scale": 10,
    "seed": 20,
    "steps": 50,
    "style_preset": "cinematic",
    # For in-painting or image-to-image task, add following
    "init_image": "<base64 encoded string of image>",
    "start_schedule": 0.6 # 0-1 scale of relevancy
})
modelId = "stability.stable-diffusion-xl"
accept = "application/json"
contentType = "application/json"
response = bedrock_runtime.invoke_model(
    body=body, modelId=modelId, accept=accept, contentType=contentType
)
response_body = json.loads(response.get("body").read())
base_64_img_str = response_body["artifacts"][0]["base64"]
image = Image.open(io.BytesIO(base64.decodebytes(bytes(base_64_img_str, "utf-8"))))
print(image)
```

# Integrated with LangChain

```
pip install langchain
```

```
from langchain import Bedrock
from langchain.embeddings import BedrockEmbeddings

llm = Bedrock()
print(llm("what is generative
AI?"))
```

**Popular Python framework for developing applications powered by language models**

- New LLM and embeddings class for Amazon Bedrock
- Includes code for using the LLM class in a conversation chain
- Includes code for creating an embedding from text

# Summary of Amazon Bedrock inference options



## On-demand

Pay-as-you-go, no commitment

- › Pricing based on input and output token count for LLMs
- › Great for prototyping, POC's, small workloads with more relaxed requirements for throughput and latency
- › Requests per minute (RPM) and Tokens per minute (TPM) limits enforced



## Provisioned throughput

Provision sufficient throughput to meet your application's performance requirements

- › Guaranteed throughput at a fixed cost
- › Higher throughput available
- › Flexible commitment term of 1 month or 6 months
- › Pay hourly rate, discounted for extended commit
- › Great for production workloads, or inference on custom models

# Fine-tuning

## Adapt model to domain data

# Fine-tuning



Easily customize FMs  
using labelled data

## Create fine-tuned models using APIs

Fine-tuned models can be invoked the same way as base models, via the playground, or through the API on Provisioned throughput.

Currently, fine-tuning is supported for Titan Text models.  
Support for other models will be enabled over time.

Customers with access to Amazon Titan Text models will now have access to fine tuning. NEW

Tagging lets you easily manage, identify, organize, search for, and filter fine tuning jobs and models

# Fine-tuning

Launch a fine tuning job right from the console or using the SDK

The screenshot shows the Amazon Bedrock console with the 'Fine-tune' section selected. The left sidebar includes links for Overview, Models (Foundation models, Model providers, Fine-tune), Examples, Embeddings, Playgrounds (Text, Image), and Terms of service. The main area has a breadcrumb path: Amazon Bedrock > Fine-tune. It features two tables: 'Models' and 'Jobs'. The 'Models' table lists fine-tuned models with columns for Model name, Source model, Provider (Amazon), and Creation time. The 'Jobs' table lists training jobs with columns for Job name, Status, Source model (Titan Large), Provider (Amazon), Fine-tune model name, and Creation time.

**Models**

Model name	Source model	Provider	Creation time
○ [color swatch]	[color swatch]	Amazon	June 27, 2023, 20:06 (UTC-04:00)
○ [color swatch]	[color swatch]	Amazon	June 27, 2023, 15:43 (UTC-04:00)
○ [color swatch]	[color swatch]	Amazon	June 23, 2023, 16:25 (UTC-04:00)
○ [color swatch]	[color swatch]	Amazon	March 10, 2023, 14:04 (UTC-05:00)
○ [color swatch]	[color swatch]	Amazon	December 27, 2022, 16:26 (UTC-05:00)
○ [color swatch]	[color swatch]	Amazon	December 21, 2022, 15:54 (UTC-05:00)

**Jobs**

Job name	Status	Source model	Provider	Fine-tune model name	Creation time
○ [color swatch]	[color swatch]	Titan Large	Amazon	[color swatch]	June 27, 2023, 20:09 (UTC-04:00)
○ [color swatch]	[color swatch]	Titan Large	Amazon	[color swatch]	June 27, 2023, 20:06 (UTC-04:00)
○ [color swatch]	[color swatch]	Titan Large	Amazon	[color swatch]	June 27, 2023, 15:43 (UTC-04:00)

# Fine-tuning

Simple point & click form based approach – no Python experience required

The screenshot shows the 'Fine-tune a model' configuration page in the Amazon Bedrock console. It is divided into several sections:

- Model configuration**:
  - Source model**: A dropdown menu labeled "Select a source model".
  - Fine-tuned model name**: An input field labeled "Enter a name here".
  - Tags - optional**: A section for adding tags, with a note: "A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs." Below it says "No tags associated with the resource." and has a button "Add new tag".
- Input data**:
  - S3 location**: An input field with placeholder "s3://bucket/path-to-your-data/" and buttons "View" and "Browse S3".
  - Validation dataset S3 location**: An input field with placeholder "s3://bucket/path-to-your-data/" and buttons "View" and "Browse S3".
- Hyperparameters**:
  - Epochs**: A dropdown menu set to "10". Note: "The total number of iterations of all the training data in one cycle for training the model."
  - Batch size**: A dropdown menu set to "1". Note: "The number of samples processed before the model is updated."
  - Learning rate**: A dropdown menu set to "0.00005". Note: "The step size for incrementing parameters at each iteration."
  - Learning rate warmup steps**: A dropdown menu set to "0". Note: "Number of iterations over which learning rate is gradually increased to the initial rate specified."
- Output data**:
  - S3 location**: An input field with placeholder "s3://bucket/path-to-your-data/" and buttons "View" and "Browse S3".
- Service access**:
  - Note: "The fine-tuning job requires permission to access your S3 buckets."
  - Choose a method to authorize Bedrock**:
    - Use an existing service role
    - Create and use a new service role
  - Service role**: A dropdown menu labeled "Choose role".

At the bottom right are buttons: "Cancel" and "Fine-tune model" (highlighted in orange).

# Fine-tuning

Monitor job progress or stop the run right on the console

The screenshot shows the 'Job: job123' configuration page in the Amazon Bedrock console. The top navigation bar includes 'Amazon Bedrock > Fine-tune > Job: job123'. The main area is divided into sections: 'Model configuration', 'Input data', and 'Hyperparameters'. In the 'Model configuration' section, the 'Status' is shown as 'Training' with a yellow background. Other fields include 'Fine-tuned model name', 'Fine-tuning job name', 'Source model name (Titan Large)', 'Data access role', 'Fine-tuning job ARN', and 'Source model ARN'. The 'Input data' section contains a 'S3 location' field. The 'Hyperparameters' section lists 'Maximum number of epochs' (10), 'Learning rate' (5.0E-5), 'Batch size' (1), and 'Learning rate warmup steps' (0).

Model configuration	
Fine-tuned model name	[REDACTED]
Fine-tuning job name	[REDACTED]
Source model name	Titan Large
Data access role	[REDACTED]
Status	Training
Fine-tuning job ARN	[REDACTED]
Source model ARN	[REDACTED]

Input data	
S3 location	[REDACTED]

Hyperparameters			
Maximum number of epochs	10	Learning rate	5.0E-5
Batch size	1	Learning rate warmup steps	0

# Security



# Data privacy



## You are always in control of your data

- Customer data is not used to improve Amazon Titan models for other customers, and is not shared with other foundation model providers
- Customer data (prompts, responses, fine-tuned models) remain in the region where they are created

# Data security



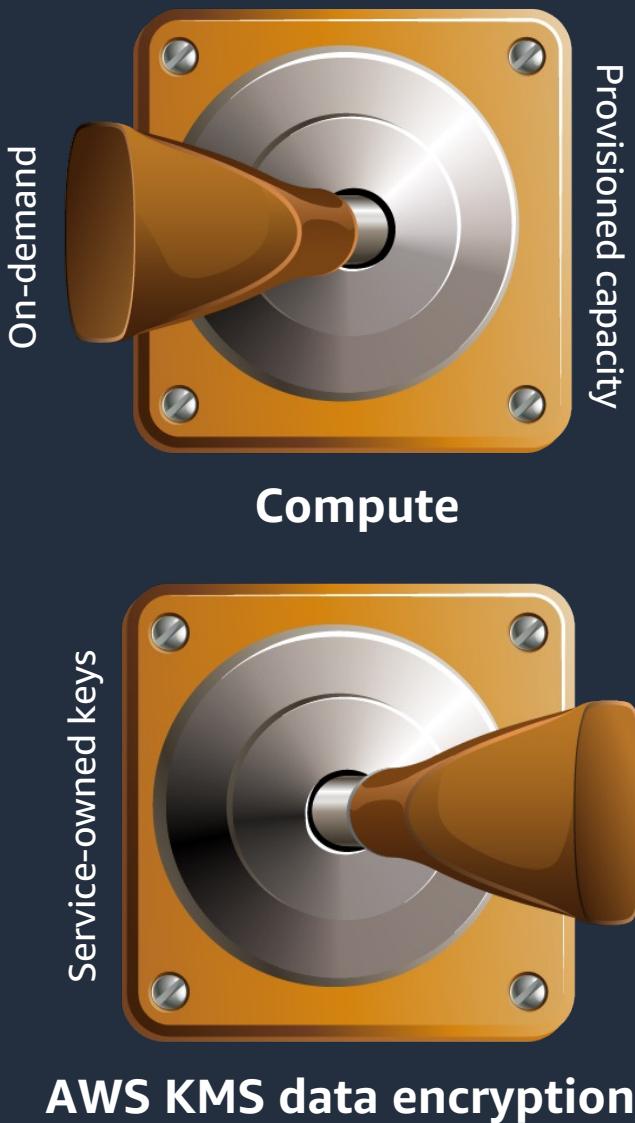
## You are always in control of your data

- Support for **AWS PrivateLink** so customers can establish private connectivity between virtual private clouds (VPCs) and the Bedrock service using VPC Endpoints
- Integration with AWS Identity and Access Management Service (IAM) to manage inference access, deny access for specific models, and enable Console access
- You can use CloudTrail to monitor API activity and troubleshoot issues as you build solutions
- Fine-tuned (customized) models are encrypted and stored using service managed keys; only you have access to your customized models through an endpoint
- Support for **Customer Managed Keys (CMK)** so customers can create and control keys to encrypt fine-tuned models
- Support for VPC configuration of fine-tuning jobs

# Configurable security controls



**Data privacy**



**AWS KMS data encryption**



**Model fine tuning**



**Access management**

# On-demand vs. provisioned compute capacity



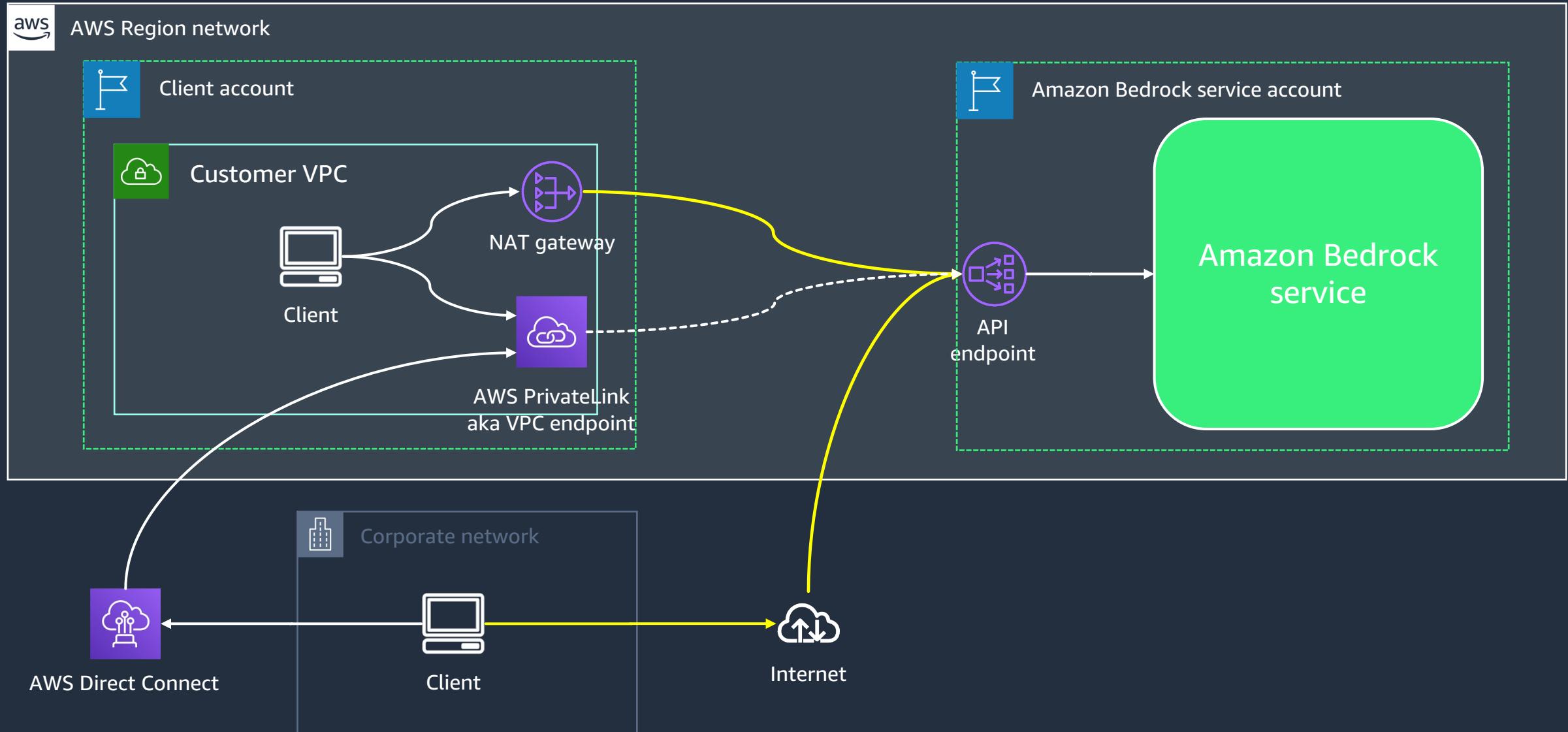
Provisioned  
capacity compute



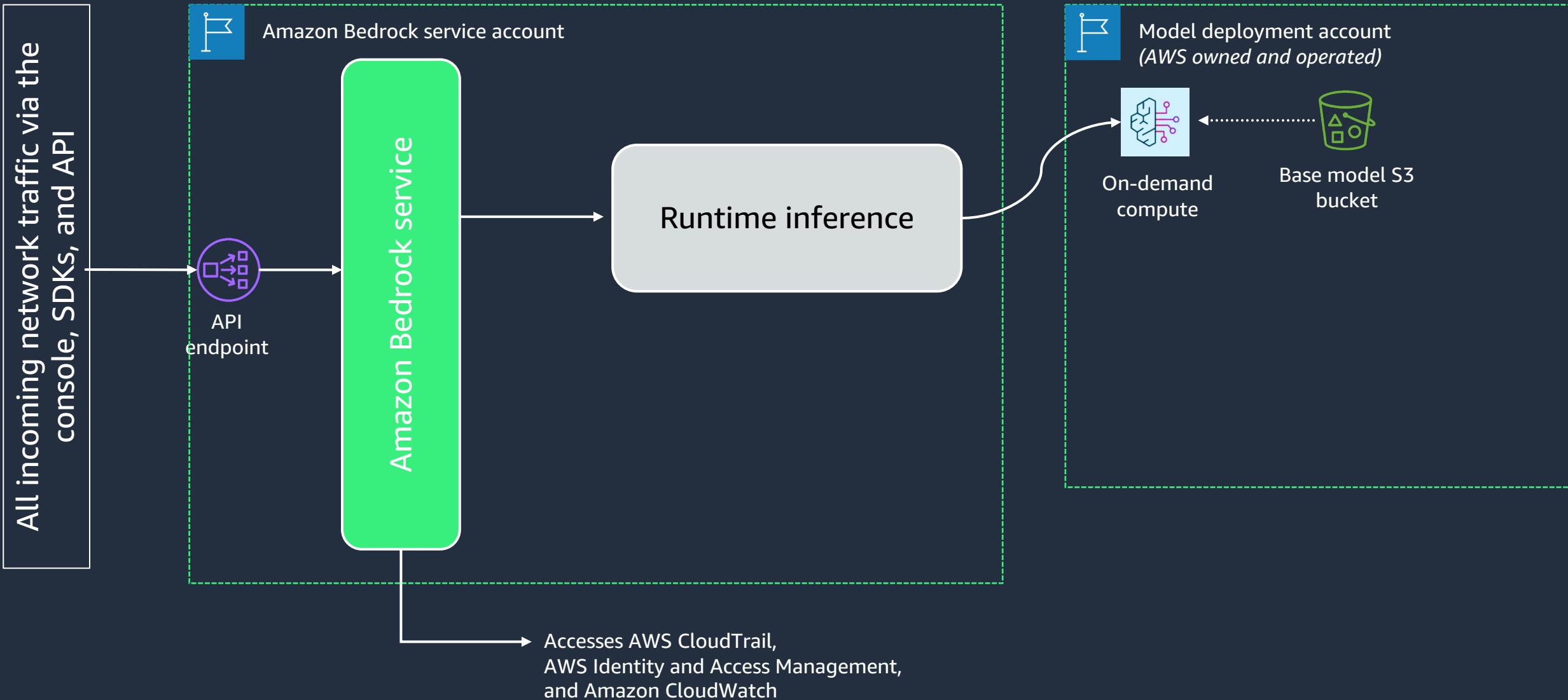
On-demand  
compute

- |   |   |
|---|---|
| <ul style="list-style-type: none"><li>1. Deployment available to a single customer</li><li>2. Holds a private copy of a baseline model that may have been fine-tuned by a customer</li></ul>  | <ul style="list-style-type: none"><li>1. Deployment available to all customers</li><li>2. Holds a baseline version of a supported model</li></ul> |
| <ul style="list-style-type: none"><li>3. No inference request's input or output text is used to train any model(s) in the deployment</li><li>4. Model deployments are inside an AWS account owned and operated by the Bedrock service team</li><li>5. Model vendors have no access to any customer data</li></ul> |   |

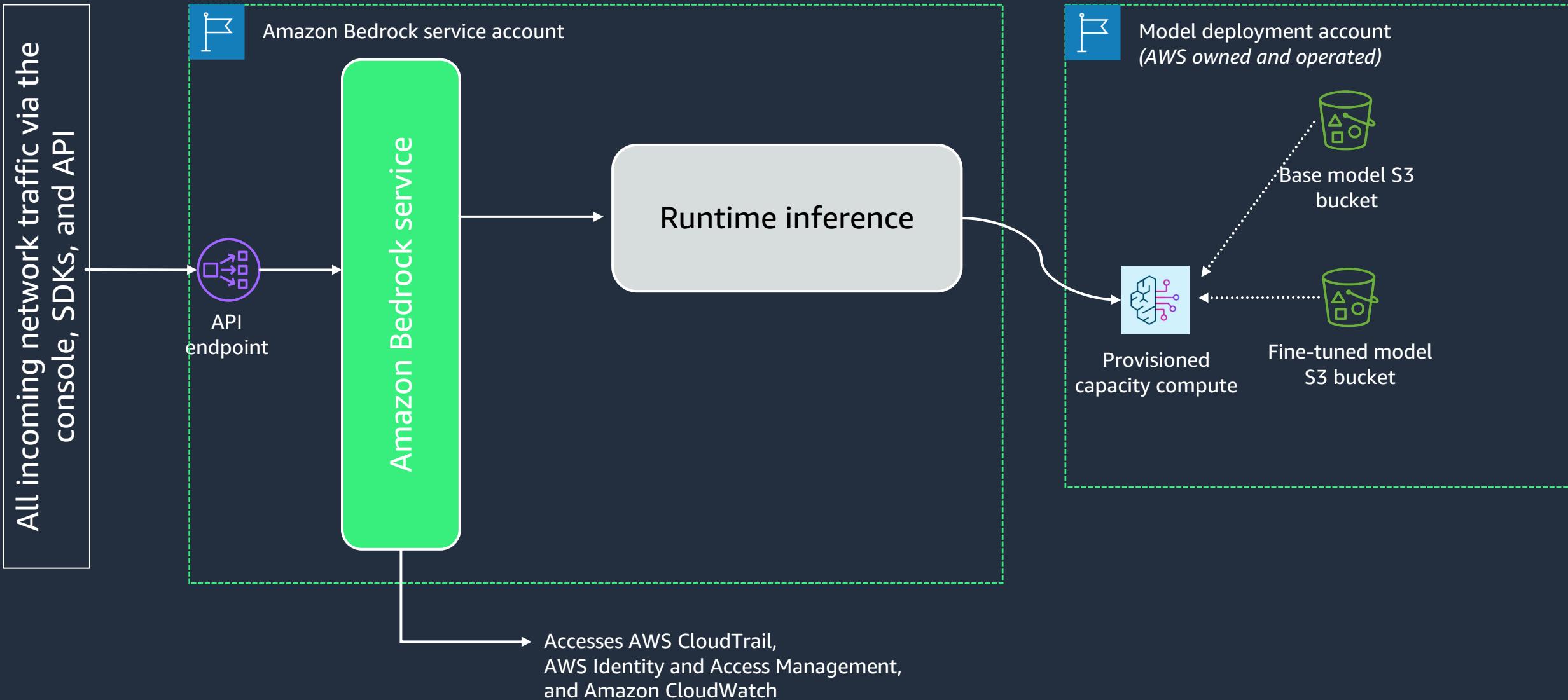
# Client connectivity



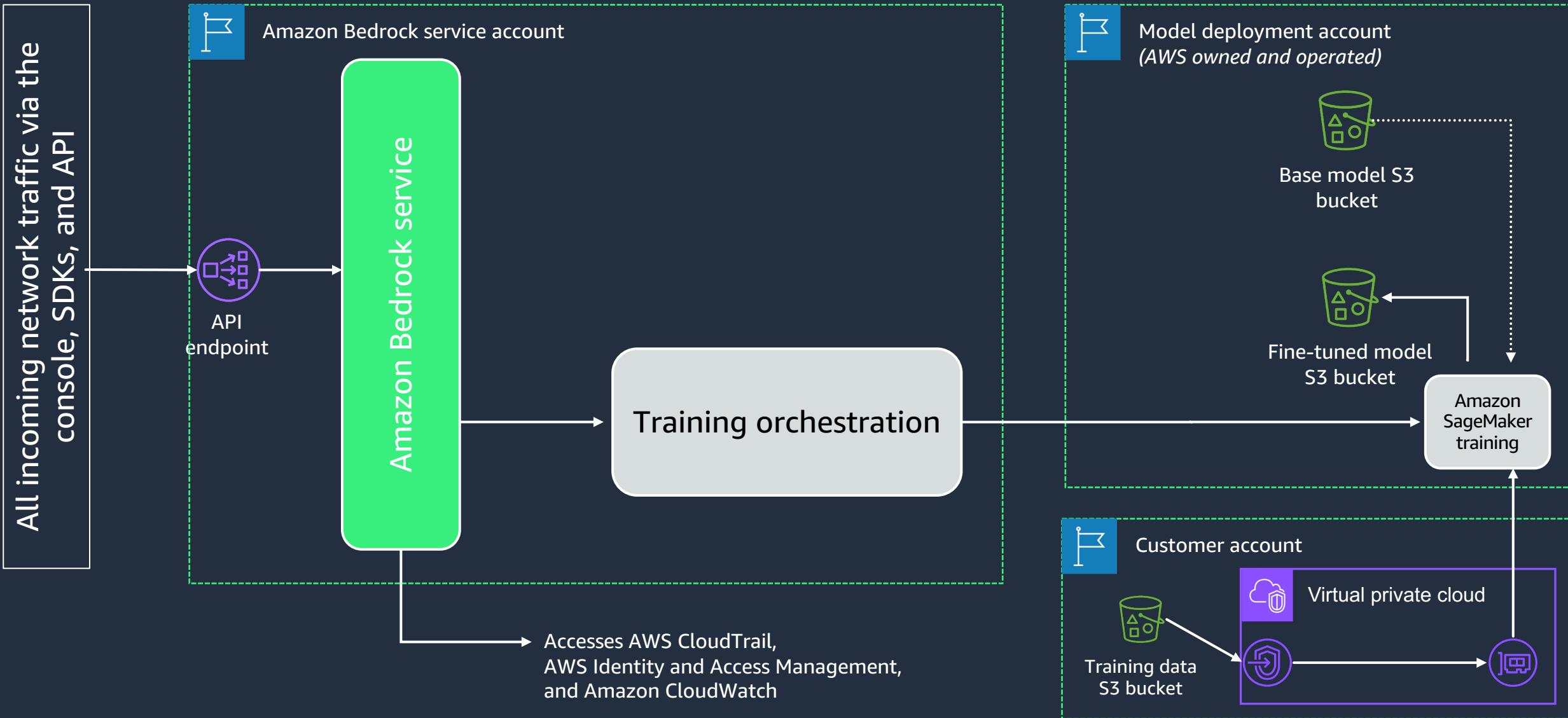
# On-demand compute architecture overview



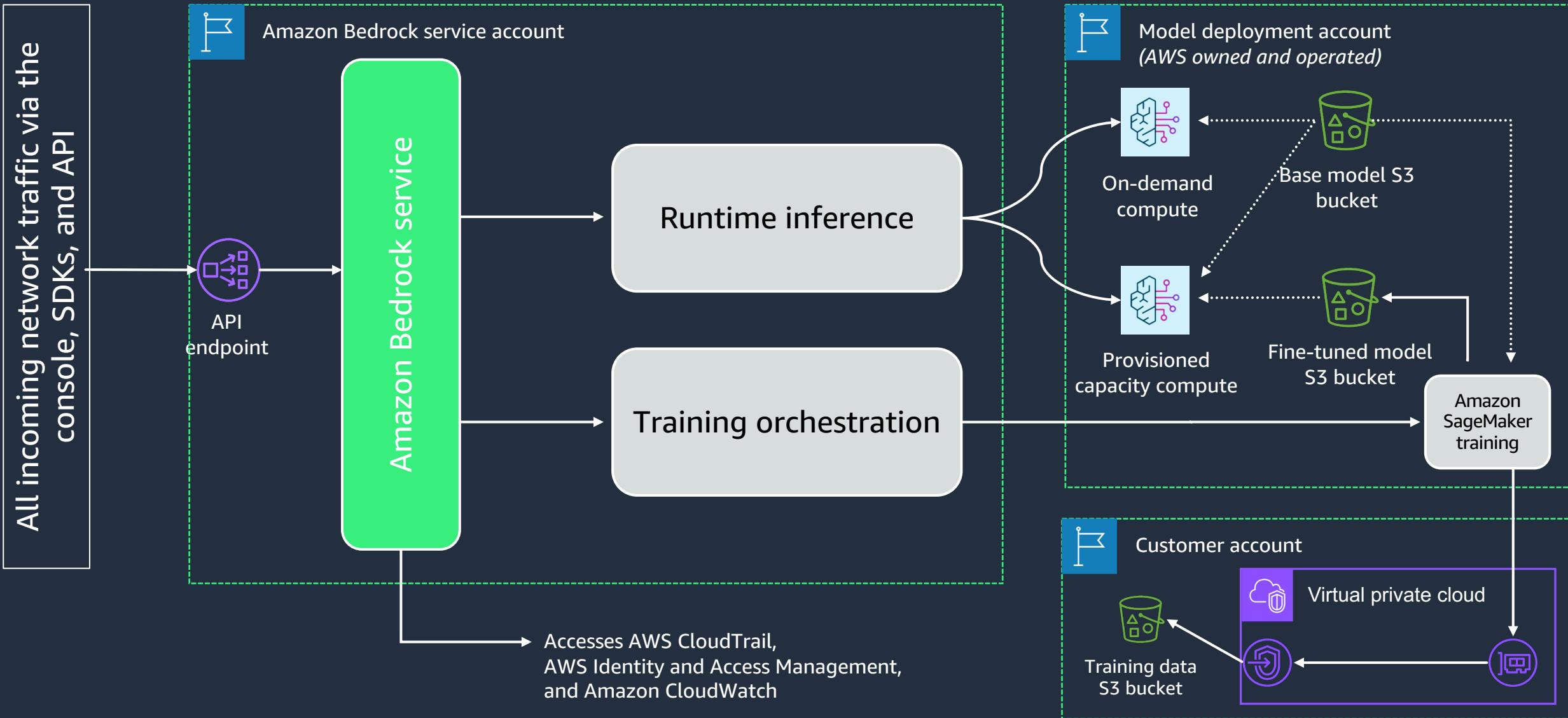
# Provisioned capacity architecture overview



# Model fine-tuning architecture overview



# Complete architecture overview



# AWS Identity and Access Management



IAM



- Identity-based policies
- Actions
- Resources
- Tags (ABAC)

# AWS IAM Fine Grained Access Controls

## AMAZON BEDROCK IDENTITY BASED POLICIES

IAM Policy



```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "BedrockConsole",  
      "Effect": "Allow",  
      "Action": [  
        "bedrock>ListFoundationModels",  
        "bedrock>InvokeModel"  
      ],  
      "Resource": "*"  
    }  
  ]  
}
```

Action(s)

Resource(s)

Condition Key(s)

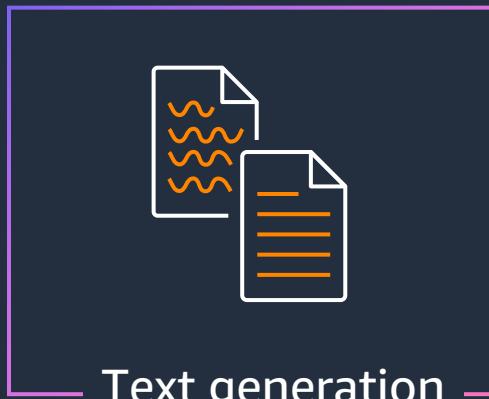
# IAM/SCP – Example deny policy

```
{  
  "version": "2012-10-17",  
  "Statement":  
  {  
    "Sid": "DenyInferenceForModelX",  
    "Effect": "Deny",  
    "Action": "bedrock:InvokeModel",  
    "Resource": "arn:aws:bedrock::::foundation-model/<name-of-model>"  
  }  
}
```

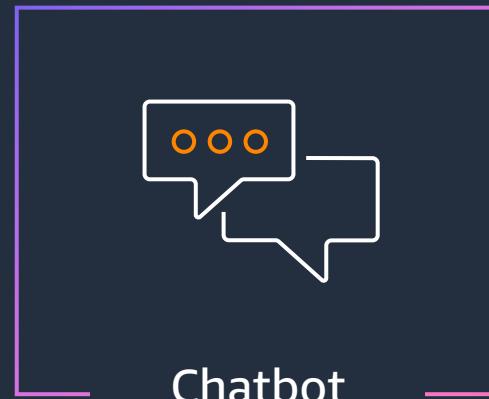


# Architecture patterns

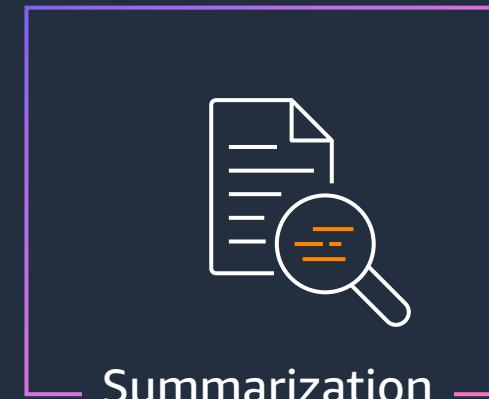
# Architecture patterns in:



Text generation



Chatbot



Summarization

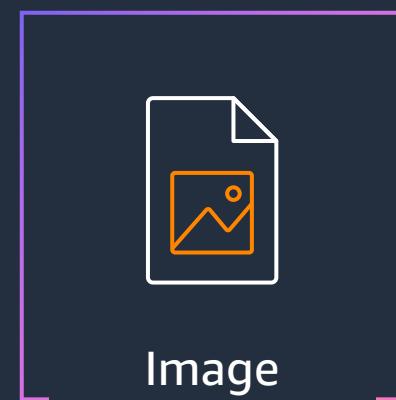


Image  
generation



Question  
answering

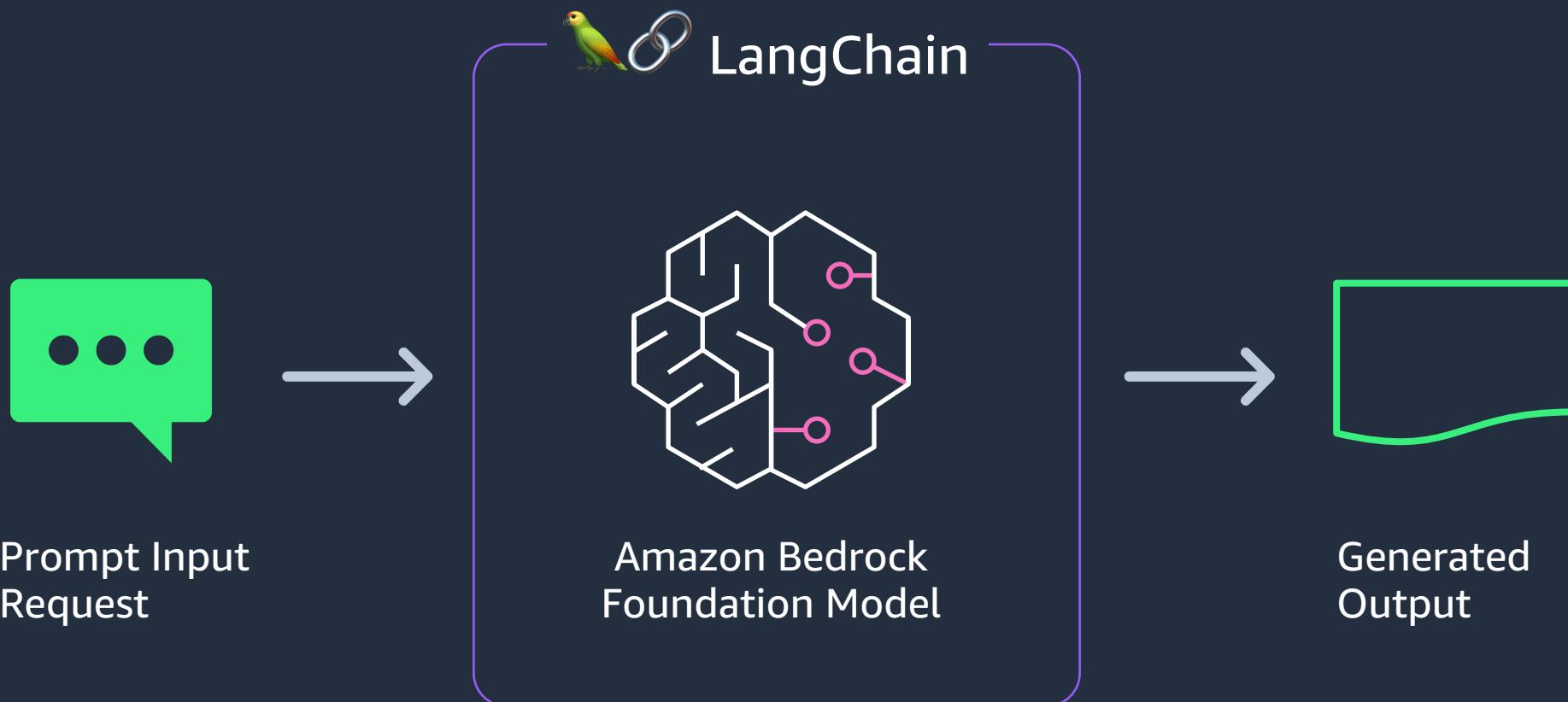
# Text Generation

WITH SIMPLE PROMPT



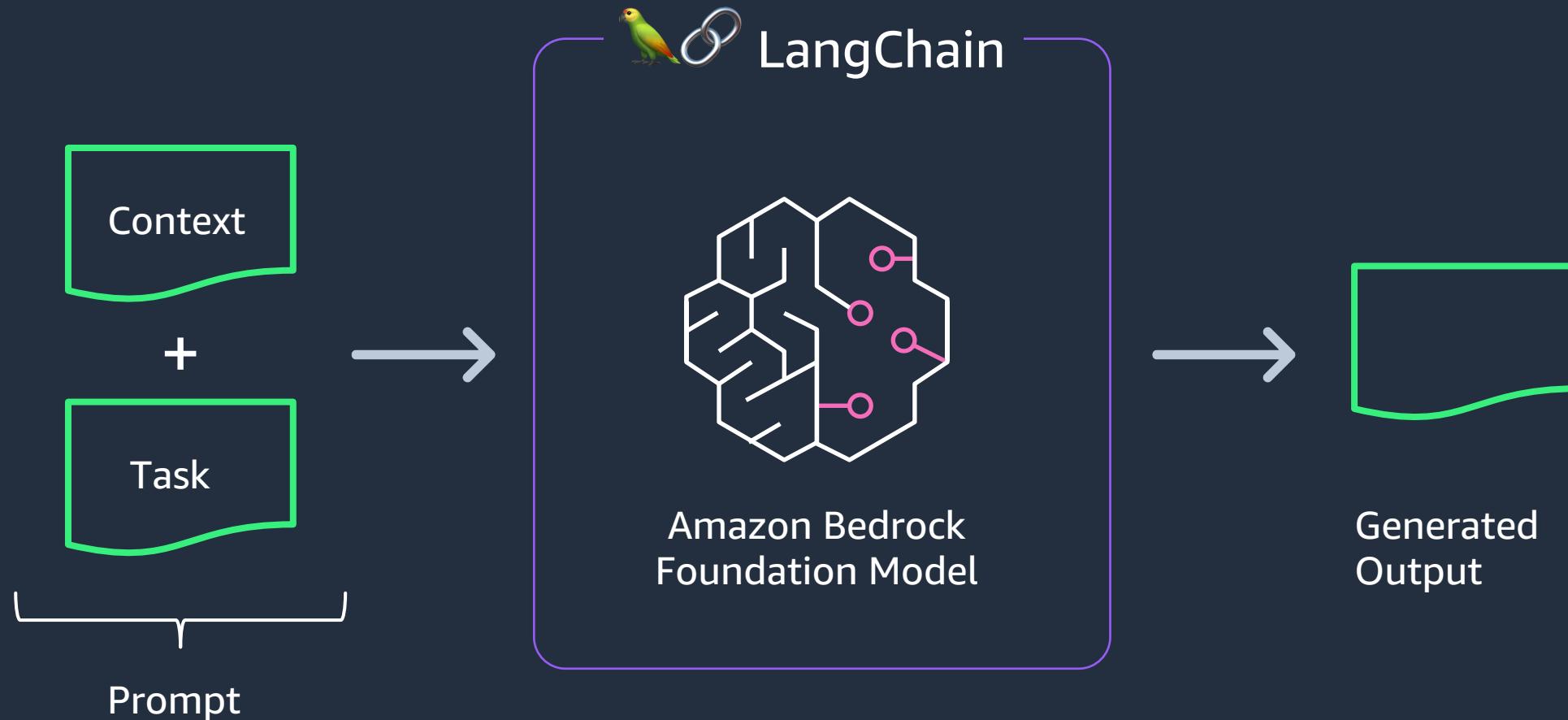
# Text Generation

WITH LANGCHAIN



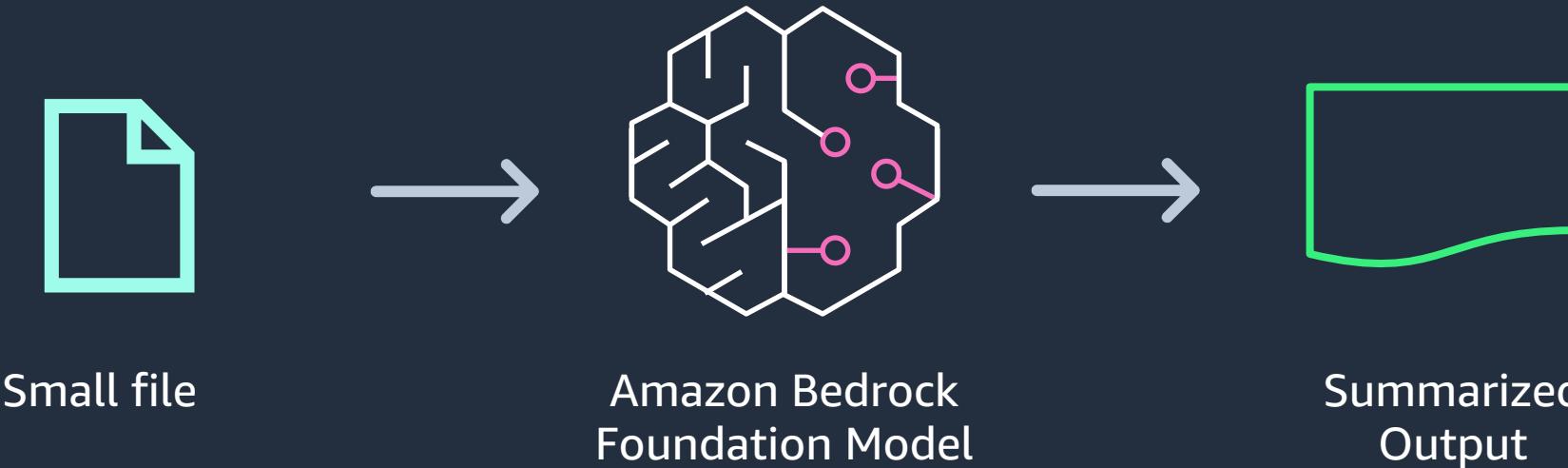
# Text Generation

WITH CONTEXT AND LANGCHAIN



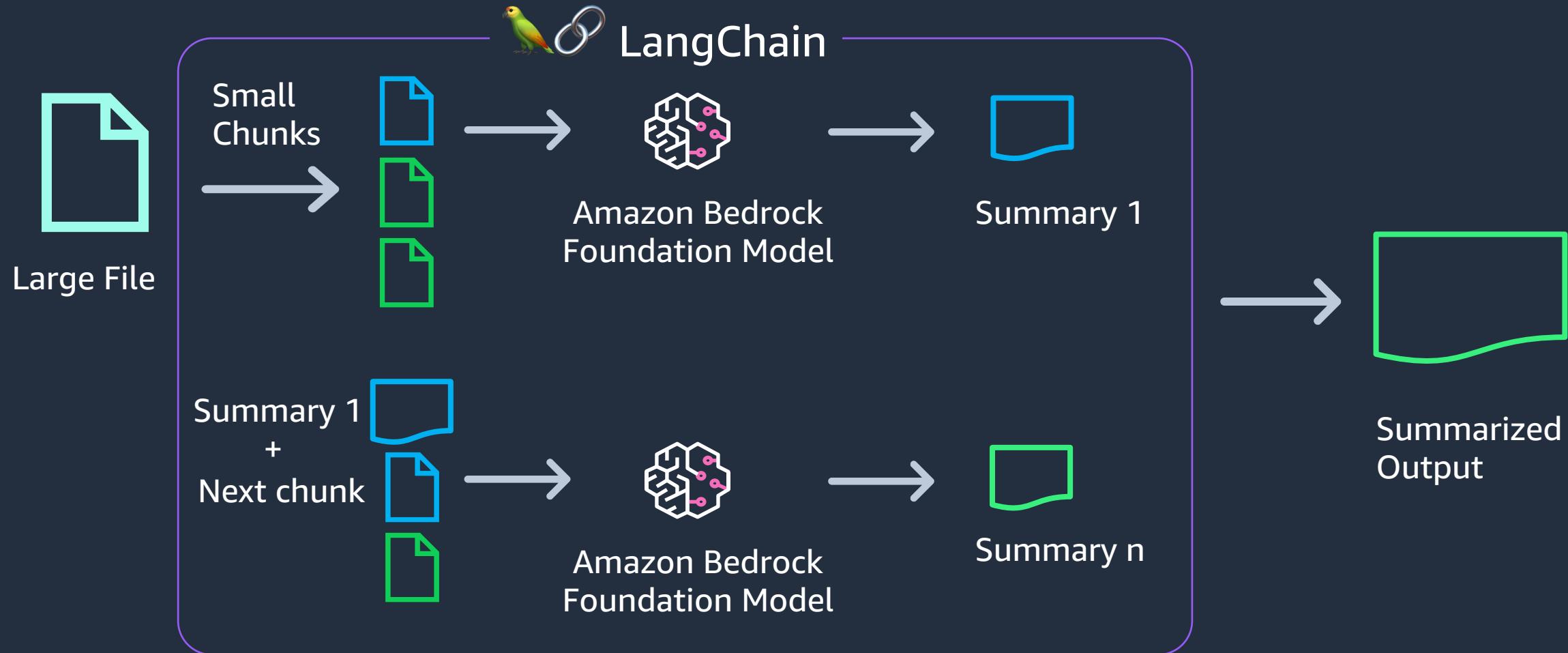
# Text Summarization

WITH SMALL FILES



# Text Summarization

WITH LARGE FILES AND LANGCHAIN



# Question Answering

WITH SIMPLE PROMPT



User  
Question

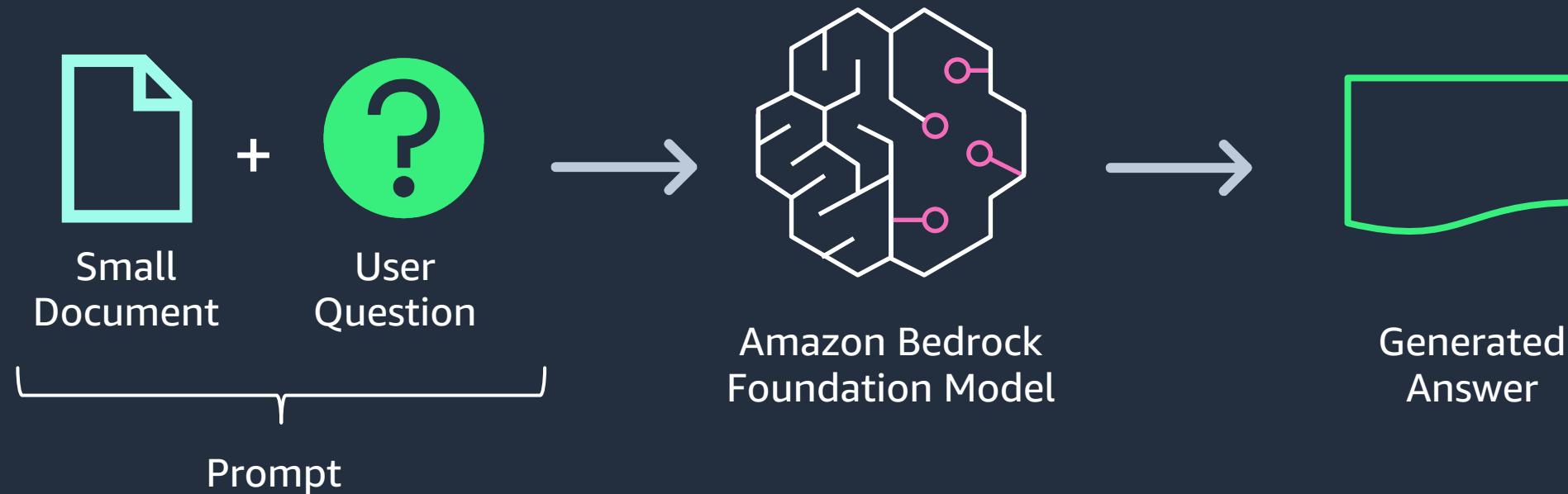


Amazon Bedrock  
Foundation Model

Generated  
Answer

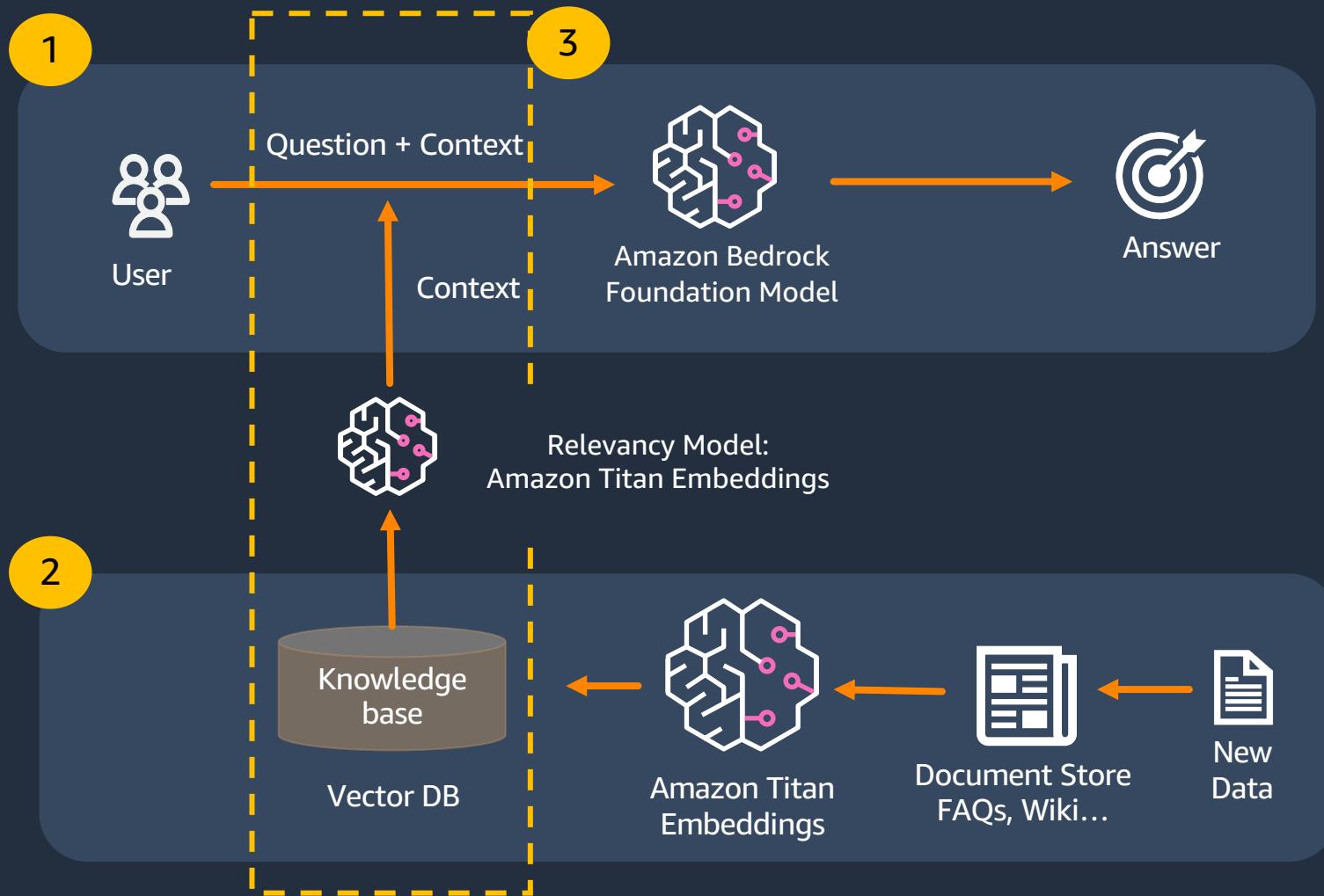
# Question Answering

WITH CONTEXT



# Question Answering

WITH RETRIEVAL-AUGMENTED GENERATION



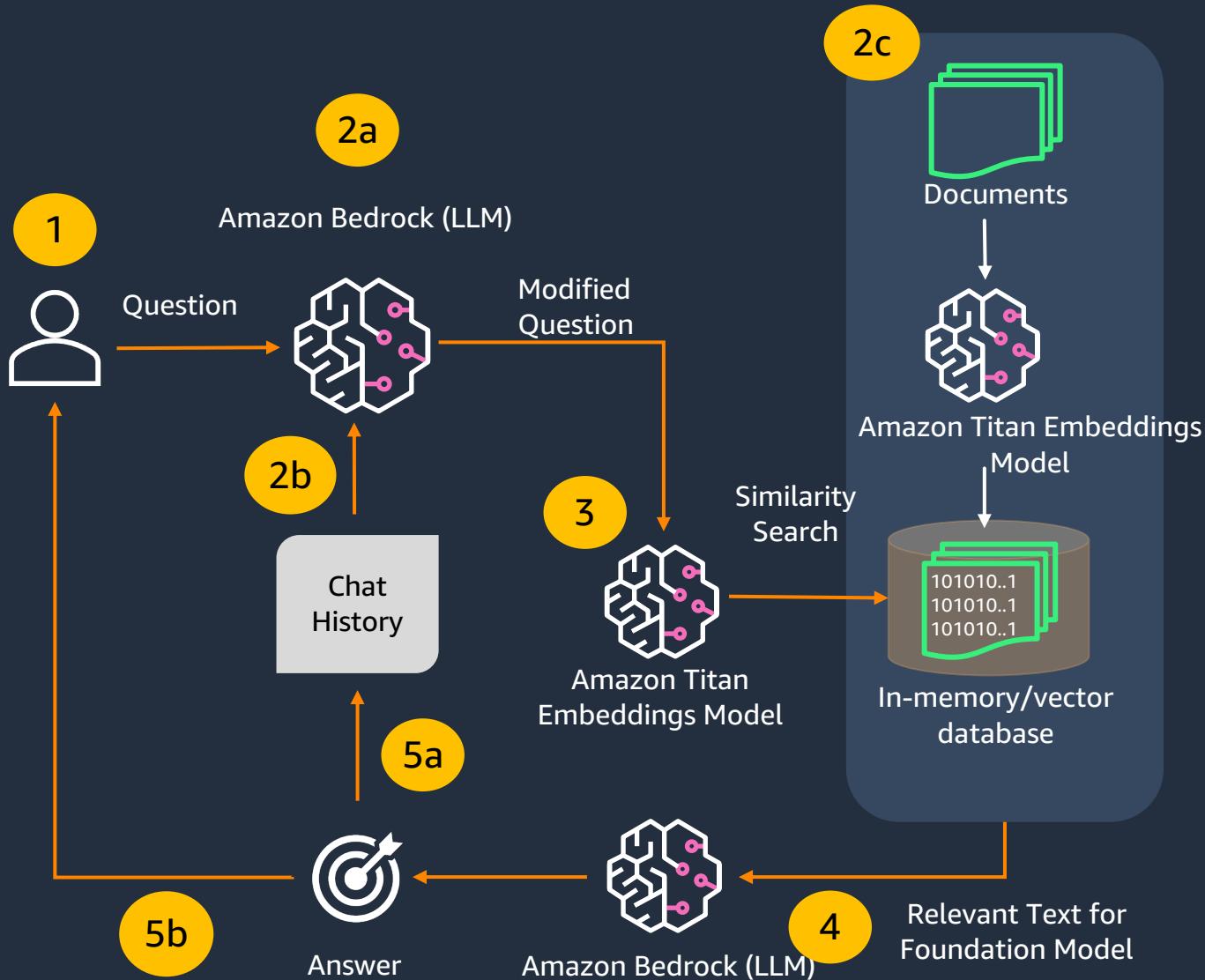
# Chatbot

## BASIC

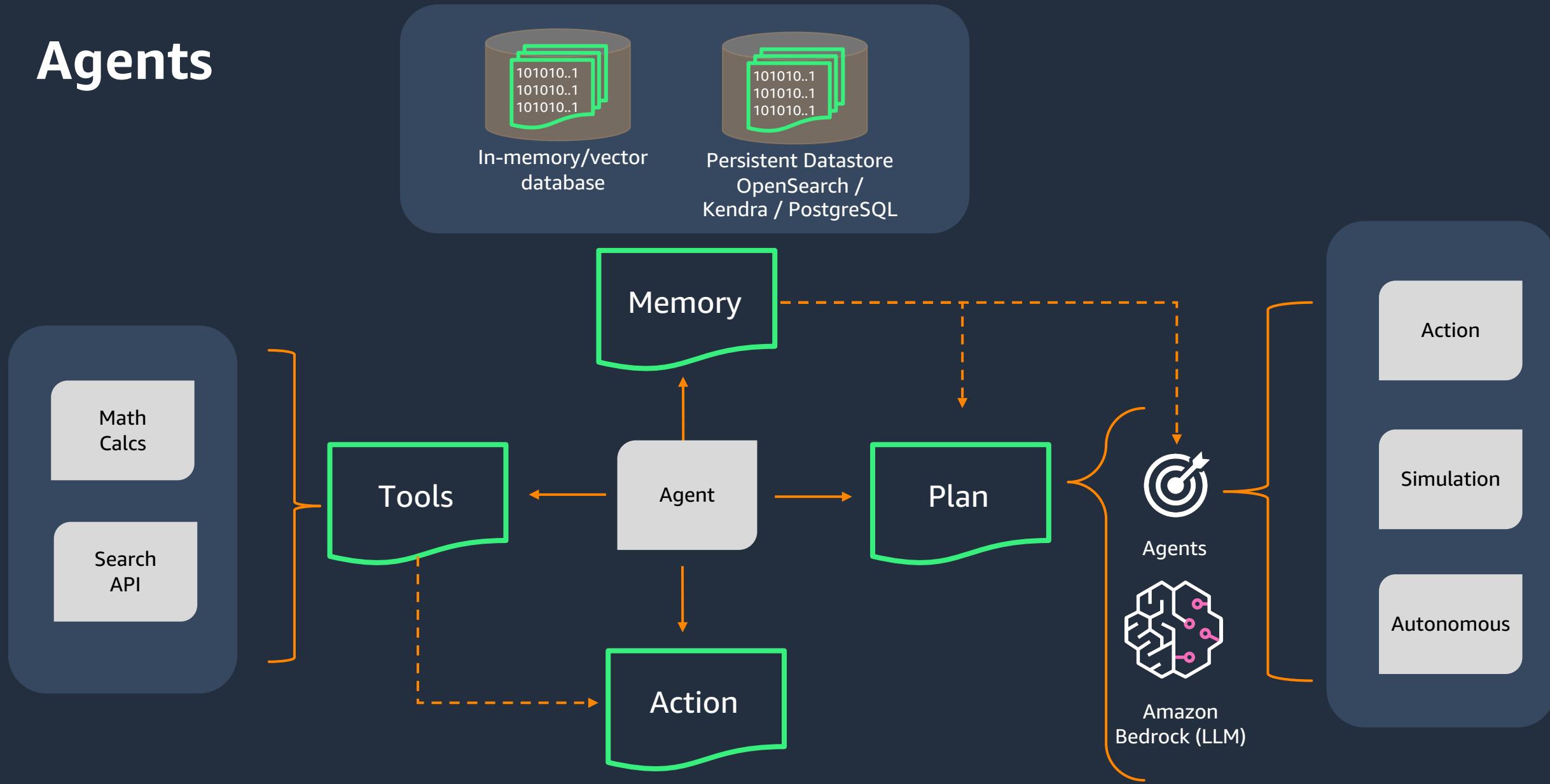


# Chatbot

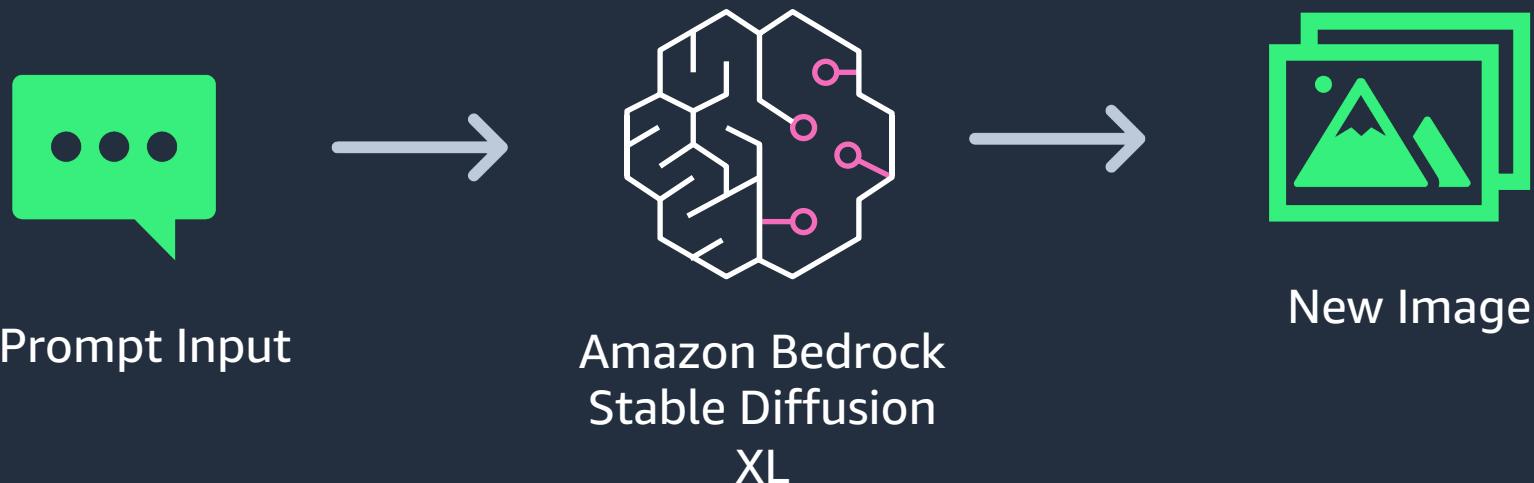
WITH CONTEXT



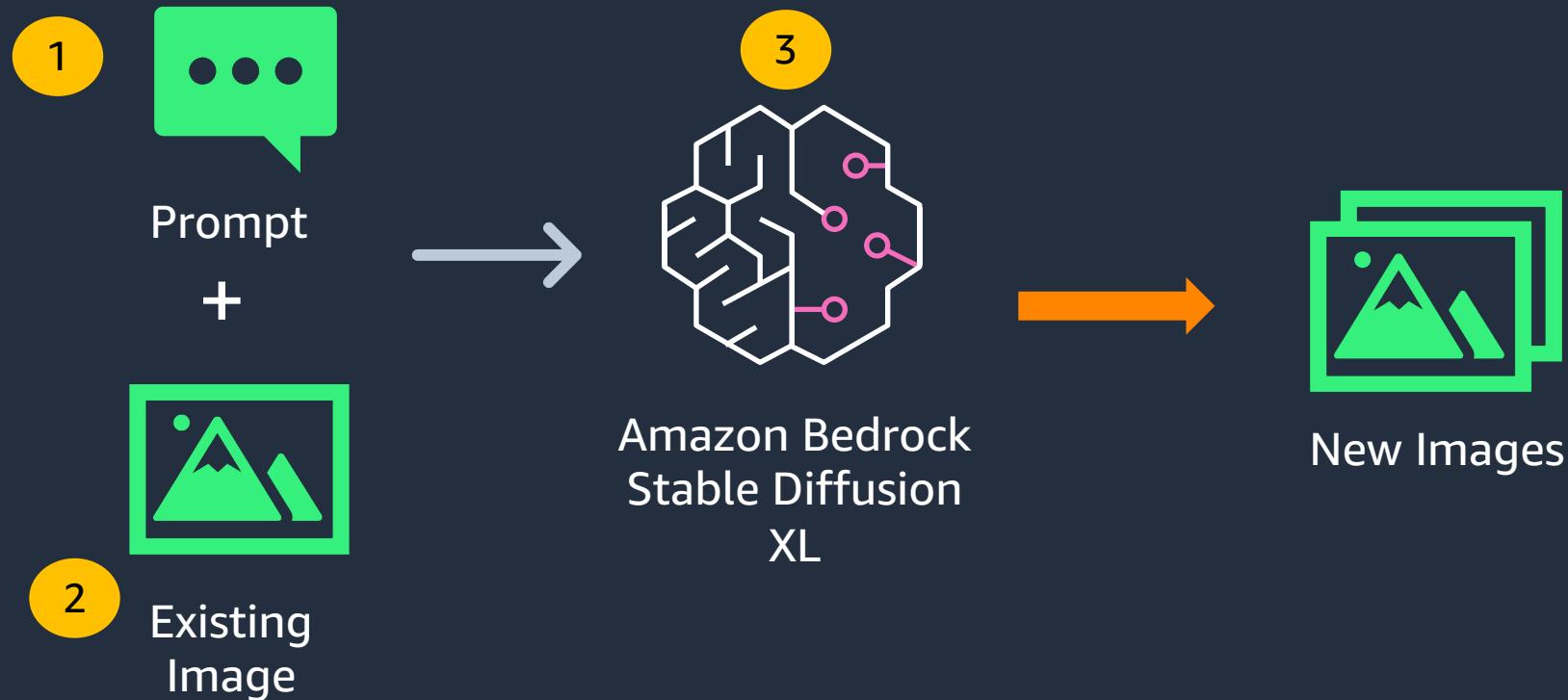
# Agents



# Text to Image



# Image to Image (In-painting)





# Thank you!



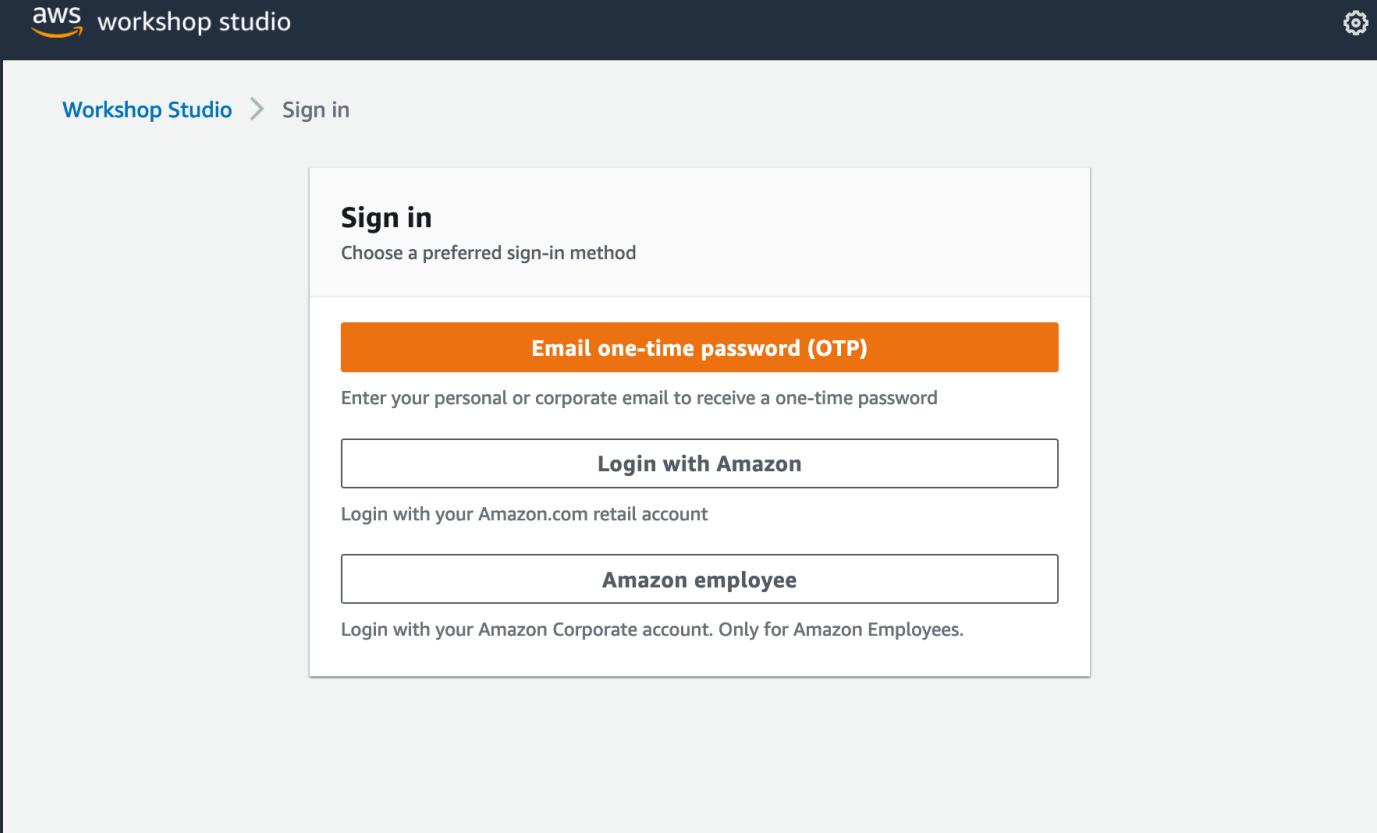
# Amazon Bedrock

## Hands-on Labs Instructions

IBM Consulting, Nordcloud, Agyla  
AWS Generative AI Bootcamp Day 2

# Step 1: Sign in via your preferred method

<https://catalog.workshops.aws/join>



# Step 2: Enter event access code



18c4-069cc5-93

The screenshot shows a web browser window for the AWS Workshop Studio. The title bar says "aws workshop studio". The main content area has a header "Workshop Studio > Join event". On the left, there's a sidebar with "Step 1: Enter event access code" and "Step 2: Review and join". The main content area is titled "Enter event access code" and contains a form field labeled "Event access code" with the placeholder text "Event access code A 12 digit code that was given to you for this event". At the bottom right of the form are "Cancel" and "Next" buttons. At the very bottom of the page, there's a footer with copyright information: "© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy policy Terms of use".

# Step 3: Review terms and join event

aws workshop studio

Workshop Studio > Join event

Step 1  
Enter event access code

Step 2  
Review and join

### Review and join

#### Event details

Name	Start time	Duration	Level
Generative AI on Amazon SageMaker Workshop	7/16/2023 08:00 PM	72 hours	300

Description  
Generative AI on Amazon SageMaker Workshop

#### Terms and Conditions

Read and accept before joining the event

Read and accept before joining the event:

1. By using AWS Workshop Studio for the relevant event, You agree to [the AWS Event Terms and Conditions](#) and the [AWS Acceptable Use Policy](#).
2. If You are under 18 years old, You may participate in the relevant event, using AWS Workshop Studio: (a) if you are at least the minimum age below based on the country or region in which you reside, and (b) with the involvement of a parent, guardian or educator.

Country or region	Minimum age
All countries or regions not listed below (including United States, Brazil, UK, and India)	13
China, Republic of Korea (South Korea)	14

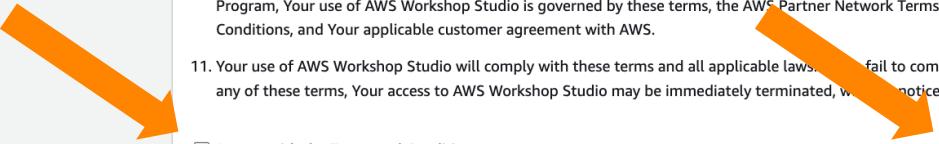
9. AWS, its affiliates and any entities or persons acting on AWS's behalf, has no obligation to enable the transmission of Your materials through AWS Workshop Studio, and may, in its discretion, edit, block, refuse to post, or remove Your materials at any time, without notice to You.

10. If You are an AWS Partner, using AWS Workshop Studio as part of Your participation in the AWS Partner Network Program, Your use of AWS Workshop Studio is governed by these terms, the AWS Partner Network Terms and Conditions, and Your applicable customer agreement with AWS.

11. Your use of AWS Workshop Studio will comply with these terms and all applicable laws. If You fail to comply with any of these terms, Your access to AWS Workshop Studio may be immediately terminated, without notice to You.

I agree with the Terms and Conditions

Cancel Previous **Join event**



# Step 4: Access AWS account

Access the [AWS Console](#), or generate AWS CLI credentials as needed

The screenshot shows the AWS workshop studio interface for the "Generative AI on Amazon SageMaker Workshop". The top navigation bar includes the AWS logo, "workshop studio", a user icon, and the name "Arnaud Lauer". A blue banner at the top indicates "Event ends in 2 days 14 hours 25 minutes". The left sidebar contains a table of contents for the workshop, including sections like "Prerequisites", "Lab 1. Jumpstart Models for text and image generation", "Lab 2. Prompt Engineering", "Lab 3. Customize and train a large language model on SageMaker", and "AWS account access". The "AWS account access" section is highlighted with a red box, containing links to "Open AWS console (us-east-1)" and "Get AWS CLI credentials". The main content area displays the "Generative AI on Amazon SageMaker Workshop" event details, including the start time (7/16/2023 10:03 AM), duration (72 hours), accessible regions (us-east-1), and a description of the workshop. Below this is a "Workshop" card with details such as title, complexity level (300), AWS services (Amazon SageMaker Studio, Amazon SageMaker), and topics (High Performance Computing (HPC), Machine Learning (ML/AI)). A "Get started" button is present in this card. The bottom section shows "Event Outputs (0)" with a search bar and a table header for "Key", "Value", "Stack name", "Description", and "Type".



# Bedrock Hands-on Labs – Workshop

- Bedrock Workshop GitHub [link](#)
- Bedrock Workshop Studio [link](#) (for reference)

# Steps to get started

1. Navigate to **us-east-1** in the **AWS account**
2. Go to **SageMaker** in the **AWS Management Console**
3. Select Domain (**StudioDomain**) from the panel on the left hand side
4. Double click on the domain profile to open it
5. You will find one user profile **studio-user**
6. Click on Launch button and select Studio
7. Wait for SageMaker Studio to open (takes a few mins)
8. Go to **File=> New => Terminal** to open the terminal
9. Run following commands in the terminal  
**git clone <https://github.com/aws-samples/amazon-bedrock-workshop.git>**
10. Open the notebook inside **00\_intro**
11. Select **Data Science 3.0** as the Kernel image

[Continue to next slide]

# Steps to get started (cont'd)

1. In every notebook, update the Cell containing AWS\_DEFAULT\_REGION and BEDROCK\_ASSUME\_ROLE with:
  1. **os.environ["AWS\_DEFAULT\_REGION"] = "us-east-1"**
  2. Leave the line with os.environ['AWS\_PROFILE'] commented
  3. **os.environ["BEDROCK\_ASSUME\_ROLE"] = "arn:aws:iam::122594109988:role/ibm-workshop-emea-virtual"**

```
# ----- ⚠️ Un-comment and edit the below lines as needed for your AWS setup ⚠️ -----
os.environ["AWS_DEFAULT_REGION"] = "us-east-1" # E.g. "us-east-1"
# os.environ["AWS_PROFILE"] = "<YOUR_PROFILE>"
os.environ["BEDROCK_ASSUME_ROLE"] = "arn:aws:iam::122594109988:role/ibm-workshop-emea-virtual" # E.g.
```

# Steps to get started (cont'd)

- Select Domain (**StudioDomain**) from the panel on the left hand side
- Double click on the domain profile to open it

Awsum! Click to copy link to clipboard

aws Services Search [Option+S] N. Virginia WSParticipantRole/Participant @ 6524-7434-1270

**Amazon SageMaker** X

Getting started

Studio

Studio Lab ↗

Canvas

RStudio

TensorBoard

Profiler

▼ Admin configurations

Domains **Domains**

Role manager

Images

Lifecycle configurations

SageMaker dashboard

Search

▼ JumpStart

Foundation models NEW

Amazon SageMaker > Domains

## Domains Info

A domain includes an associated Amazon Elastic File System (EFS) volume; a list of authorized users; and a variety of security, application, policy, and Amazon Virtual Private Cloud (VPC) configurations. Each user in a domain receives a personal and private home directory within the EFS for notebooks, Git repositories, and data files.

► Domain structure diagram

Name	ID	Status	Created on	Modified on
StudioDomain	d-or2oqiblkaqz	InService	Oct 12, 2023 10:28 UTC	Oct 12, 2023 10:33 UTC

Find domain name

C View Edit Create domain

< 1 >

93

# Steps to get started (cont'd)

- Double click on the domain profile to open it
- Find your **studio-user** as the name of the profile

Awsum! Click to copy link to clipboard

aWS Services Search [Option+S] N. Virginia WSParticipantRole/Participant @ 6524-7434-1270

Amazon SageMaker X

Getting started Studio Studio Lab Canvas RStudio TensorBoard Profiler

▼ Admin configurations Domains Role manager Images Lifecycle configurations

SageMaker dashboard Search

▼ JumpStart Foundation models NEW

Amazon SageMaker > Domains > Domain: StudioDomain

## StudioDomain

### Domain details

Configure and manage the domain.

User profiles Space management Environment Domain settings

**User profiles** Info

A user profile represents a single user within a domain. It is the main way to reference a user for the purposes of sharing, reporting, and other us

Search users

Name	Modified on	Created on
studio-user	Oct 12, 2023 10:34 UTC	Oct 12, 2023 10:34 UTC

Personal apps

- Studio
- Canvas
- TensorBoard
- Profiler

Collaborative

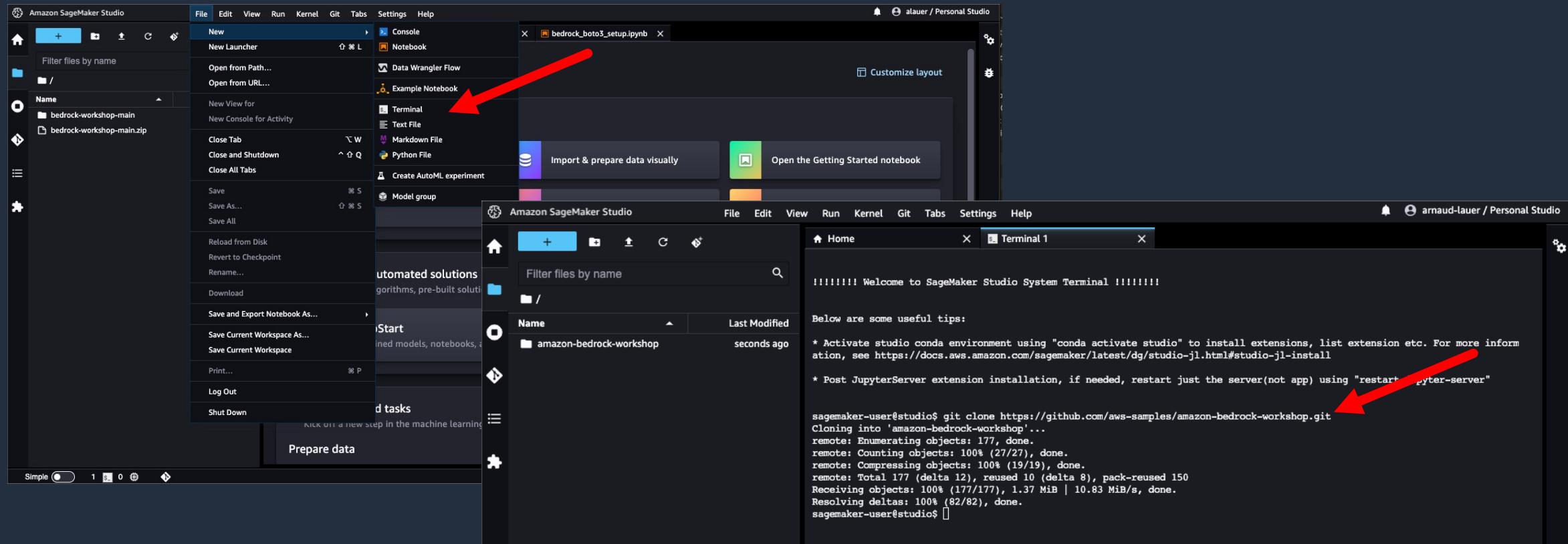
- Spaces

Launch ▲

94

# Steps to get started (cont'd)

- Launch Terminal Window in SageMaker Studio
- Run commands: **git clone <https://github.com/aws-samples/amazon-bedrock-workshop.git>**



# Steps to get started (cont'd)

The screenshot shows the Amazon SageMaker Studio interface. On the left, there's a file browser with a folder named 'bedrock-workshop-main / 00\_Intro /' containing a file 'bedrock\_boto3\_setup.ipynb'. In the center, a notebook titled 'bedrock\_boto3\_setup.ipynb' is open, showing the title 'Bedrock boto3 Setup' and some introductory text. At the top right, it says 'Cluster' and 'Data Science 2.0 | Python 3 | 2 vCPU + 4 GiB'. A red arrow points from the 'bedrock\_boto3\_setup.ipynb' file in the sidebar to the 'Starting notebook kernel...' message in the top right. Another red arrow points from the 'Data Science 2.0 | Python 3' cluster settings at the top right to the orange callout box.

**1. Open the notebook inside 00\_intro**  
**2. Select Data Science 3.0 as the Kernel image**

In this demo notebook, we demonstrate how to set up the Bedrock Python SDK for your Jupyter notebooks.

### Prerequisites

Before executing any of the notebook in this workshop, execute the following cells to add Bedrock extensions to the `boto3` Python SDK

```
[2]: %pip install ../dependencies/botocore-1.30.1-py3-none-any.whl ../dependencies/boto3-1.27.1-py3-none-any.whl ../dependencies/awscli-1.28.1-py3-none-any.whl
```

```
Processing /root/bedrock-workshop-main/dependencies/botocore-1.30.1-py3-none-any.whl
Processing /root/bedrock-workshop-main/dependencies/boto3-1.27.1-py3-none-any.whl
Processing /root/bedrock-workshop-main/dependencies/awscli-1.28.1-py3-none-any.whl
Collecting jmespath<2.0.0,>=0.7.1 (from botocore==1.30.1)
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting python-dateutil<3.0.0,>=2.1 (from botocore==1.30.1)
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
    247.7/247.7 kB 2.7 MB/s eta 0:00:000:01
Collecting urllib3<1.27,>=1.25.4 (from botocore==1.30.1)
  Downloading urllib3-1.26.16-py2.py3-none-any.whl (143 kB)
    143.1/143.1 kB 2.2 MB/s eta 0:00:00a 0:00:01
```

Simple    1 \$ 0 ⚡ Python 3 (Data Science 2.0) | Idle    Kernel: Starting... | Instance MEM    Mode: Command    Ln 1, Col 1 bedrock\_boto3\_setup.ipynb 0 🔔

## Steps to get started (cont'd)

In every notebook, update the Cell containing AWS\_DEFAULT\_REGION and BEDROCK\_ASSUME\_ROLE with:

**os.environ["AWS\_DEFAULT\_REGION"] = "us-east-1"**

Leave the line with os.environ['AWS\_PROFILE'] commented

**os.environ["BEDROCK\_ASSUME\_ROLE"] =**

**"arn:aws:iam::122594109988:role/ibm-workshop-emea-virtual"**

```
# ---- ⚠ Un-comment and edit the below lines as needed for your AWS setup ⚠ ----  
os.environ["AWS_DEFAULT_REGION"] = "us-east-1" # E.g. "us-east-1"  
# os.environ["AWS_PROFILE"] = "<YOUR_PROFILE>"  
os.environ["BEDROCK_ASSUME_ROLE"] = "arn:aws:iam::122594109988:role/ibm-workshop-emea-virtual" # E.g.
```



# Thank you!