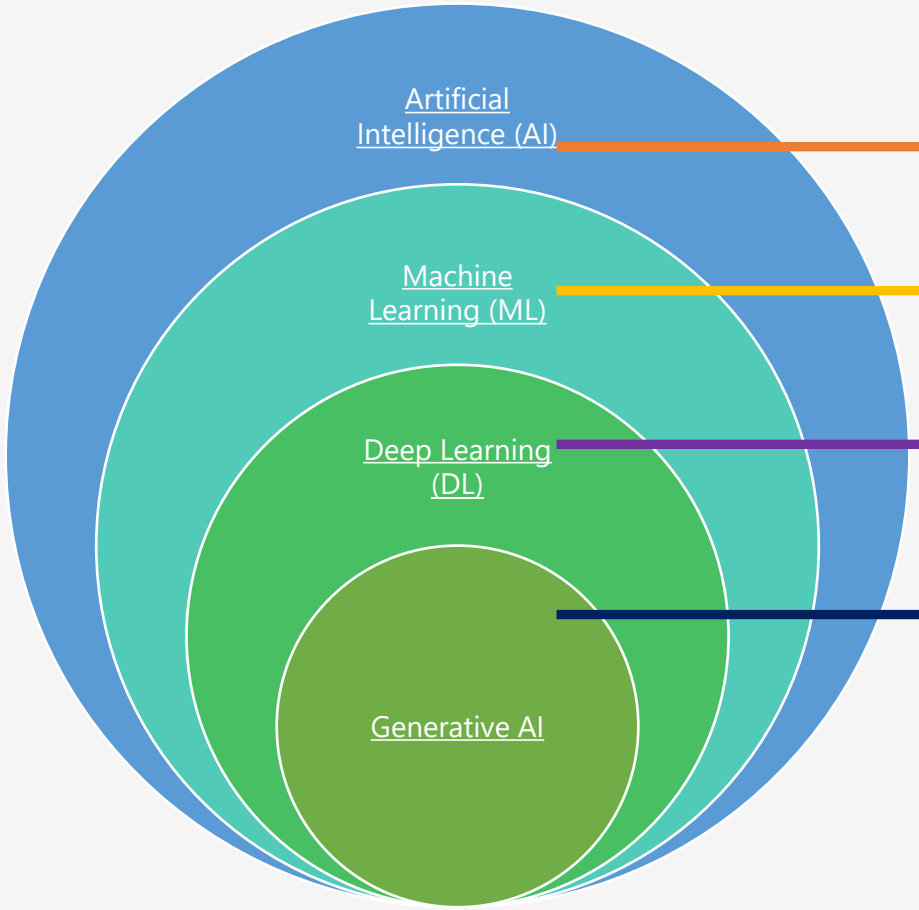
Udemy Course on Amazon Bedrock & Generative AI Beginner to Advanced

Image Generated using the use case built in this course

Section on :
Evolution of Generative AI

What is Generative Artificial Intelligence – Context ?



Science of making machines to **mimic human behavior**

Branch of AI which focuses on the use of **data and algorithms** to imitate human behavior

Subset of ML which use **neural networks** to perform tasks with limited human assistance

Subset of deep-learning models that can **generate high-quality text, images, videos**

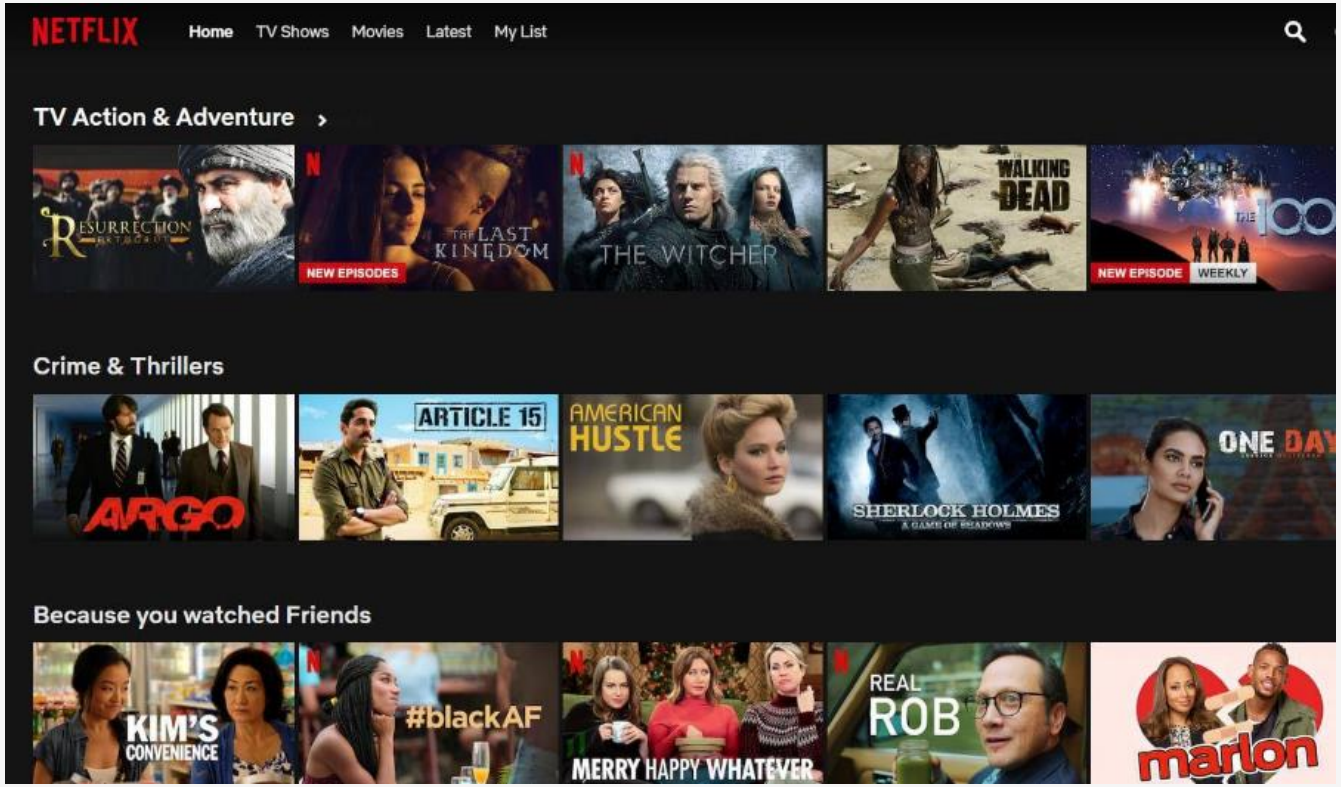
Artificial Intelligence - Overview

Artificial Intelligence is the science and engineering of **making** intelligent machines, especially **intelligent computer programs** that **simulate human intelligence and decision making.**

Artificial Intelligence - Examples

Self-driving cars use artificial intelligence to make real-time decisions based on sensor data.

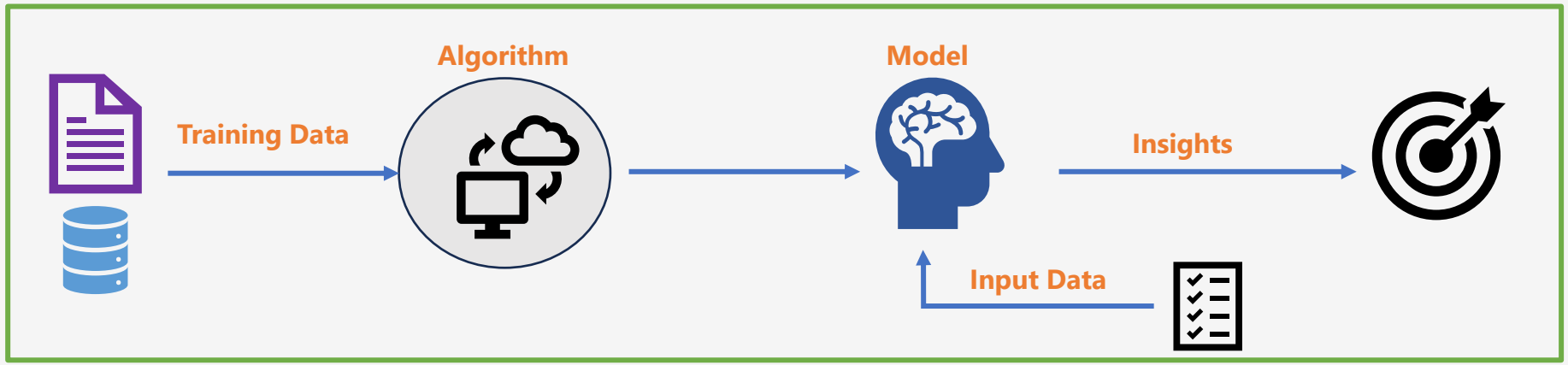
Artificial Intelligence - Examples



Recommendation engines such as ones from **Netflix and Amazon** use AI to provide custom insights

Machine Learning - Overview

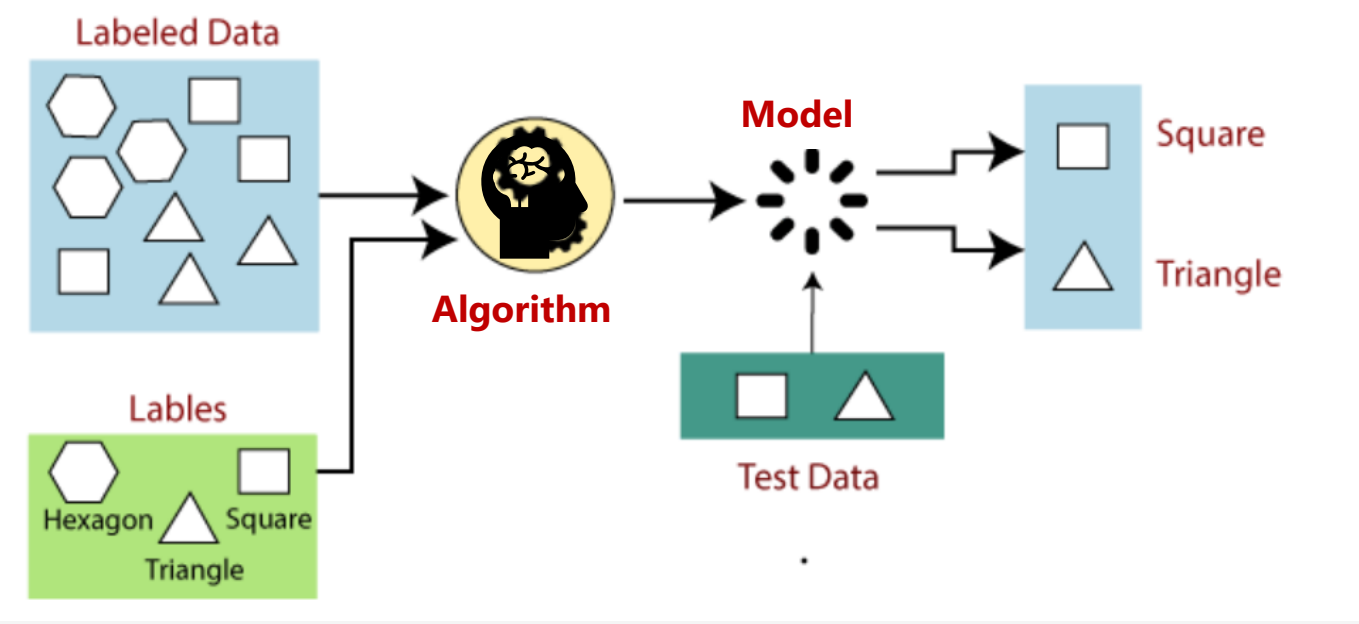
- Machine Learning is a **subset of AI**
- In ML the **algorithm is trained using the historical data** to make **predictions on new data.**



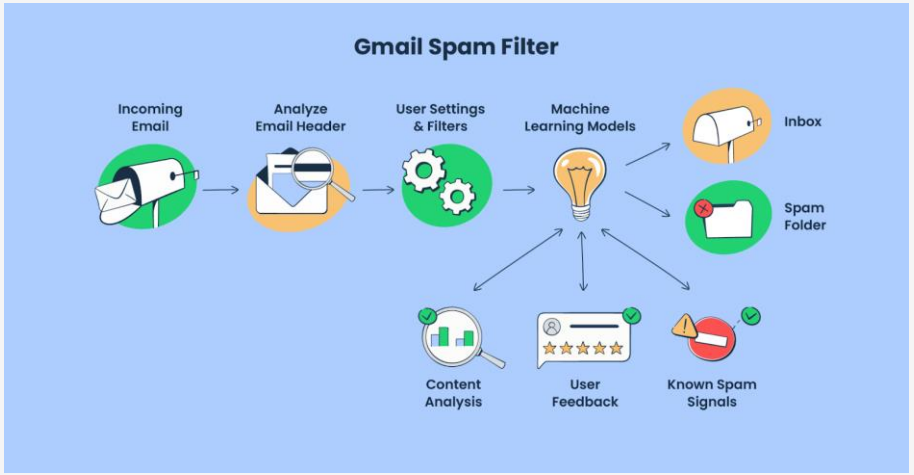
Machine Learning – Broad Categorization

Machine Learning – Supervised Learning

- In supervised learning, **algorithms are trained using <u>labelled dataset</u>.**
- Once the **training is completed**, the **model is tested using test data** and then it predicts the output.



Use Case (Classification) – Gmail Spam Filter



Advertisement	Sales
$90	$1000
$120	$1300
$150	$1800
$100	$1200
$130	$1380
$200	??

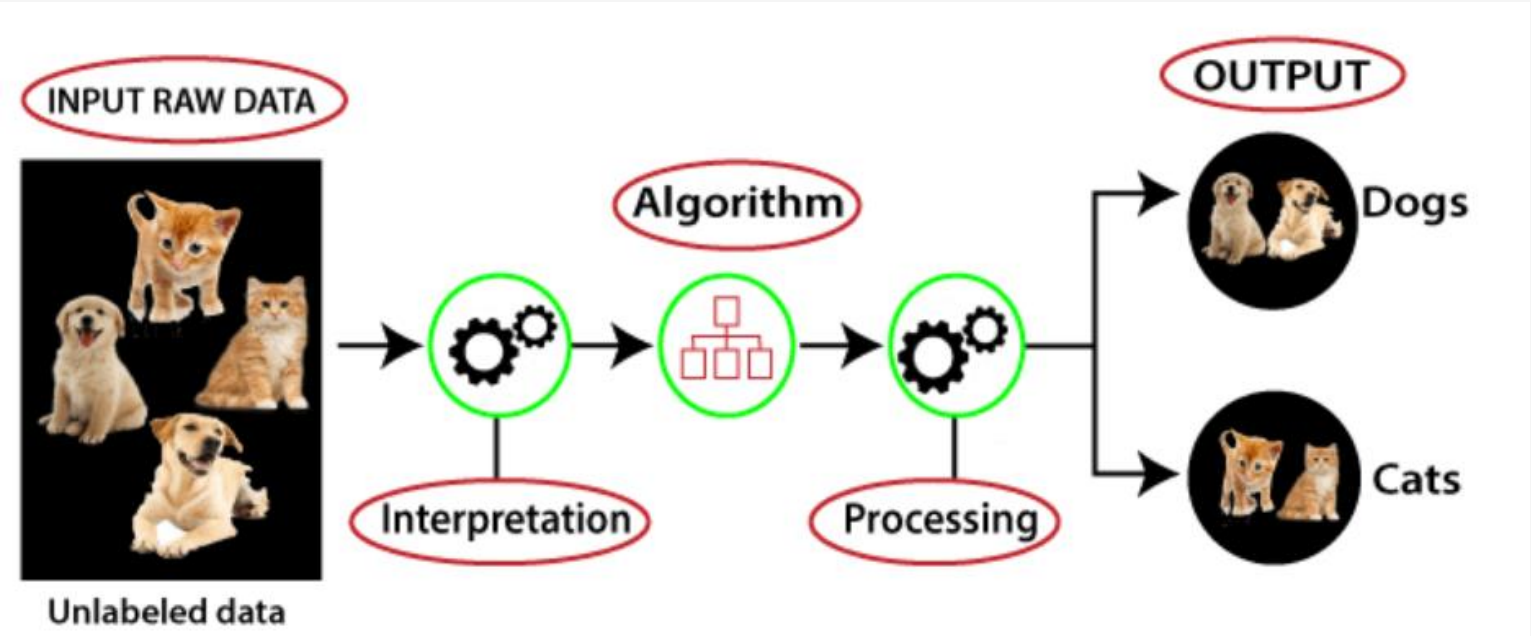
Use Case (Regression) – Forecasting

Common Algorithms - Classification and Regression

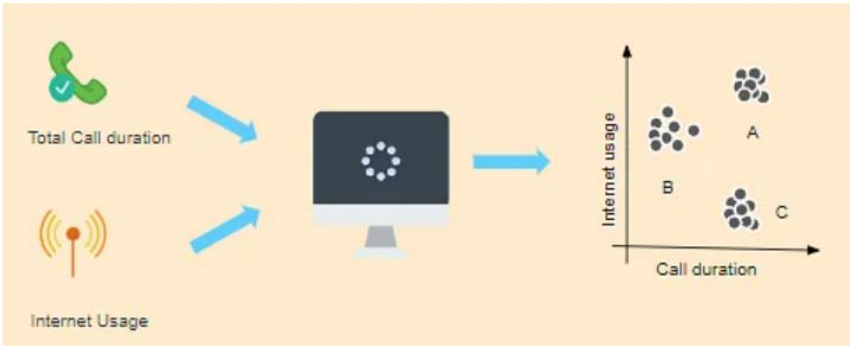
Image Source : internet

Machine Learning – Unsupervised Learning

- Unsupervised learning is a type of machine learning in which **algorithm are trained using <u>unlabeled dataset</u>**
- Based on the dataset, model finds **hidden patterns and insights** based on the **similarities of the data**



Use Case (Clustering) – Reduce Churn Rate



Use Case (Association) – Market Basket Analysis

Common Algorithms - Clustering and Association

Image Source : internet

Machine Learning – Reinforced Learning

- Reinforcement learning (RL) is a ML technique that focuses on **training an algorithm following the trial and error approach**.
- No pre-defined dataset
- The **algorithm (agent) evaluates a current situation** (state), **takes an action**, and **receives feedback** (reward and feedback) from the environment after each act.

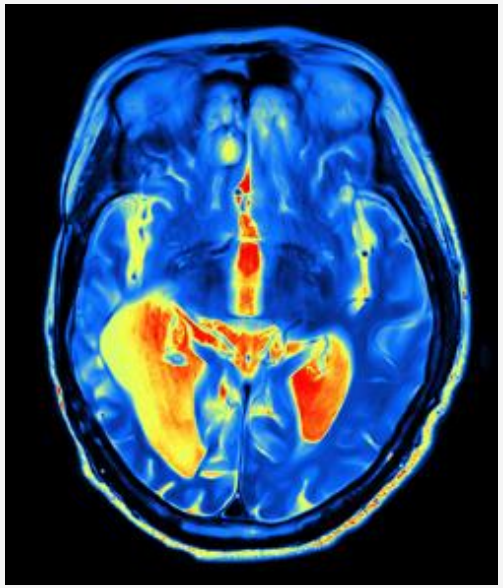
Real World Use Case

- **Training Industrial Robots**
- **Training Self Driving Cars**

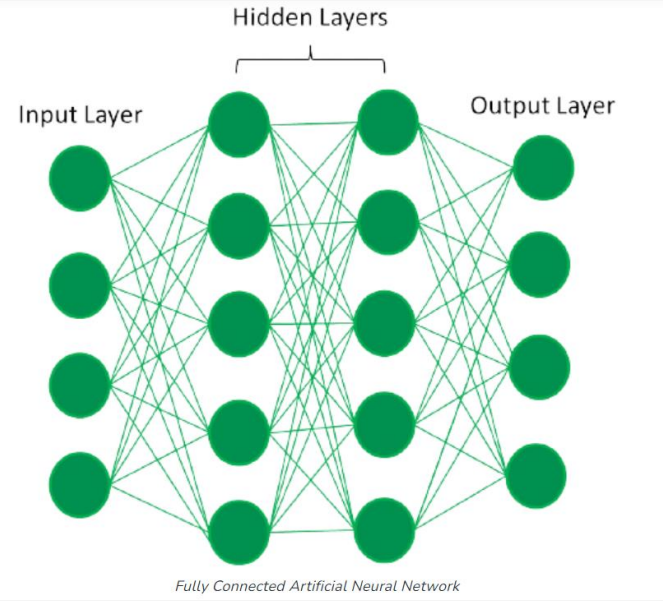
Image Source : internet

Machine Learning – Deep Learning and Artificial Neural Networks

Neural Pathways in Human Brain



Artificial Neural Network

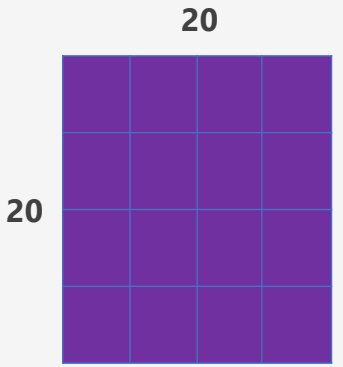


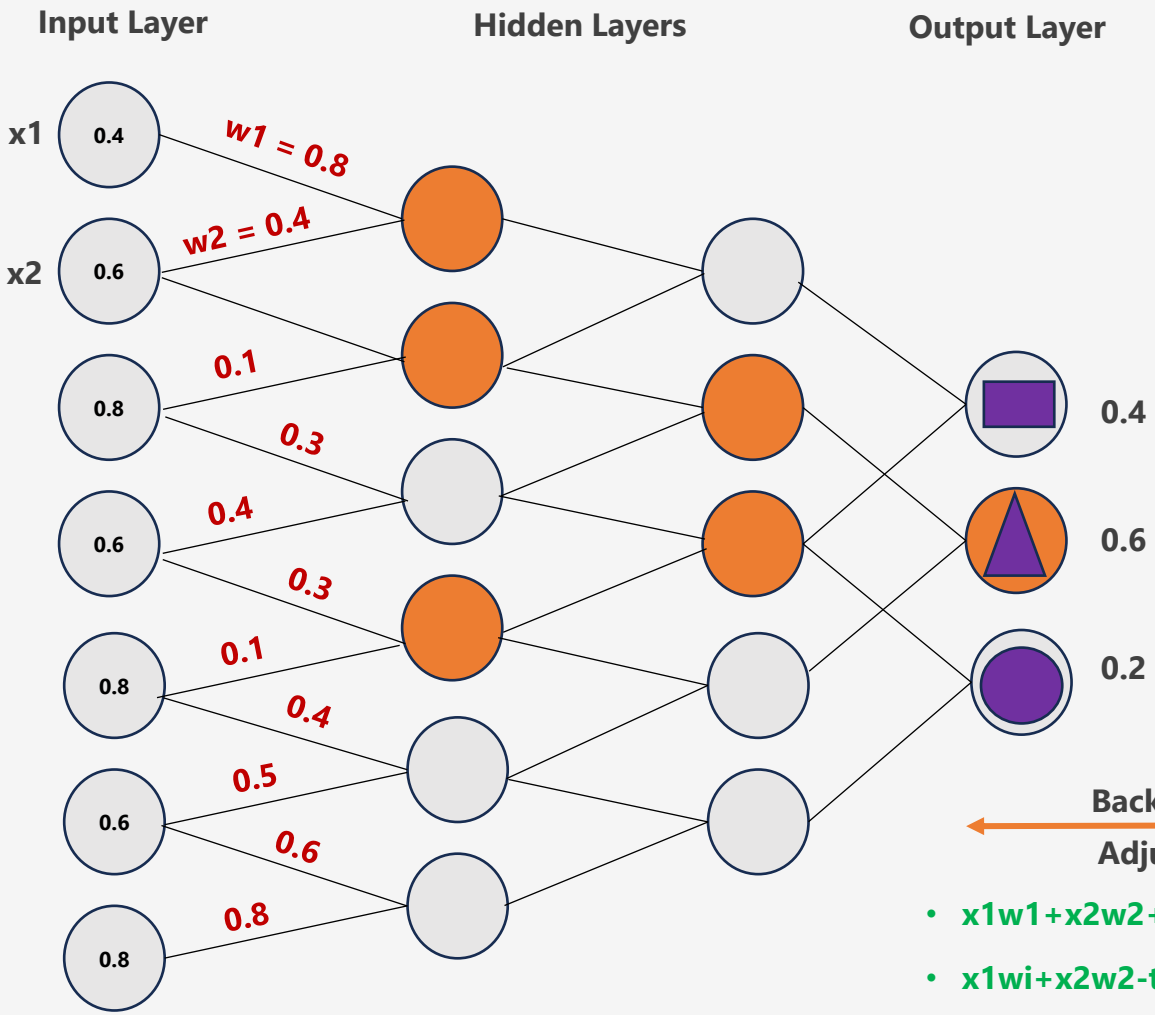
Fully Connected Artificial Neural Network

- **Artificial Neural networks** are designed to **mimic the structure of the human brain**
- Deep Learning is a subset of Machine Learning that **focuses on building artificial neural networks** that **can learn from data**.
- Neural network with **multiple hidden layers and multiple nodes** in each hidden layer is known as a **deep learning (>3)**

Image Source : internet

Machine Learning – Deep Learning with Artificial Neural Networks





Output	Error
1	+0.6
0	-0.6
0	-0.2

Backward Propagation

Adjust Weights

- **x1w1+x2w2+… > threshold value**
- **x1wi+x2w2-threshold >0**
- **x1w1+x2w2+Bias >>> Activation Function**

Key terms

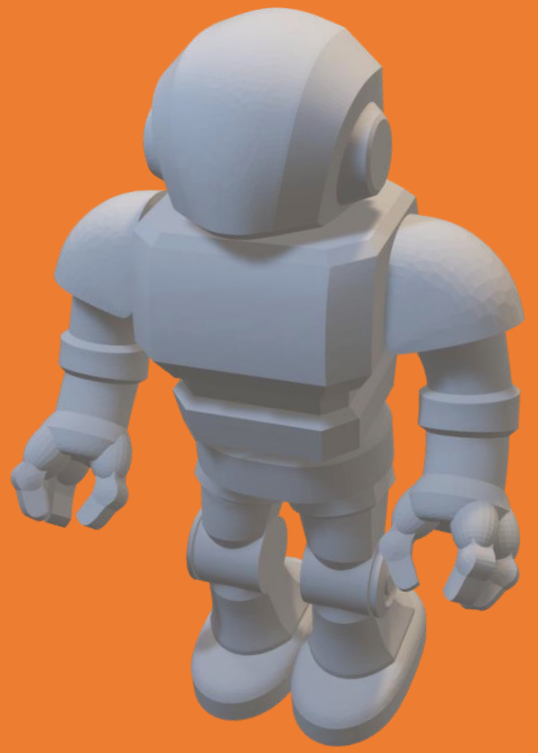
- Neurons (Binary Value)
- Weights
- Bias/Threshold
- Activation Function
- Feedforward network
- Actual Output
- Error
- Forward Propagation
- Backward Propagation

Generative Artificial Intelligence(AI)

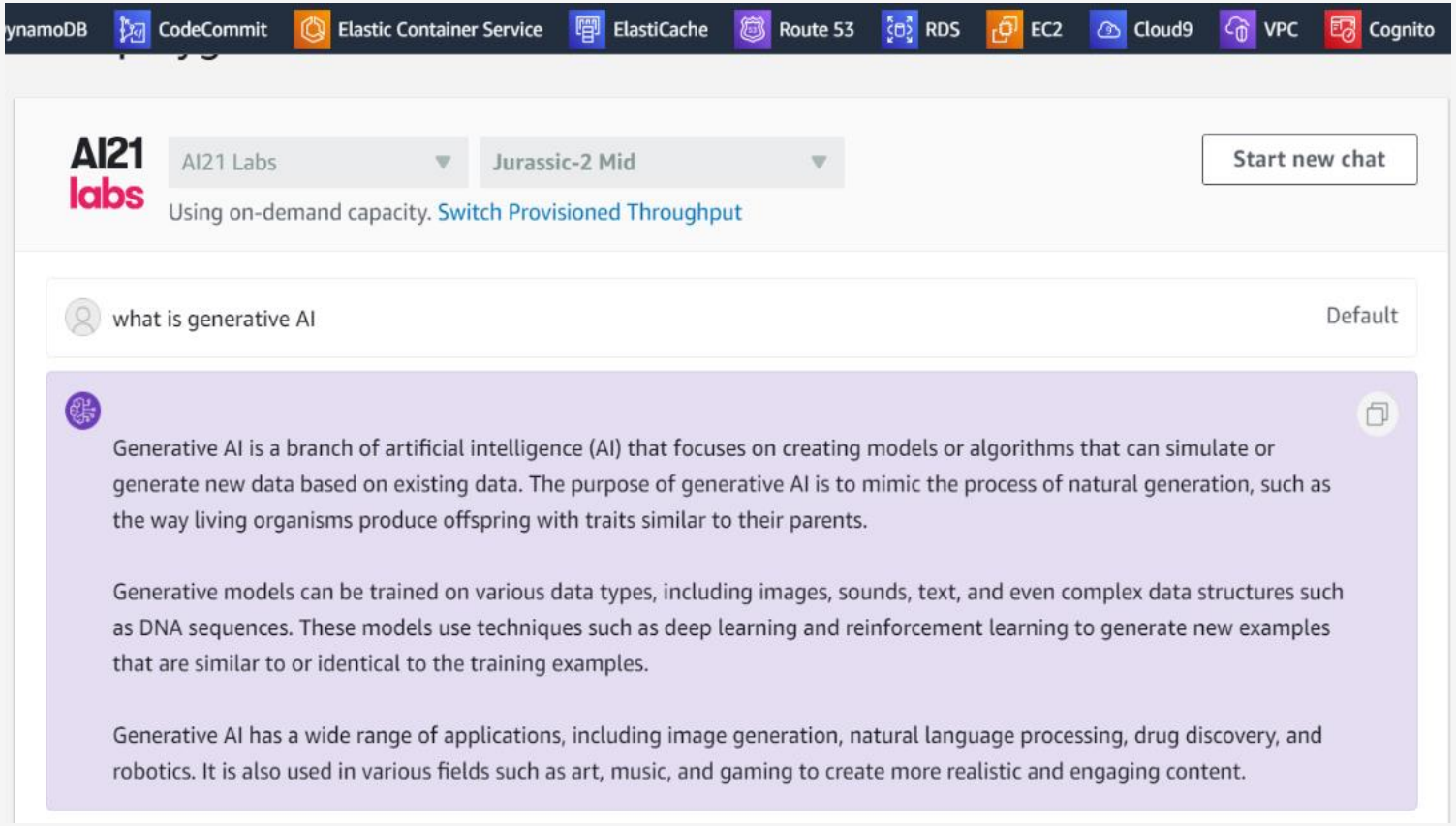
Generative AI

- Generative AI is a **subset of Deep Learning** which focuses on **creating models** capable of **generating new content** :
  - Chat and Text
  - Images and Videos
  - Code Generation

Section on :
Generative AI Fundamentals



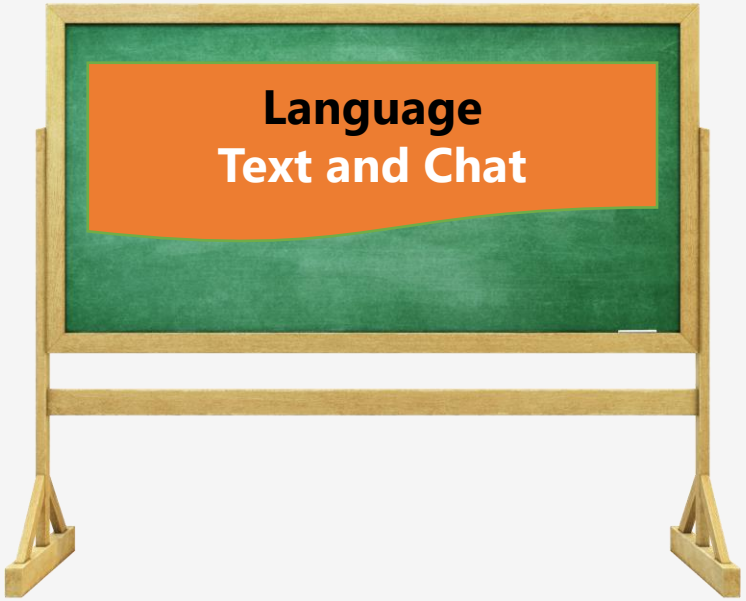
What is Generative AI ?



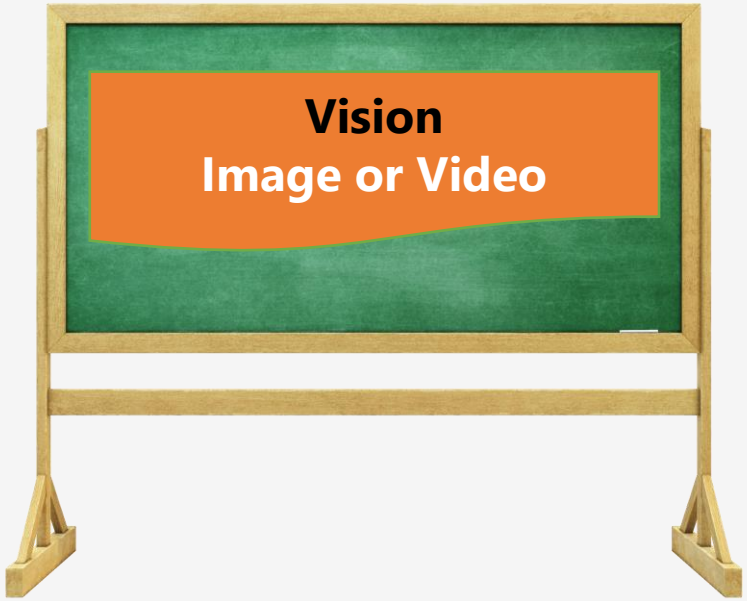
Generative AI refers to

- Branch of AI that use deep-learning **models**
- Generate high-quality text, images, code based on the data they were trained on
- Popular examples of generative AI include
  - ChatGPT
  - Bard

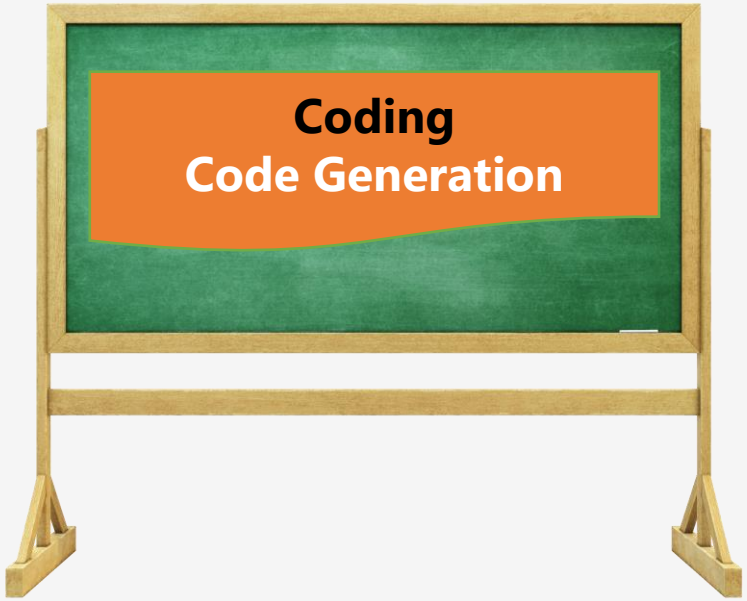
Generative AI – Use Cases



- ✓ Text Summarization - **Healthcare**
- ✓ Content Generation – **HR & Legal**
- ✓ Chatbot/ Q&A – **Customer Service**
- ✓ Models : **Jurrasic – 2 form AI21 Labs**



- ✓ Image Generation - **3D-Design for construction industry**
- ✓ Video Generation - **Content for marketing campaigning**
- ✓ Models : **Stable Diffusion from Stability AI**

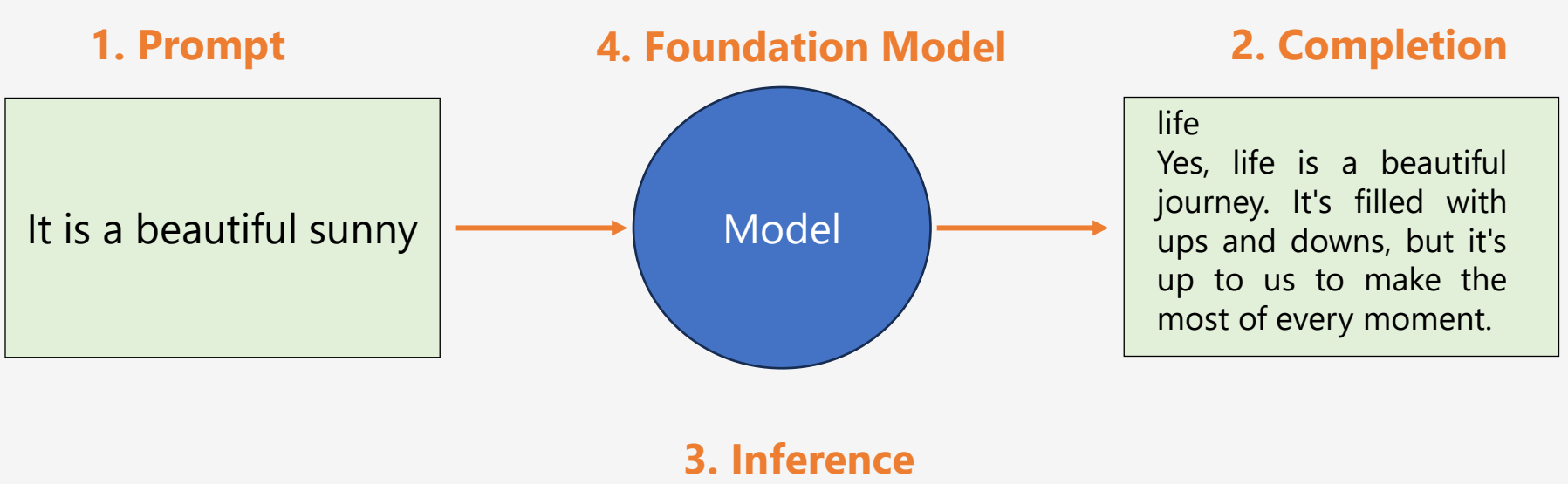


- ✓ Code Generation
- ✓ 15+ Program Lang
- ✓ Multiple IDE's
- ✓ AWS Code Whisperer

Console Walkthrough Amazon Bedrock

- Bedrock offers a choice of **foundation models** from leading AI companies.
- Provides access to variety of FMs from **Amazon, AI21 Labs, Anthropic, Cohere, Meta, and Stability AI**

How Generative AI works - Basic Concepts and Terminology - 1

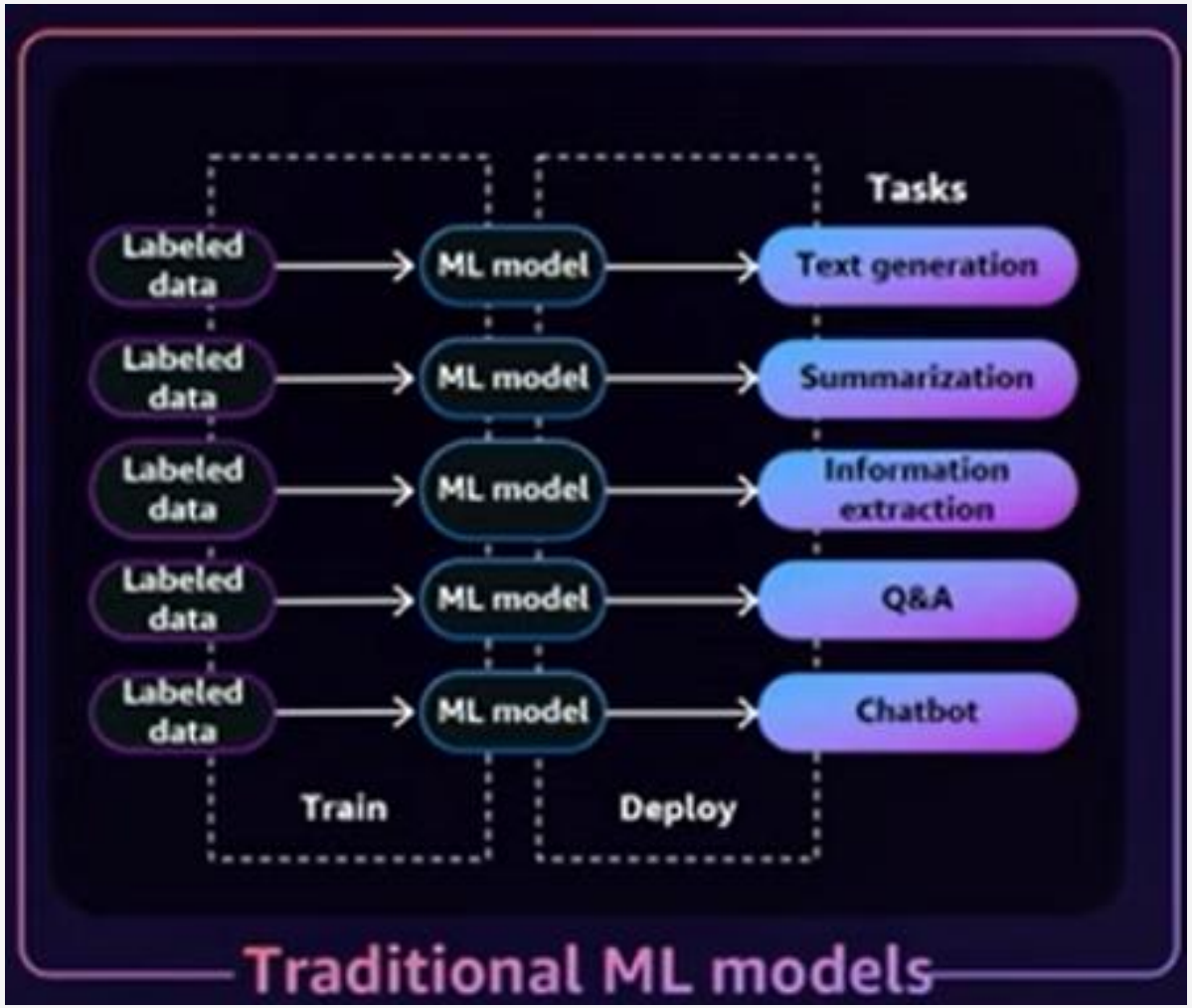


GenAI Terminology

1. **Prompt**
2. **Completion**
3. **Inference**
4. **Foundation Model**

1. Prompt – Input provided to Model
2. Completion – Output of Model
3. Inference : Act of using model to generate text is called Inference

Challenges in Traditional Machine Learning Models

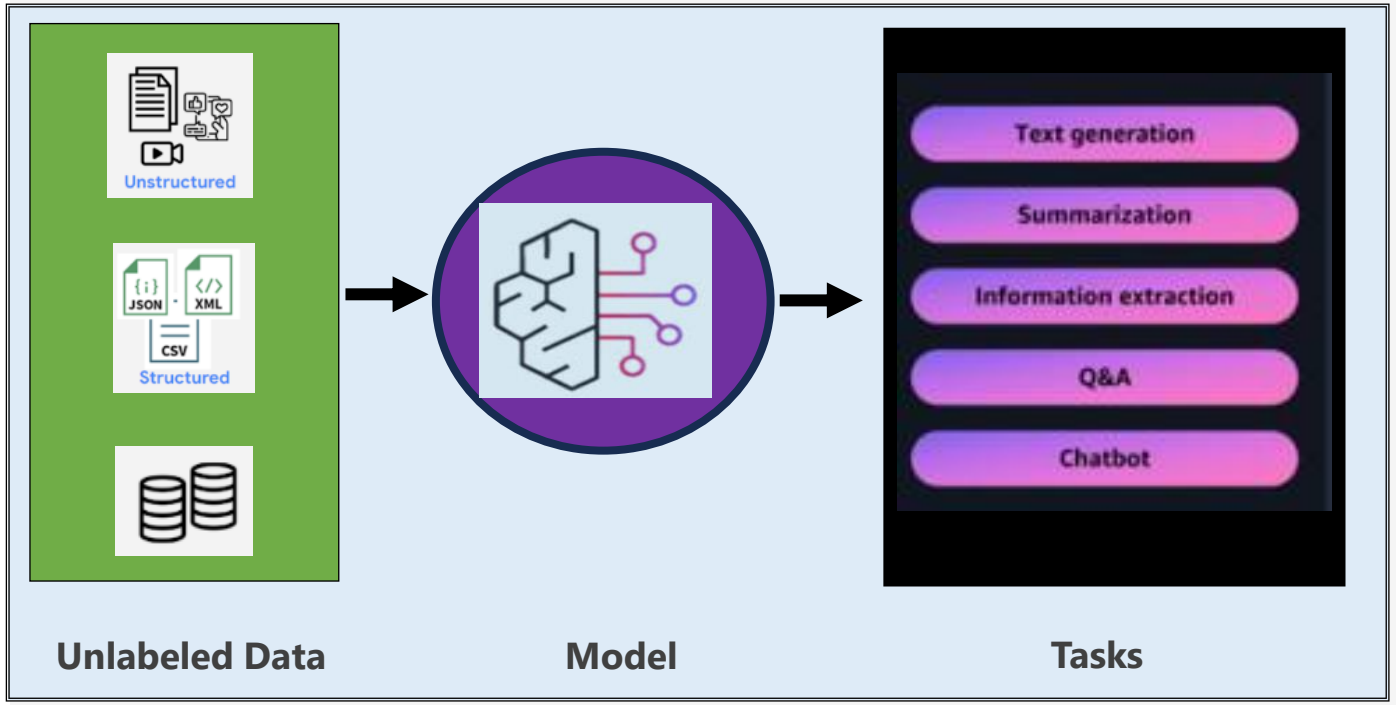


Source : AWS

Challenges in Traditional ML Models

- Each Model focused on specific Task such as
  - Text Generation
  - Summarization
  - Q&A
- Needs labeled data which requires human intervention and can be cost prohibitive

Key Characteristics of Foundational Models



Foundation Model

Organization's can utilize **existing FM's as the base for building** task-specific models and **Adapt** them

Key Characteristics

- **Deep neural networks** techniques
- **Unlabeled data** - Trained on unlabeled data and Self-supervised learning such as internet data - Wikipedia
- **Generalized** – One FM for multiple tasks
- **Pre-trained** – FM's are pre-trained models
- **Large** – Exposed to vast amounts of data, enabling it to learn patterns & features from data.
- **Expensive** - Developing **FM's is an expensive process** that involves **large amounts of computational resources** and **expertise**.

Popular Foundation Models

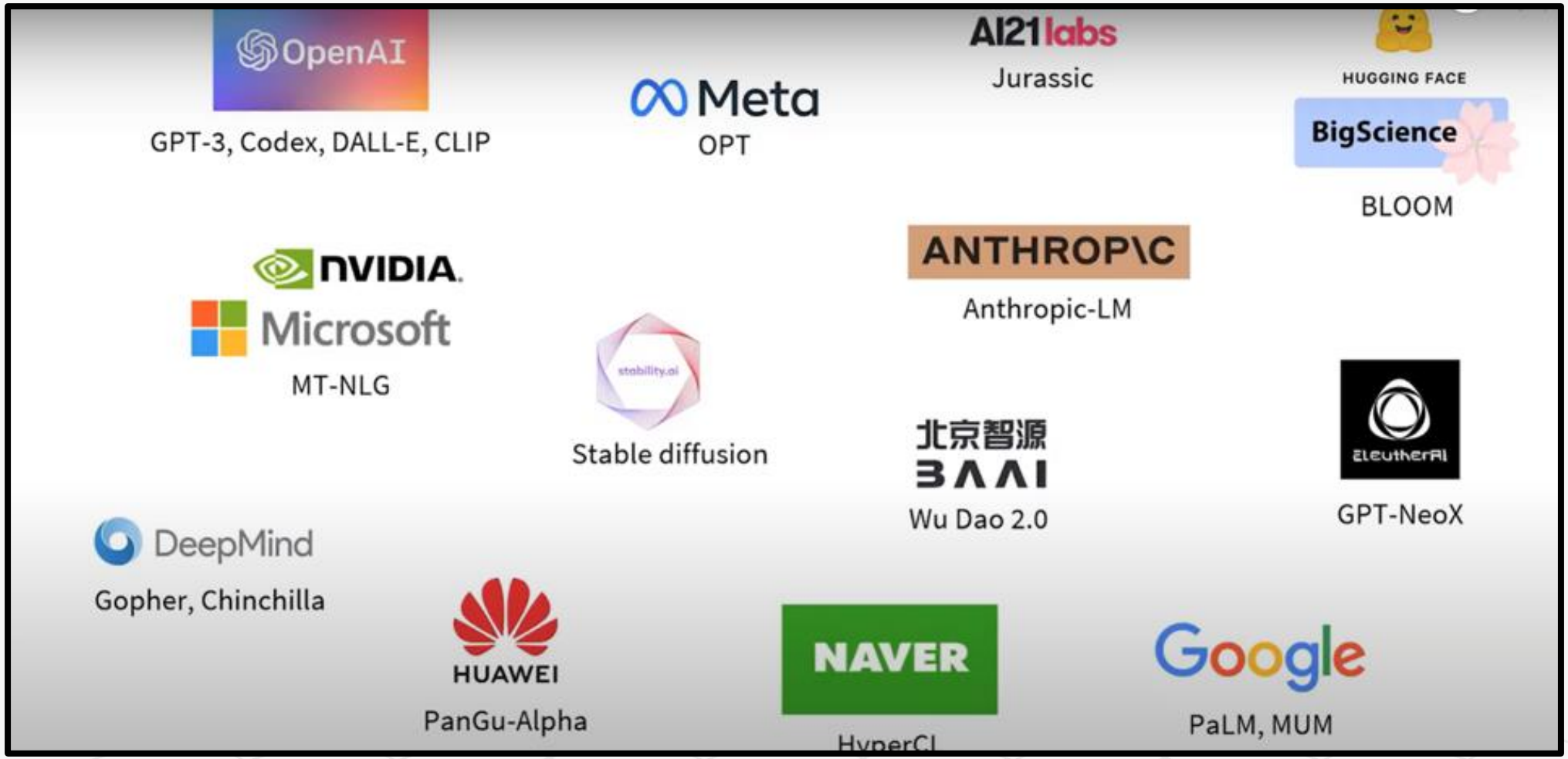
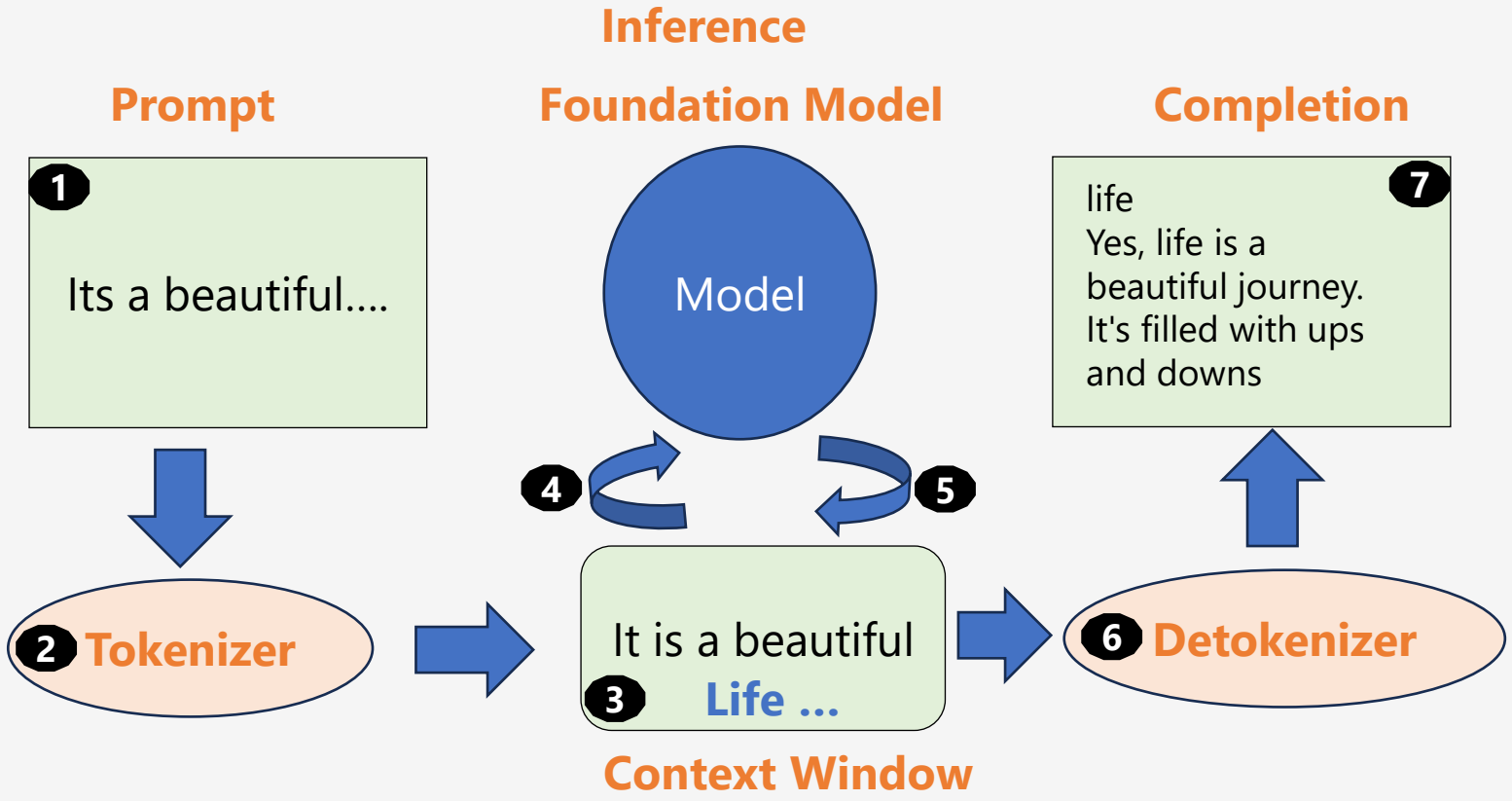
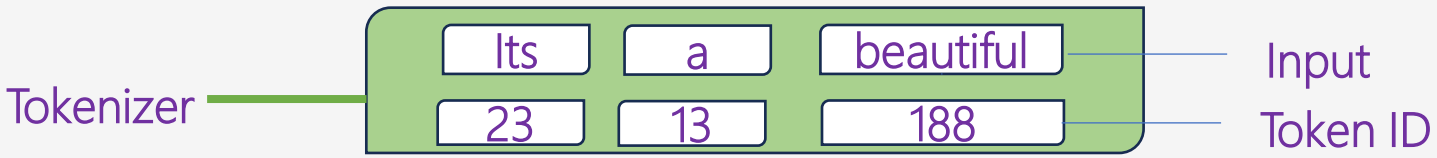


Image Source : internet

How Generative AI works - Basic Concepts and Terminology - 2





GenAI Terminology

- **Prompt**
- **Tokenizer**
- **Context Window**
- **Foundation Model**
- **Max Token/Stop Sequence**
- **Completion**
- **Inference**

- **Context Window -** number of tokens the model can take as input at once

Foundation Model – Tokens, Parameters and Temperature

Tokens

Parameters

Temperature

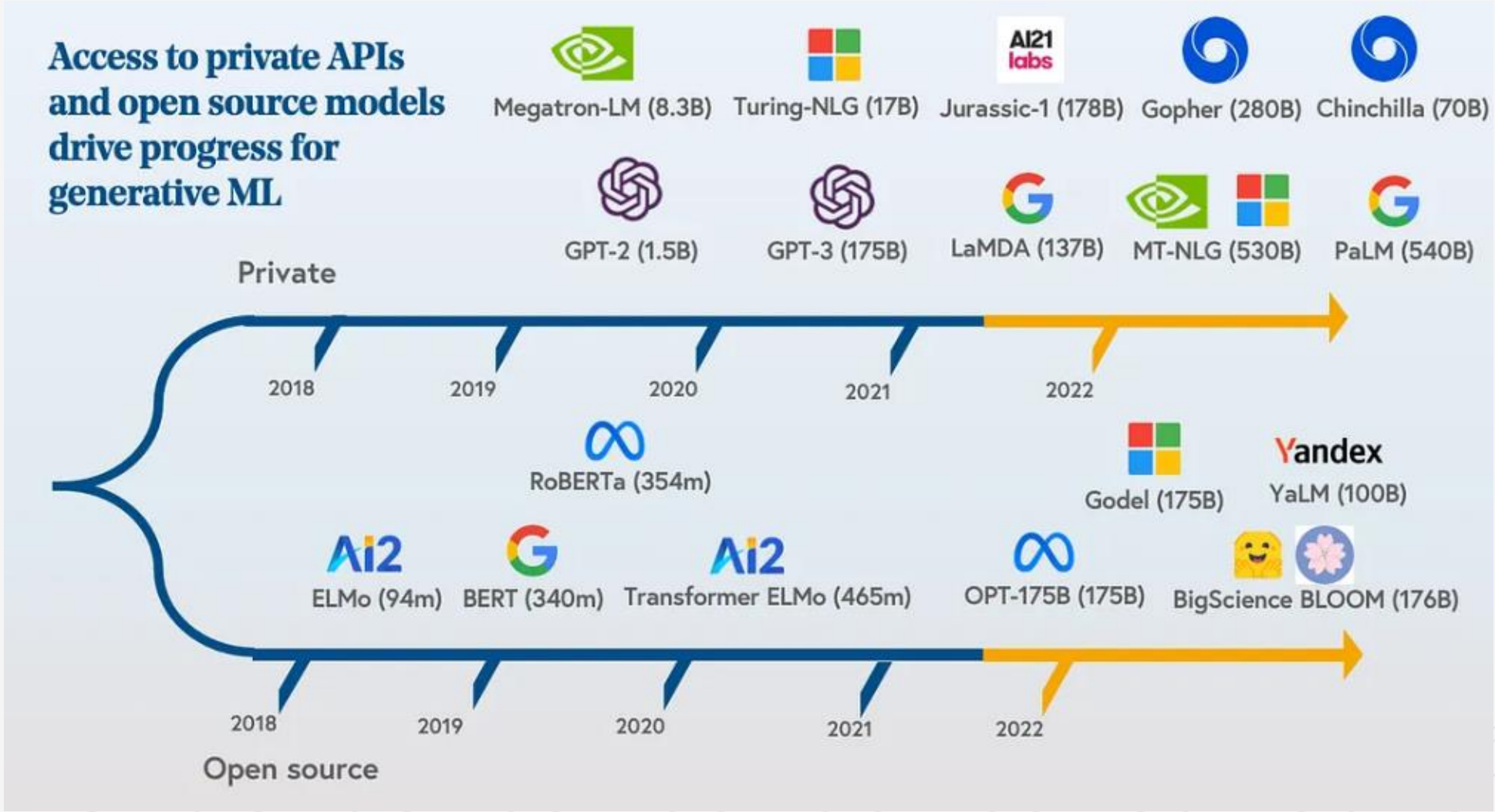
Model	Context length	Number of English pages (1 page = 500 words)
GPT 3.5	4,096	6
GPT 4	8,192	12
Llama 1	2,048	3
Llama 2	4,096	6

Tokens - **One token** generally corresponds to **~4 characters of text** for common English text

Parameters - can be considered count of **connections & weights** between nodes in a neural network.

Temperature - This parameter controls the **creative ability** of your model.

Popular Foundation Models (Parameters perspective)

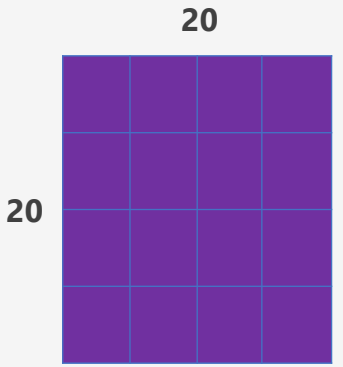


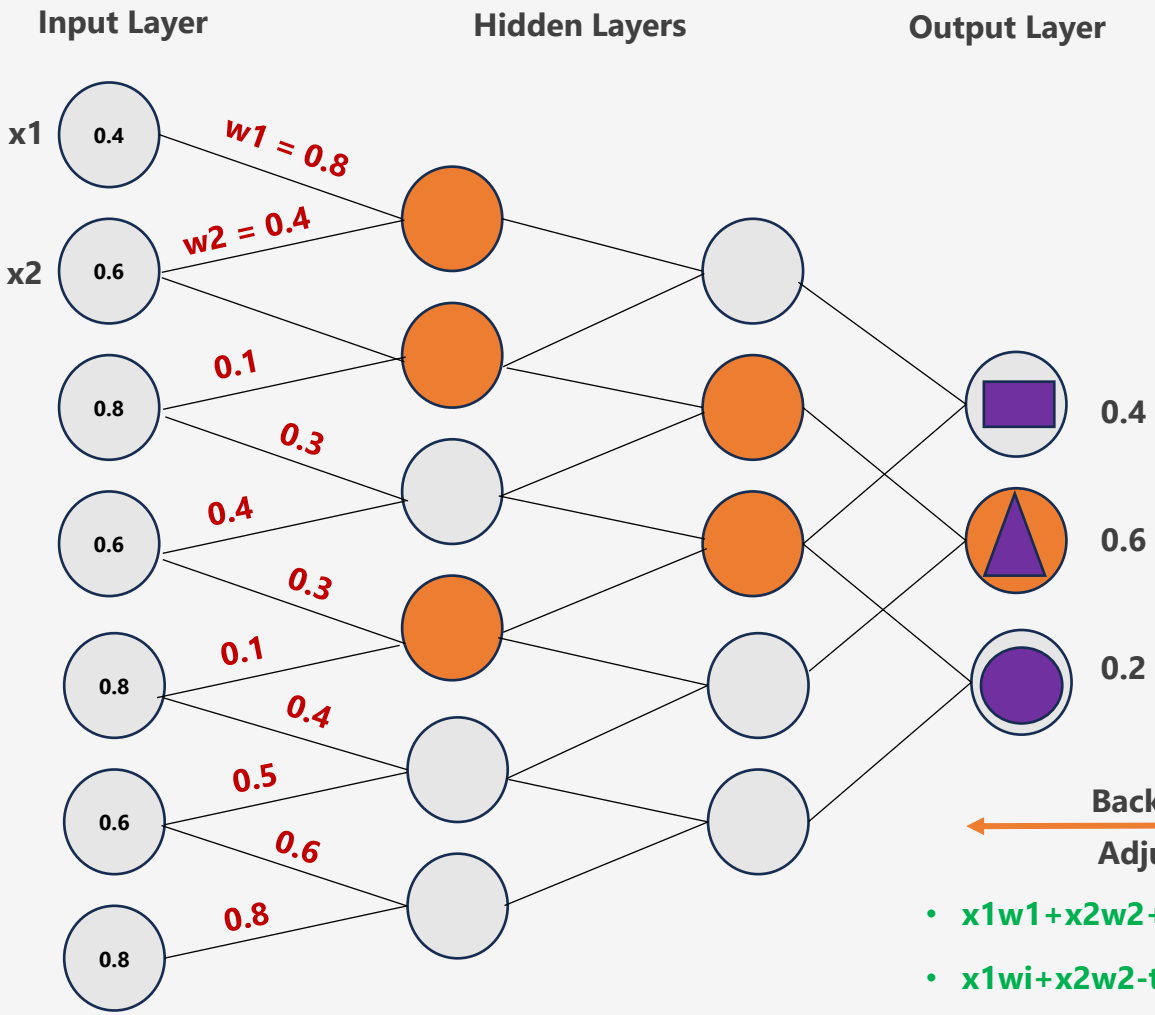
GPT 4

- ~1.8 trillion parameters
- 120 layers

Image Source : internet

Machine Learning – Deep Learning with Artificial Neural Networks





Output	Error
1	+0.6
0	-0.6
0	-0.2

Backward Propagation

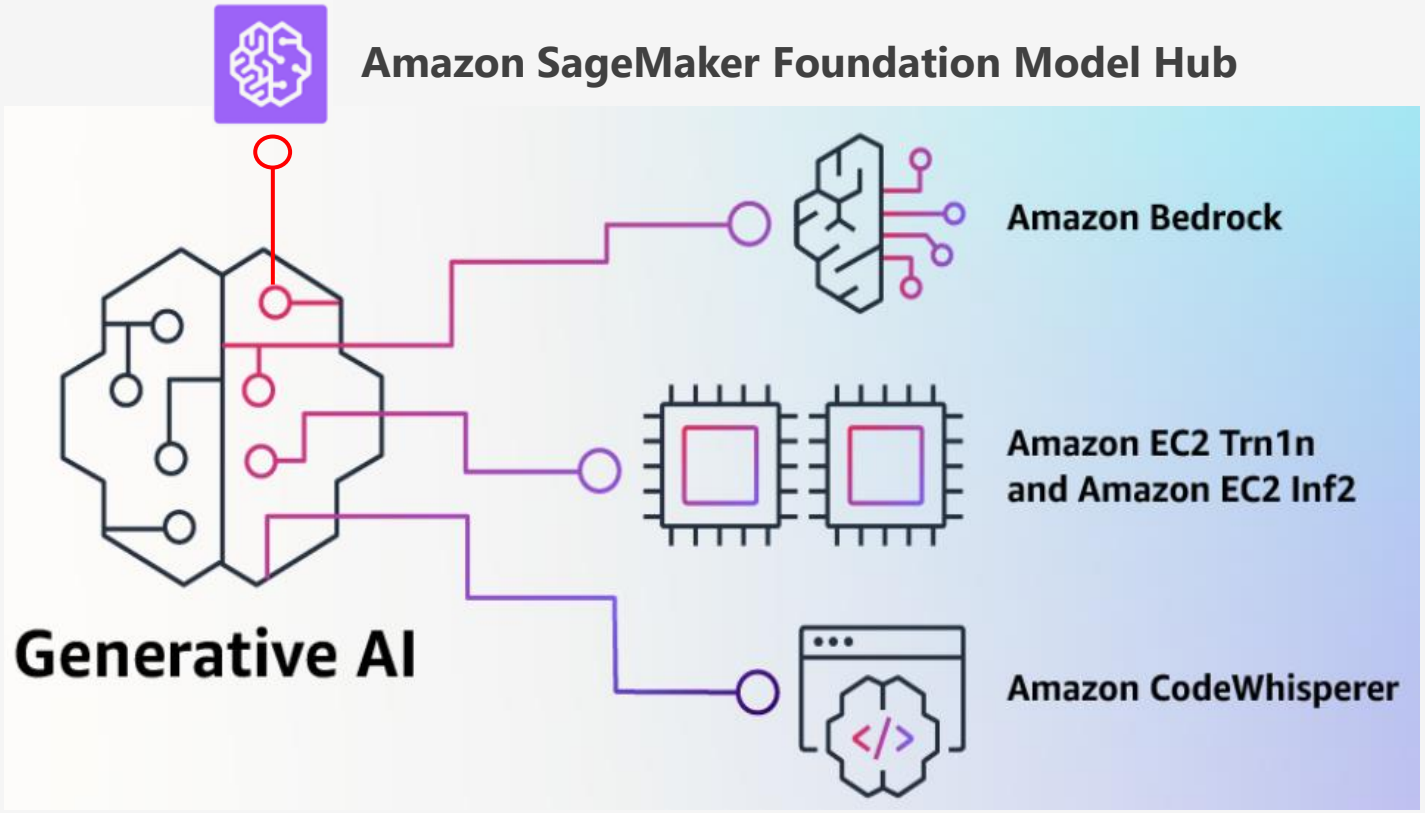
Adjust Weights

- **x1w1+x2w2+… > threshold value**
- **x1wi+x2w2-threshold >0**
- **x1w1+x2w2+Bias >>> Activation Function**

Key terms

- Neurons (Binary Value)
- Weights
- Bias/Threshold
- Activation Function
- Feedforward network
- Actual Output
- Error
- Forward Propagation
- Backward Propagation

Service Offerings in Generative AI from AWS



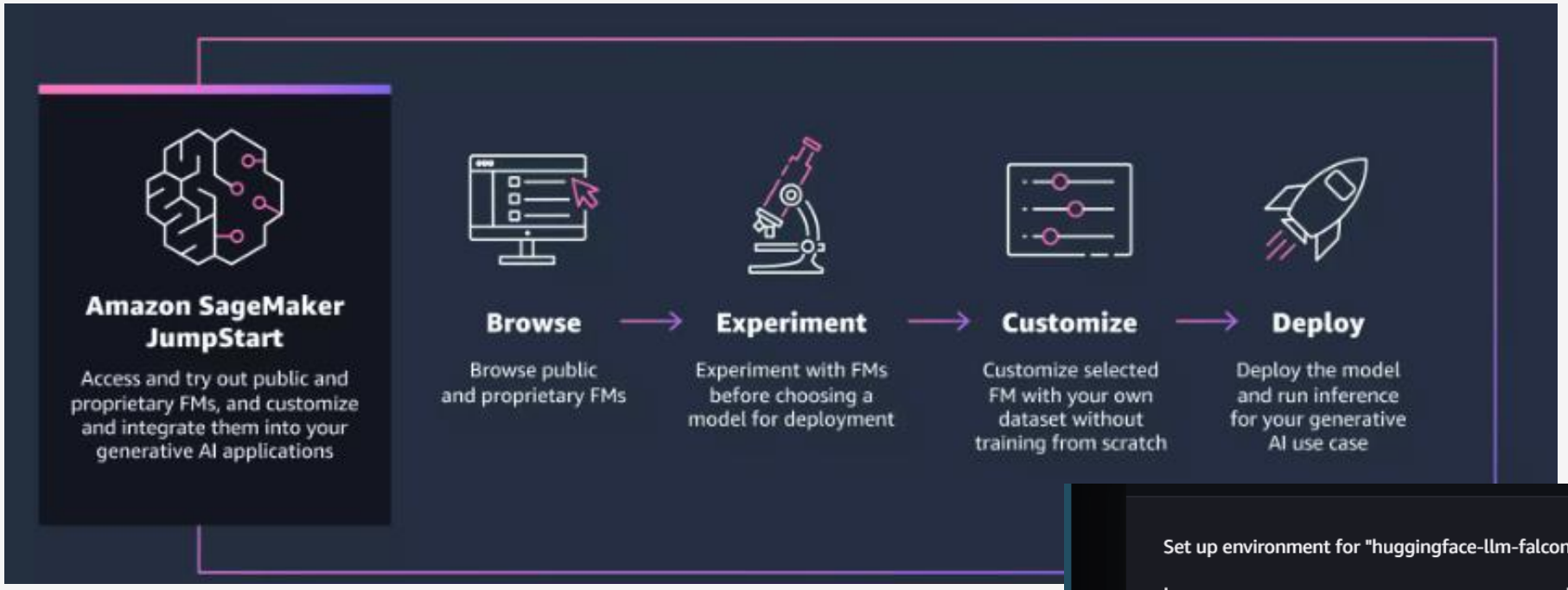
1. **Amazon EC2 Trn1n and EC2 Inf2**
   - IaaS (self managed) for ML experts to build, train and deploy your own models
2. **Amazon SageMaker JumpStart**
   - Provides pretrained, open-source models
   - Deploy open source FMs with custom configuration
3. **Amazon Bedrock**
   - Fully managed serverless service that offers a choice of high-performing FMs from leading AI companies.

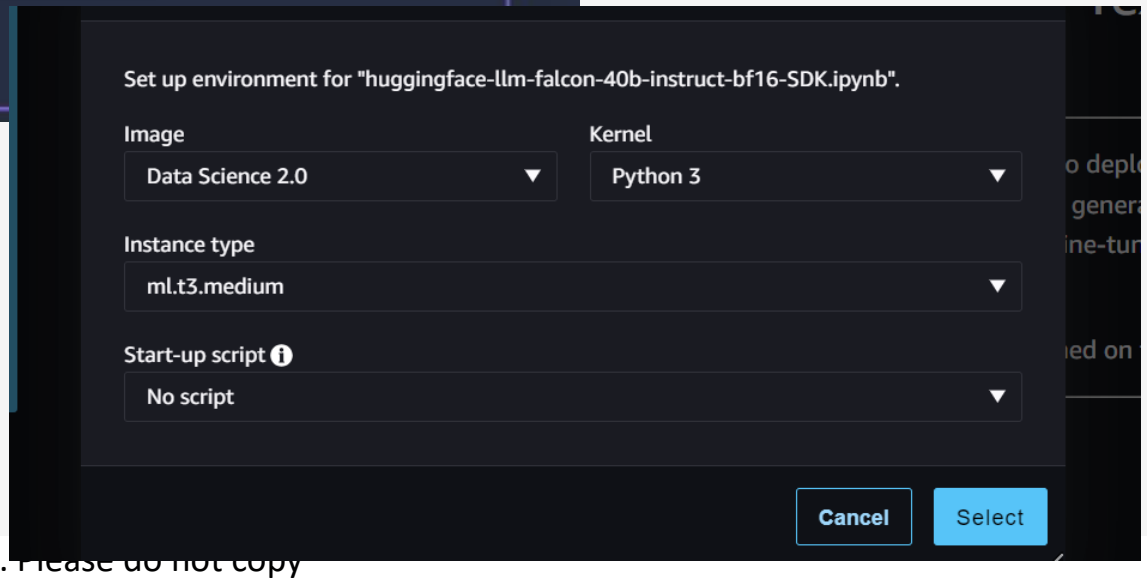
4. **Amazon Code Whisperer**
   - An AI coding companion in 15+ prog. languages for code generation

Train and Deploy own Models on Amazon EC2 Trn1n and EC2 Inf2

Source : AWS

AWS Service Offerings in Generative AI

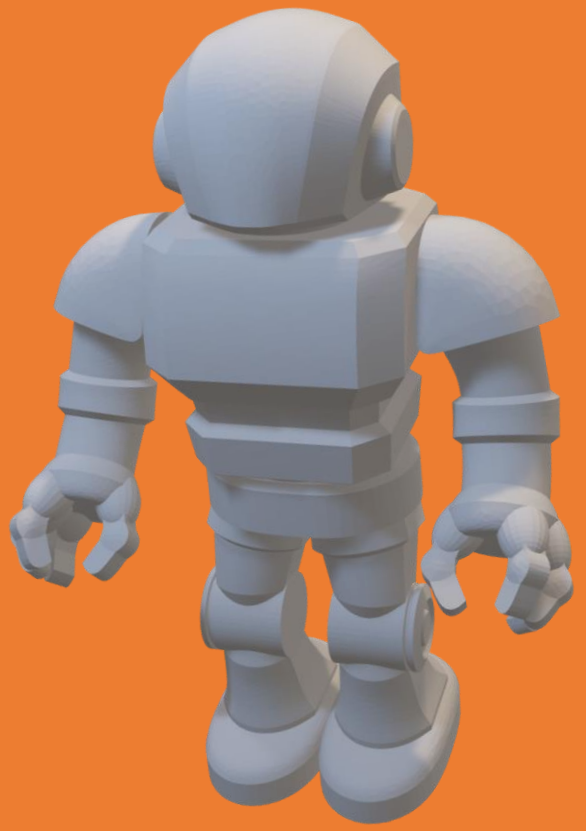




Foundational Models available on SageMaker JumpStart

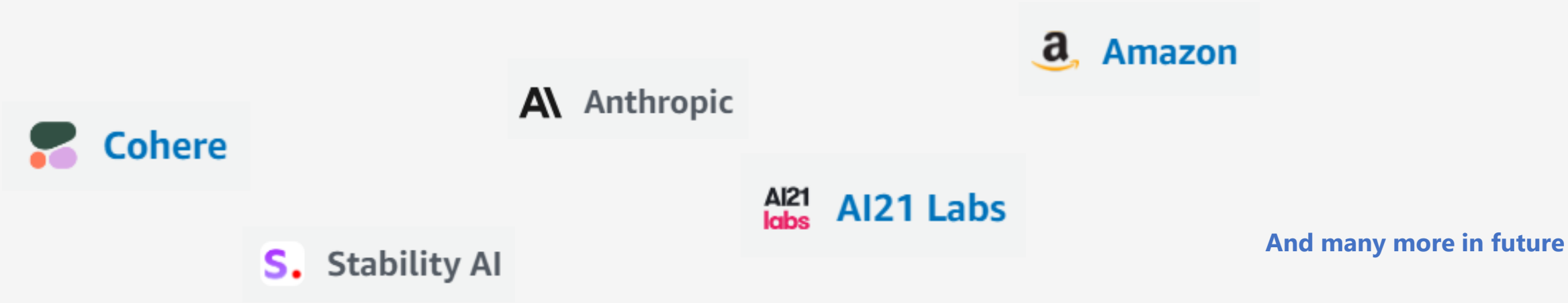
Source : AWS

Section on :
Amazon Bedrock – Deep Dive



What is Amazon Bedrock ?

- Amazon Bedrock is a **fully managed, serverless service from AWS**
- Makes **base Foundation Models from Amazon and third-party model providers** accessible through **an API**.



How does Amazon Bedrock work – High Level ?

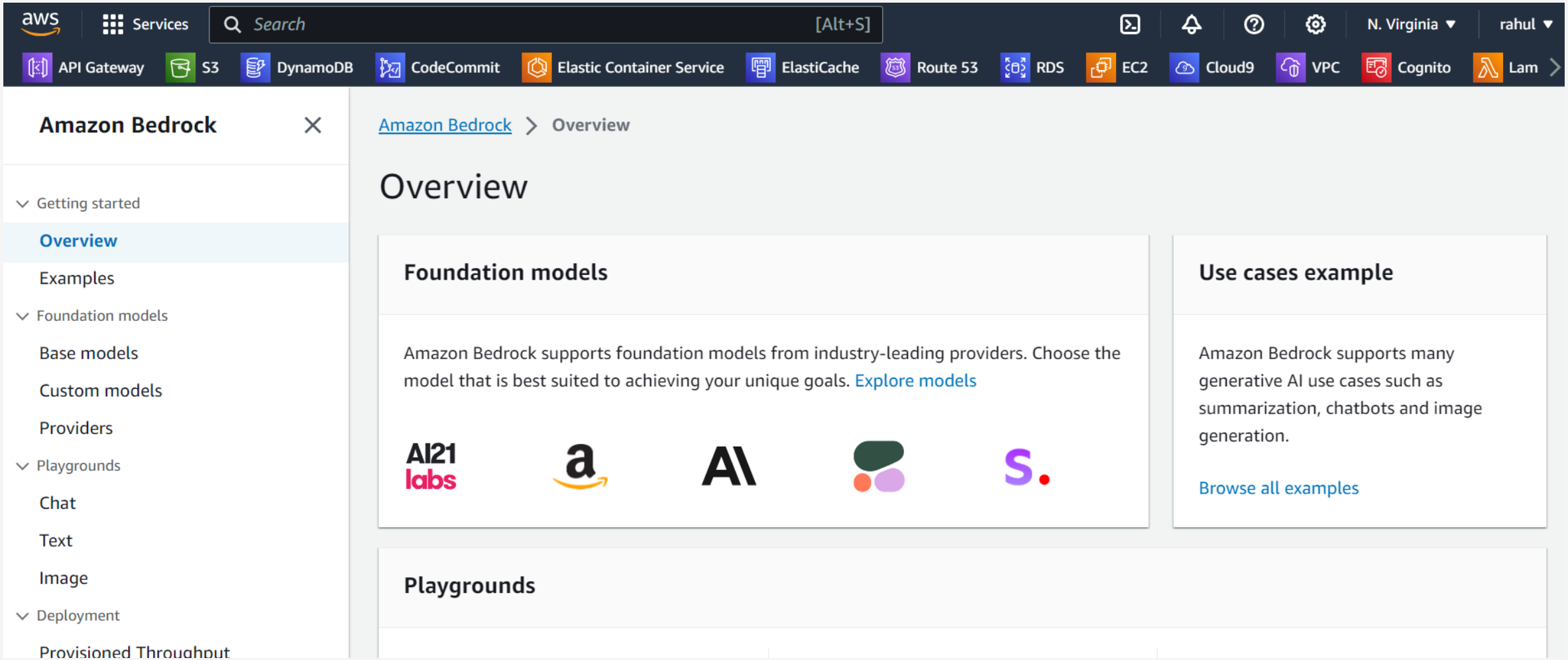
Prompt = Who is the Best Tennis Player ?

```
body=json.dumps({"prompt": user_prompt,"temperature": 0.9,"max_tokens":400}),
contentType = "application/json",
accept = "*/*",
modelId='cohere.command-text-v14')
```

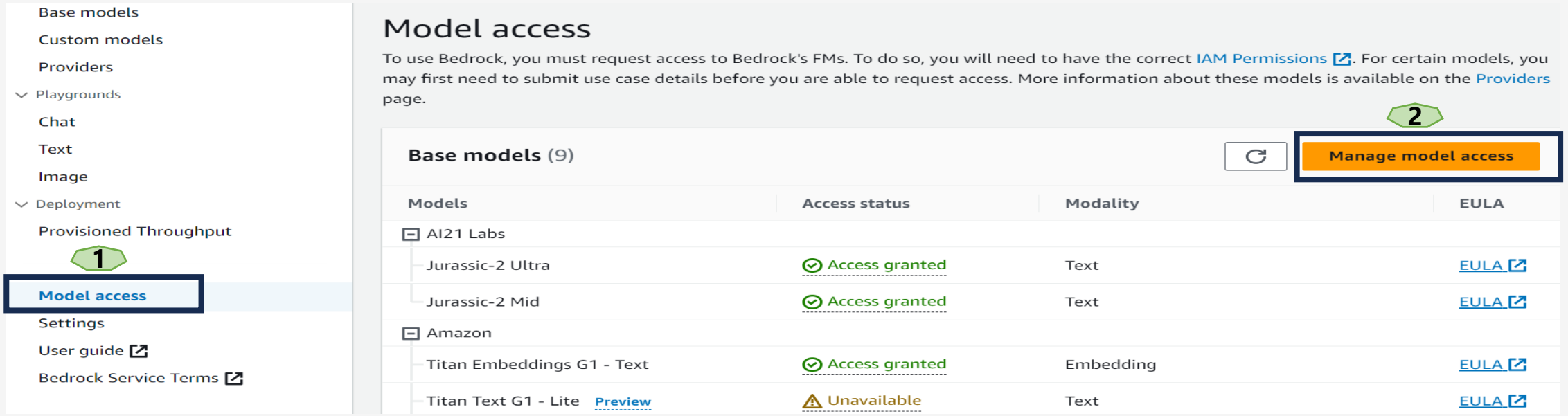
Key features of Foundation Models Amazon Bedrock supports

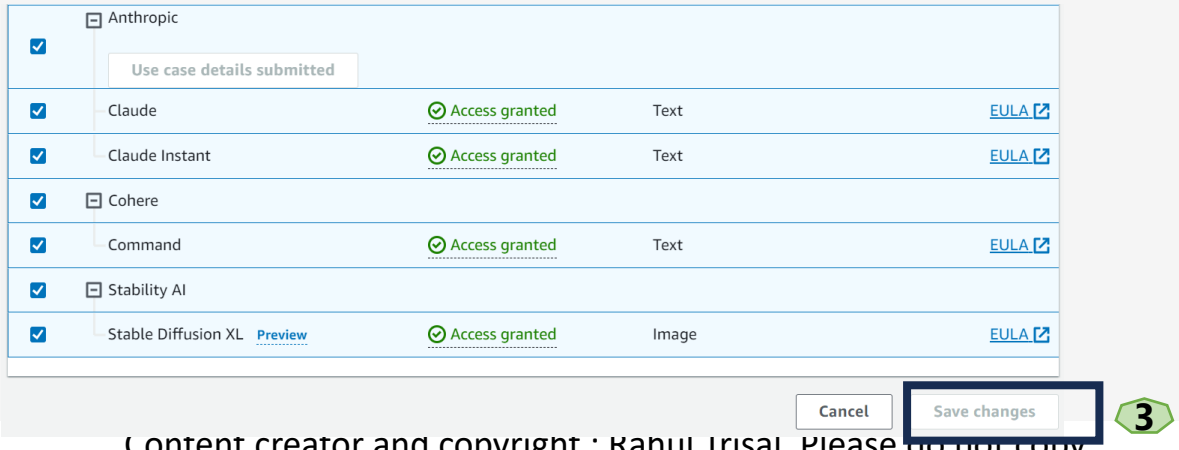
Source : AWS

Amazon Bedrock – Console Walkthrough



Amazon Bedrock – Request Access to Models



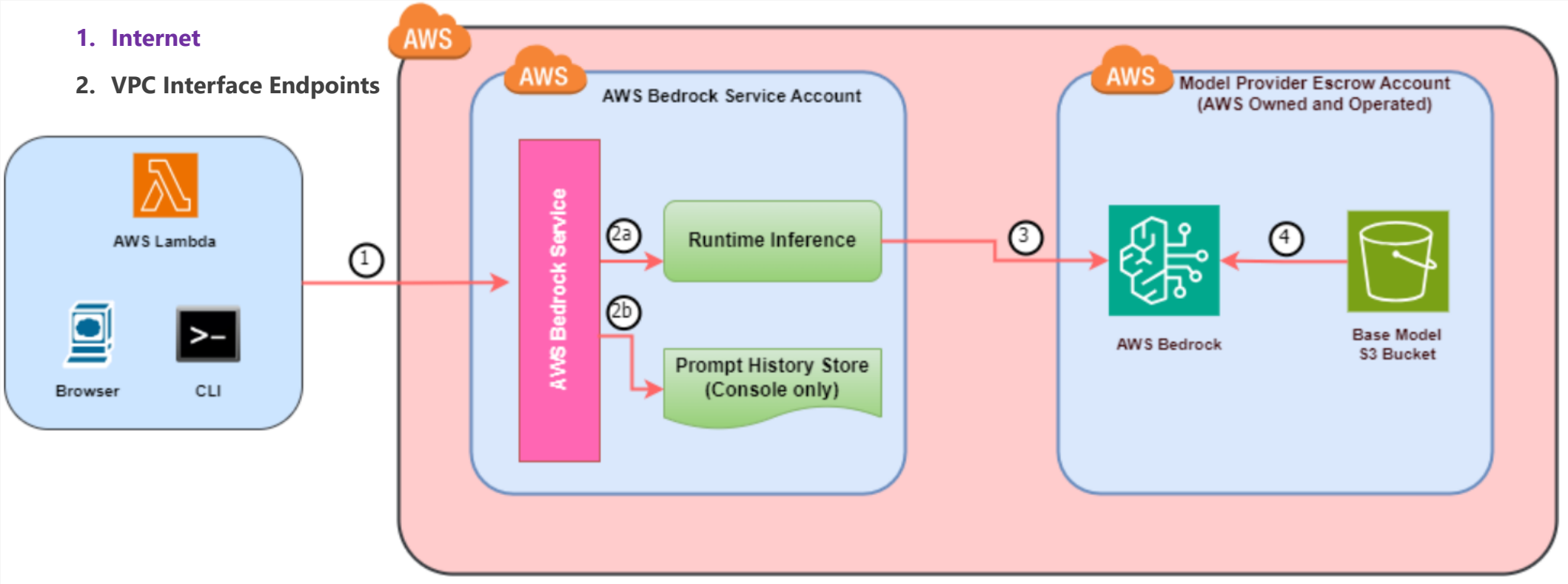


How does Amazon Bedrock work – High Level ?

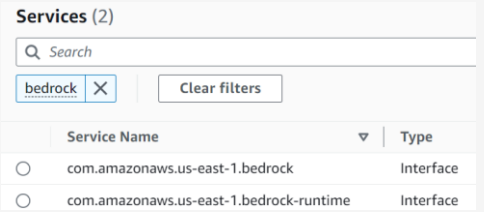
Prompt = Who is the Best Tennis Player ?

```
body=json.dumps({"prompt": user_prompt,"temperature": 0.9,"max_tokens":400}),
contentType = "application/json",
accept = "*/*",
modelId='cohere.command-text-v14')
```

Amazon Bedrock – Architecture – Part 1

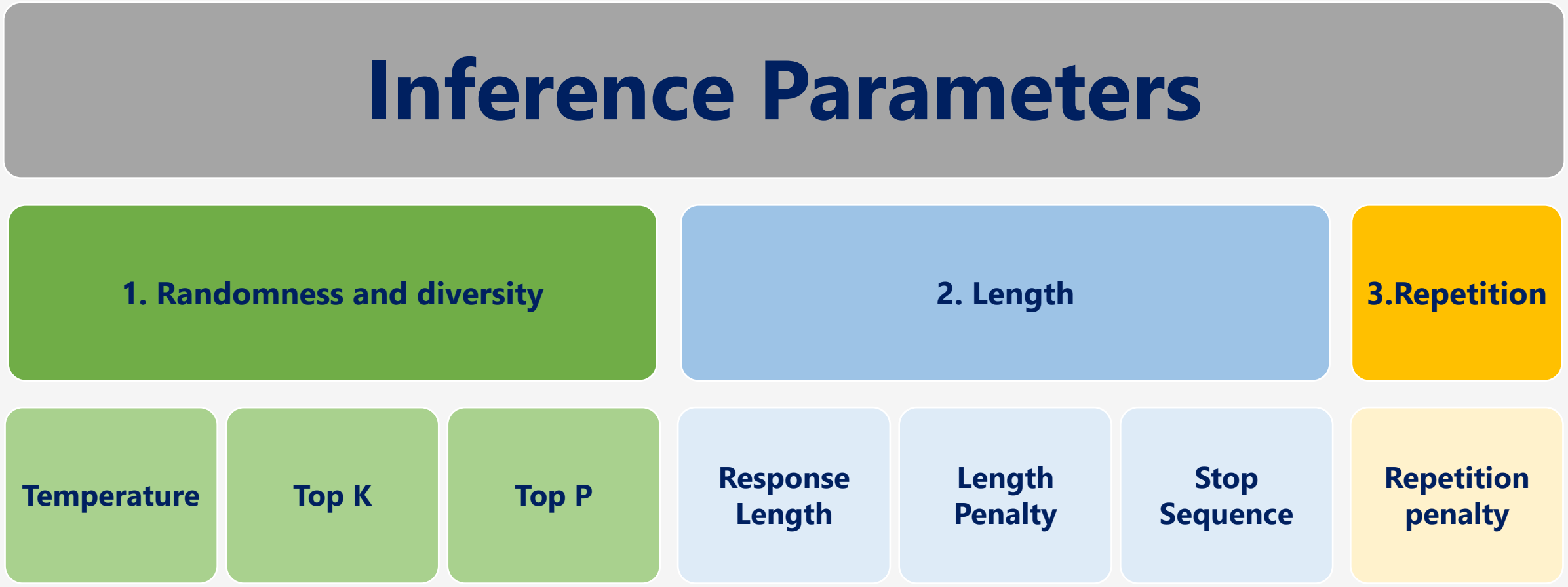


- Runtime Inference : Used to **redirect to right model endpoint** based on the API request
- Base Model : **Baseline model** provided to every AWS account



Foundation Models and Impact of Inference Parameters

Inference parameters influence the response generated by the model.



Inference Parameter – 1. Randomness and Diversity

Prompt : I hear the hoof beats of …..

Parameter	Possible Values	Description	Example
Temperature*	Foundation Models use : **Probability to construct the words in a sequence**		• Horse --- > 0.4 • Wind ----- > 0.2 • Unicorns -- - > 0.1 • Change in distance --- > 0.05
	Low value (closer to zero)	Model tends to select the **higher-probability words.**	• Horse
	High value (closer to 1 or 5)	Temperature further away from zero, the model may select a **lower-probability word**	• Change in distance
Top K	1-500	**Cutoff** where the model no longer **selects the words.**	if K=50, the model selects from **50** of the **most probable words**
Top P	0.01-0.99	**Caps choices** based on the **sum of their probabilities.**	If Top P=0.5, Horse and Wind selected

*** For Cohere FM**

Inference Parameter – 2. Length

Prompt : Write an essay on horse

Parameter	Possible Values	Description
Length : Controls the **length** of the **generated response**. Helps **control cost.**		
Max Length	1- 4096 tokens	Specify the **maximum number of tokens** to use in the generated response.
Stop Sequence	Any keyword	• Configure up to **four sequences** that the model recognizes. • After a **stop sequence**, the model stops **generating further tokens.** • The **returned text doesn't contain the stop sequence**.

*** For Cohere and Anthropic**

Inference Parameter – 3. Repetition

Parameter	Possible Values	Description
Repetition : Help **control repetition** in the generated response.		
Presence penalty	0-5	Use a **higher value to lower the probability** of generating new tokens that **already appear at least once in the prompt** or in the completion.
Count penalty	0-1	Use a **higher value to lower the probability of generating new tokens** that already appear at least once in the prompt or in the completion. Proportional to the number of appearances.
Frequency penalty	0-500	Use a **high value to lower the probability** of generating new tokens that already appear at least once in the prompt or in the completion. The value is proportional to the frequency of the token appearances (**normalized to text length**)
Penalize special tokens		Reduce the probability of repetition of special characters – Whitespaces, Punctuations…

*** For AI21 not anthropic**

Amazon Bedrock - Pricing

Two Modes - On-Demand and Provisioned Throughput

On-Demand Mode

- Pay for what you use, with no time-based term commitments
- Text generation models
  - Charged for every input token processed and every output token generated
- Image generation models
  - Charged for every image generated.
- Embeddings models
  - Charged for every input token processed

AWS Bedrock - Pricing

Provisioned Throughput

- Primarily designed for **large consistent inference workloads** that need **guaranteed throughput.**
- Can **purchase model** units for a **specific base or custom model** with **time commitment (1-6 months)**
- Custom models can only be accessed using Provisioned Throughput.
    - Console Custom Models
    - Pricing

Pricing Examples

- A121 Labs
- Anthropic

AWS Bedrock - Pricing

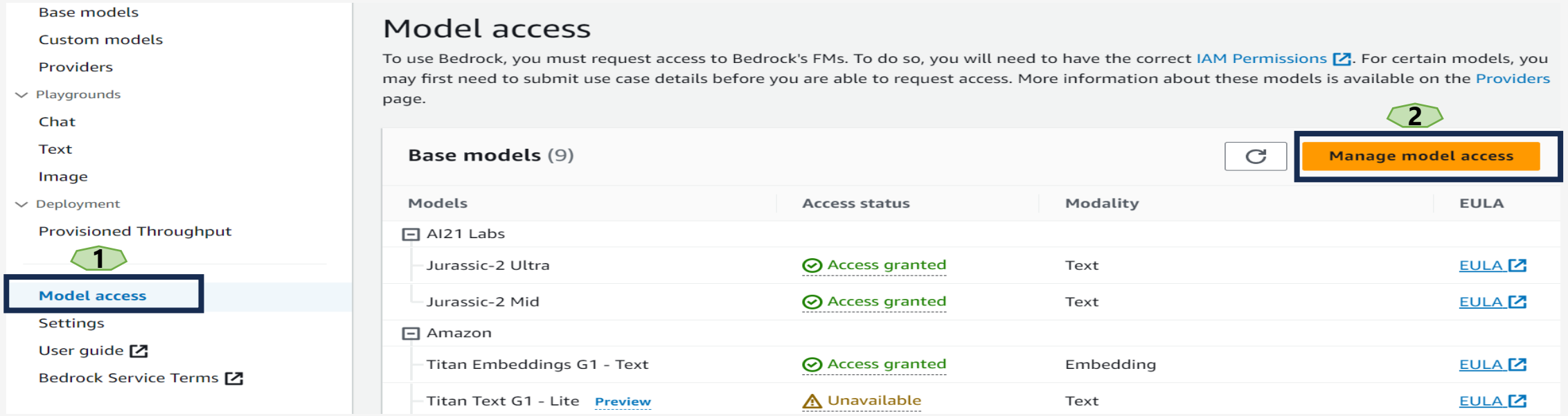
Token Limits - Tokens shared between prompt and completion.

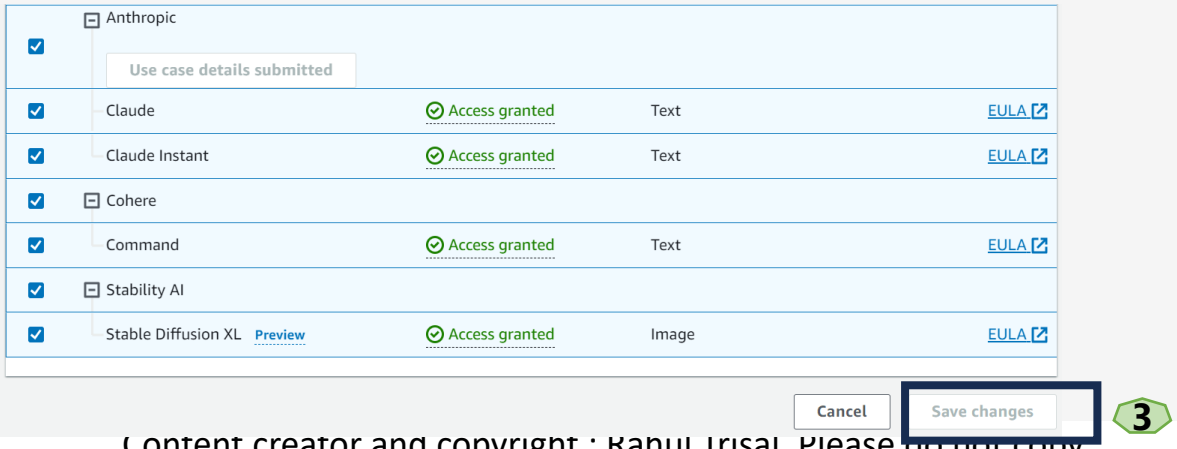
https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them

Media and Entertainment Industry Use Case :

Generate Movie Poster Design

Use Case Implementation Pre-Requisites 1 : Request Access to Models





Use Case Implementation Pre-Requisites 2 : Upgrade boto3 version

1. Create a AWS Lambda Function – bedrock-boto3Upgrade and add import boto3
2. Check the boto3 version. Should be > 1.28.63
3. Use following command to check version - print(boto3.__version__)
4. Upgrade the boto3 version for AWS Lambda Function using Lambda Layer - https://repost.aws/knowledge-center/lambda-python-runtime-errors
   - Add Layer Version ARN
   - Check the boto3 version. Should be > 1.28.63
5. Code for Bedrock invocation from AWS Lambda Function
   - Link - https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/bedrock-runtime.html

Manually create a Lambda layer that uses the latest Boto3 version

1. **Create a lib folder:**

   LIB_DIR=boto3-udemy03/python

   mkdir -p $LIB_DIR

2. **Install the library to LIB_DIR:**

   pip3 install boto3 -t $LIB_DIR

3. **Zip all the dependencies to /tmp/boto3-mylayer.zip:**

   cd boto3-udemy03

   zip -r /tmp/boto3-udemy03.zip .

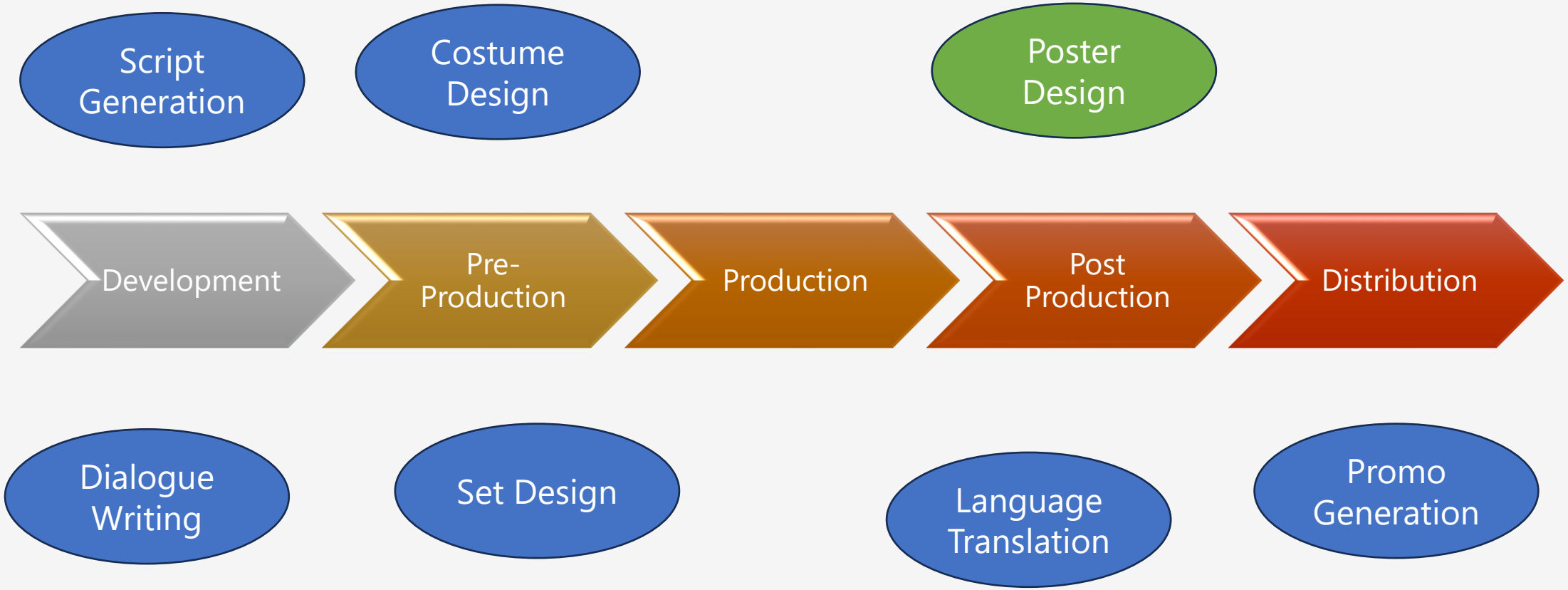
4. **Publish the layer:**

   aws lambda publish-layer-version --layer-name boto3-udemy03 --zip-file fileb:///tmp/boto3-udemy03.zip

Media and Entertainment Industry Use Case :

Generate Movie Poster Design

Generative AI – Media and Entertainment Industry(Film Making)

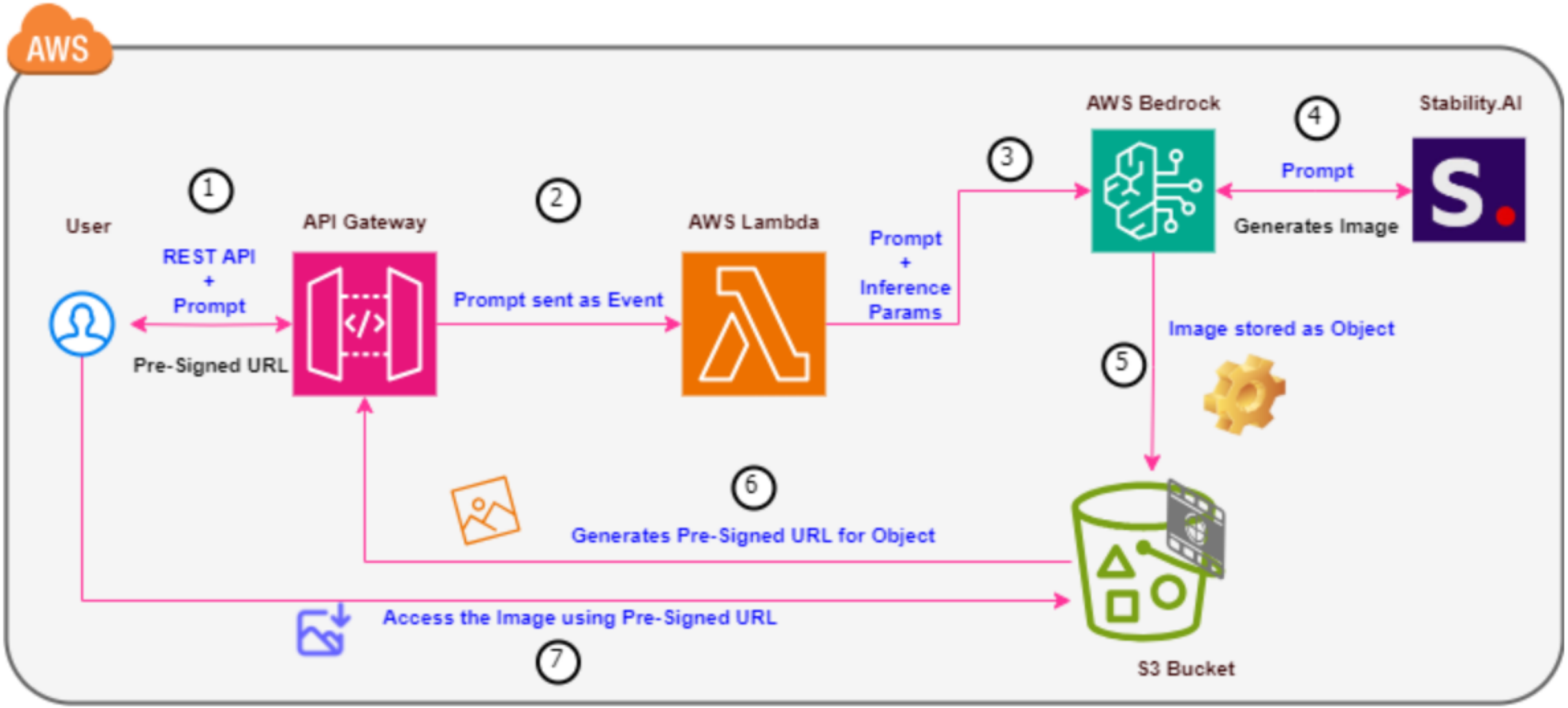


Stages in Movie making and potential role of Generative AI

Generative AI Use Case – Generate Movie Poster Design

Prompt : Generate movie poster photograph with hero shooting the villain with lots of guns in a scenic location with leaves and bullets flying

Movie Poster Design – Generative AI Architecture to Build



Movie Poster Design - Broad Implementation Steps

1. Create an **S3 Bucket** – movieposterdesign01
2. Create a **AWS Lambda Function** - moviePosterDesignFunction
    - **Connect to AWS Bedrock** – Stability Diffusion Model and Generate an Image based on Prompt
    - **Store the Image as an Object** in the **S3 Bucket**
    - Generate a **Pre-Signed URL** for image generated
    - Send the **URL** as a **response to AWS API Gateway**
3. **Create a REST API using the AWS API Gateway** to allow user to pass **the 'prompt'** and **view image using Pre-Signed URL -** movePosterDesignAPI
4. Test using **Postman API Tool**

Generative AI with Bedrock – Lambda Function Pre-requisites

1. Create a AWS Lambda Function
2. Check the boto3 version. Should be > 1.28.63
3. Use following command to check version - print(boto3.__version__)
4. Upgrade the boto3 version for AWS Lambda Function using Lambda Layer - https://repost.aws/knowledge-center/lambda-python-runtime-errors
   - Add Layer Version ARN
   - Check the boto3 version. Should be > 1.28.63
5. Code for Bedrock invocation from AWS Lambda Function
   - Link - https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/bedrock-runtime.html

Movie Poster Design - Implementing Lambda Function Steps

1. Create an IAM Role and Increase timeout limit

    - Full Access to Bedrock – Policy Name - AmazonBedrockFullAccess
    - Full Access to S3 – Policy Name- AmazonS3FullAccess

2. Code for Bedrock invocation from AWS Lambda Function

3. Configure Test Event

    { "prompt": “generate an image of a dog with sunset"}

Movie Poster Design : Lambda Coding Steps

Broad Steps for writing Lambda Function

#1. import boto3

#2. Create client connection with Bedrock and S3 Services – Link

#3. Store the input data (prompt) in a variable

#4. Create a Request Syntax - Details from console and documentation(JSON object) - Link

#5. 5a. Retrieve from Dictionary, 5b. Convert Streaming Body to Byte using json load 5c. Print

#6. 6a. Retrieve data with artifact key, 6b. Import Base 64, 6c. Decode from Base64 - Link

Movie Poster Design : Lambda Coding Steps

Broad Steps for writing Lambda Function

#7. 7a. Upload the File to S3 using Put Object Method – Link 7b. Import datetime 7c.

Generate the image name to be stored in S3 - Link

#8. Generate Pre-Signed URL - Link

Movie Poster – API Gateway creation and Lambda Integration

4. Create an REST API from API Gateway

- Resource - movePosterDesignAPI
- Method - GET

5. API Gateway - Method Request

- URL query string parameters – Add input parameter
- Enable - Request Validation

Movie Poster– API Gateway creation and Lambda Integration

6. **API Gateway - Integration Request**

- application/json

{

"prompt" : "$input.params(‘prompt’)”

}

7. Deploy the API to a Stage’(Stage=Dev or any other)

8. Test using API Gateway Console and Postman

Stable Diffusion Inference Parameters

Parameter	JSON object format	Description	Minimum	Maximum	Default
Prompt strength	cfg_scale	• Determines how much the **final image portrays the prompt**. • **Lower number** to increase **randomness** in the generation.	0	30	10
Generation step	steps	• Determines how many times the image is sampled. • More steps result in a more accurate result.	10	150	30
`Seed	seed	• Set as a random number and not needed • Use the **same seed and the same settings** as a previous run to allow inference to **create a similar image**.	0	4294967295	0

https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-diffusion.html

Prompt Design for Stable Diffusion - 1

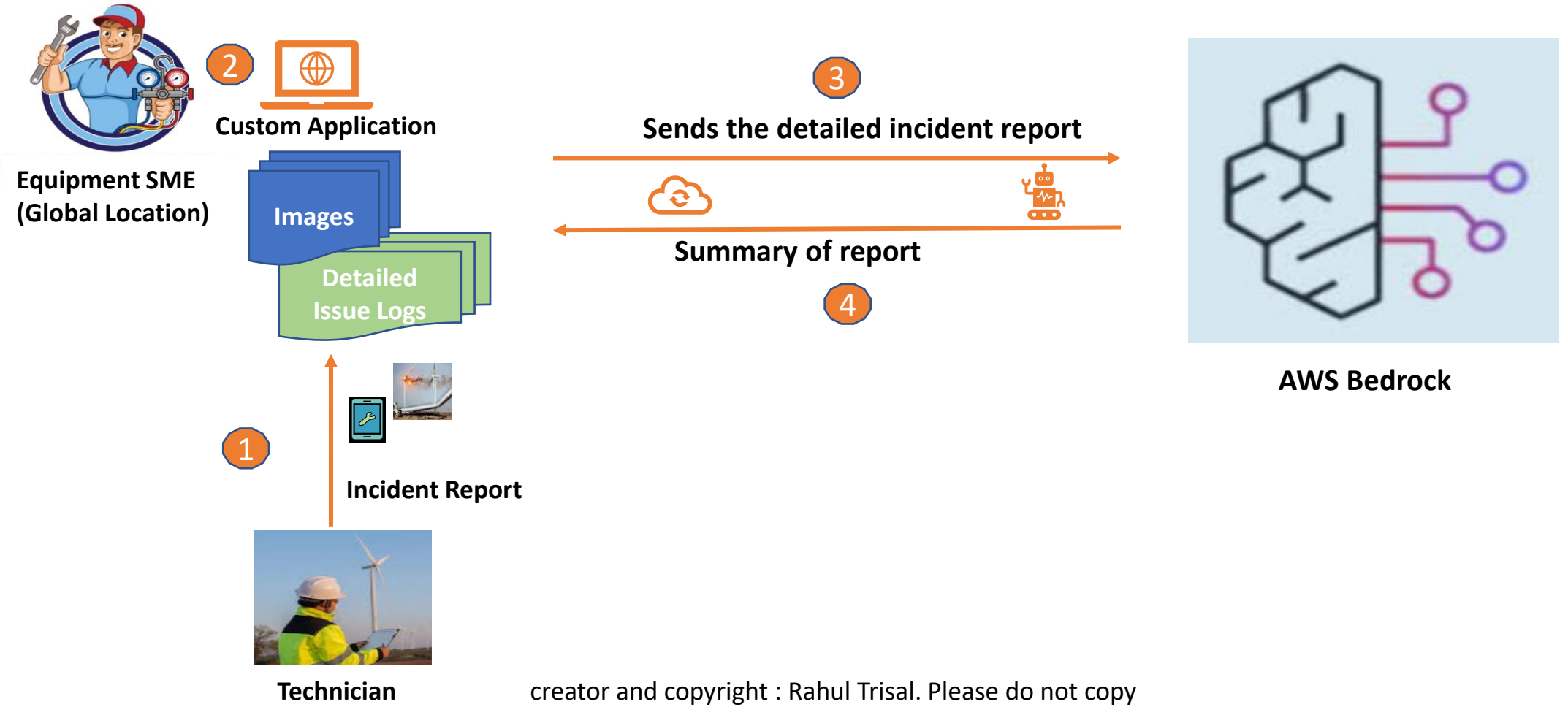
An image of a spy agent fighting in a rival country with guns and helicopters with backdrop of a shopping complex with heavy snow and old Greek architecture building late in the evening with sunsetting behind mountains. The image should be a photograph with Aaron Jasinski style

Use Case – Text Summarization for Manufacturing Industry

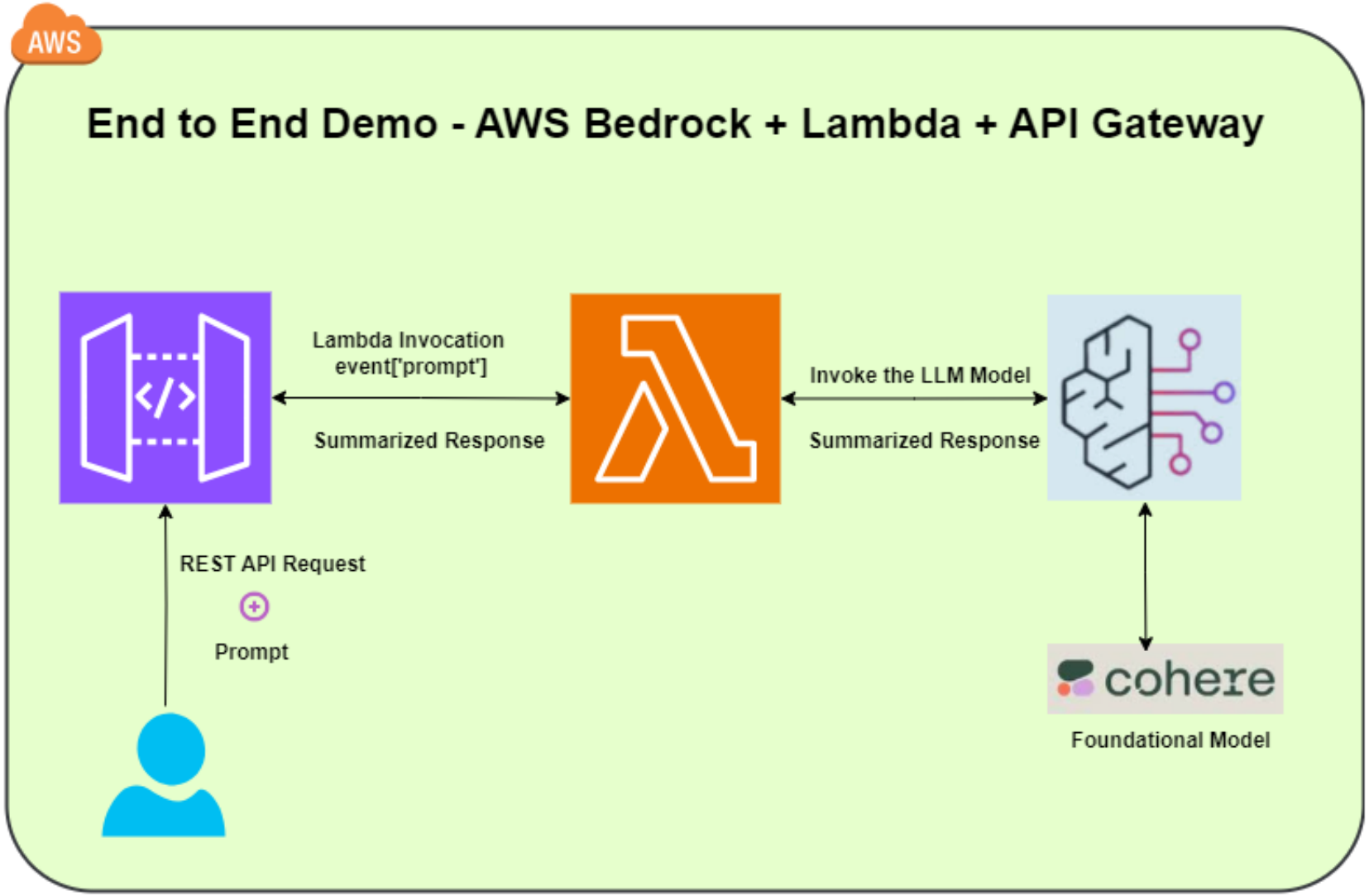
Generative AI – Manufacturing Industry Use Case

Use Case :

Text Summarization using **AWS Bedrock** for **faster issue resolution to improve productivity** of technicians.



Generative AI – Use Case Architecture



Generative AI with Bedrock : Step by Step – Pre-Requisites

1. Create a AWS Lambda Function - demoManufacturing

2. Check the boto3 version. Should be >= 1.28.63

3. Use following command to check version - print(boto3.__version__)

4. Upgrade the boto3 version for AWS Lambda Function using Lambda Layer -

   https://repost.aws/knowledge-center/lambda-python-runtime-errors

   - Add Layer Version ARN
   - Check the boto3 version. Should be > 1.28.63

Generative AI with Bedrock : Step by Step Guide

1. Create an IAM Role and Increase timeout limit

2. Code for Bedrock invocation from AWS Lambda Function

   - Link - https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/bedrock-runtime.html

3. Configure Test Event

```
{  "prompt": "How is weather in Bengaluru"}
```

Generative AI with Bedrock : Step by Step Guide

Broad Steps for writing Lambda Function

#1 **Import boto3** and create **client connection** with bedrock

#2 a. Store the input in a variable, b. print the event

#3. Create Request Syntax - Get details from console & body should be json object - use json.dumps for body

#4. Convert Streaming Body to Byte(.read method) and then Byte to String using json.loads#

5 a. Print the event and type , b. Store the input in a variable

#6. Update the 'return' by changing the 'body'

Generative AI with Bedrock : Step by Step Guide

4. Create an REST API from API Gateway

- Resource – demoManufacturing
- Method - POST

5. API Gateway - Method Request

- URL query string parameters – Add input parameter

Generative AI with Bedrock : Step by Step Guide

6. API Gateway - Integration Request

- application/json

```
{ "prompt": "$input.params('prompt')" }
```

7. Deploy the API to a Stage

8. Test using API Gateway Console

9. Sample Text - Link

Prompt for Summarization Task

This is a on-site log report of turbine breakdown.

Issue Log Date – 25-12-2023, Model Number – TB-CL-7882, **Issue** - Cracks appeared in the part MR 7882-9571 next to the rotor hub. The nut connecting the rotor blade to the rotor hub seems to be damaged. The Anemometer readings seem to be within range. The electric braking seems to be unused. No indication of damage to any other component of the turbine except normal wear and tear.

Potential Root Cause – Seems due to reduced tensile strength of the nut connecting the blade to the rotor. Last Maintenance Date – 12-12-2023, **Last Maintenance Issues Recorded** - No known issues recorded and all the parameters were within range.

Summarize the text in 2 lines.

Possible and Actual – Completion from the Model

Possible Response:

Model No-TB-CL-7882, **Issue** - The nut connecting the rotor blade to the rotor hub seems to be damaged,

Potential Root Cause – Seems due to reduced tensile strength of the nut connecting the blade to the rotor.

No known issues recorded in last maintenance.

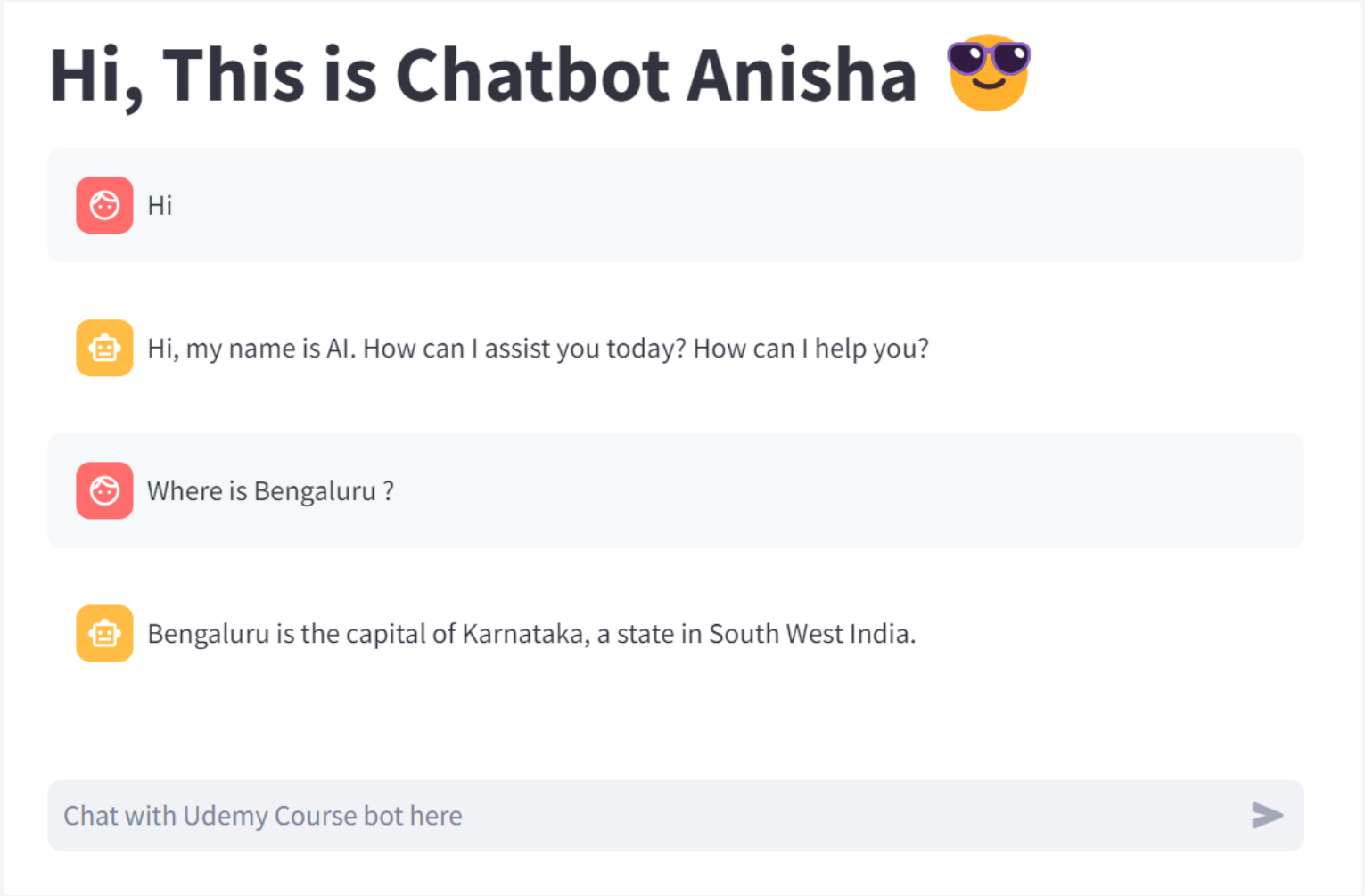
Actual Response :

The turbine experienced cracks in a part and possible damage to the nut connecting the rotor blade, but the a nemometer readings were within range. It is unclear what has caused the issue, but it is assumed to be due to reduced tensile strength.

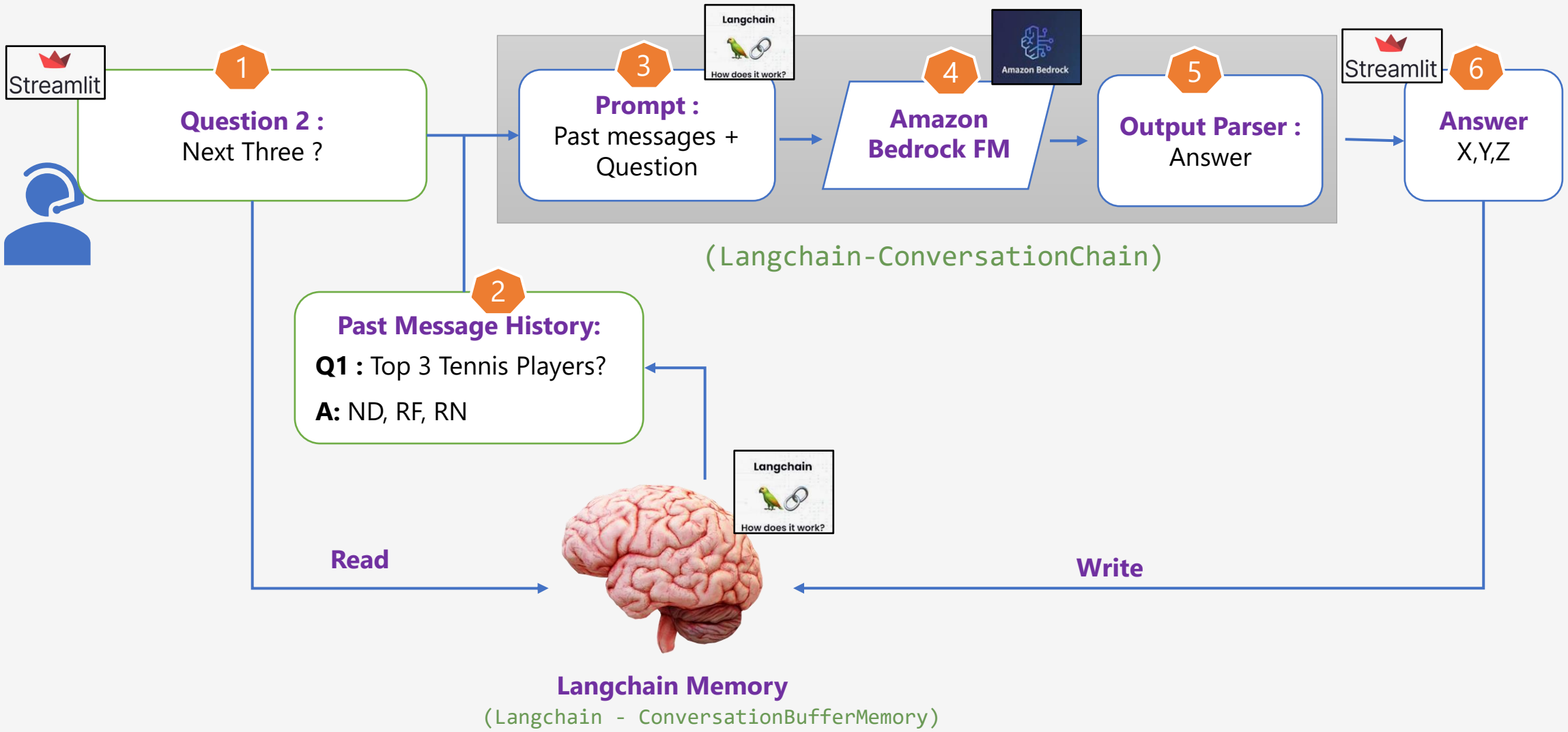
Use Case – Building a Chatbot

Building a Chatbot using Amazon Bedrock : Architecture

Chatbot using Amazon Bedrock – What we will build as part of Use Case



Chatbot Architecture – Amazon Bedrock + Langchain + Streamlit



Pre-Requisites

Building a Chatbot using Amazon Bedrock

Pre-requisites – List of items to be installed

1. Download VS Code
2. Install Python
3. Install AWS CLI
4. Configure IAM Role for VSCode
5. Anaconda Navigator Download
6. Install Boto3
7. Install Langchain
8. Install Streamlit
9. Bedrock
10. AWS Profile

Pre-requisites – How to install 1/2

- Download VS Code - https://code.visualstudio.com/download
- Install Python
- Install AWS CLI - https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html
    - aws -- version
    - Extensions – AWS Toolkit and AWS Boto3
- Configure IAM Role for VSCode - https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-files.html
    - Create IAM Role
    - aws configure
    - Provide credentials
    - Set profile - aws configure set region us-east-1 --profile Udemy_Rahul
    - Profile Setting Details - https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-files.html

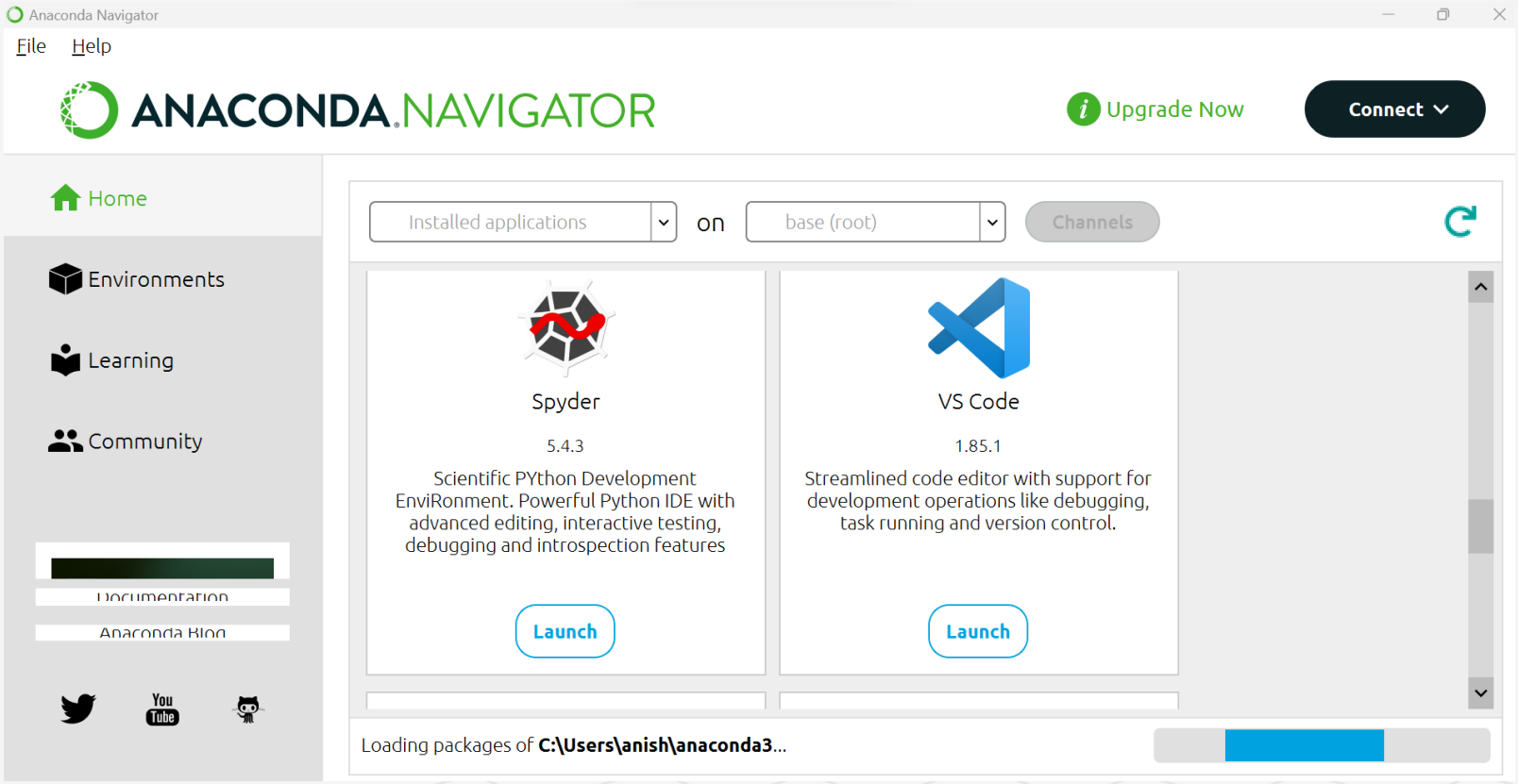
Pre-requisites - How to install 2/2

- Anaconda Navigator Download - https://docs.anaconda.com/free/anaconda/install/windows/
- Open VS Code from Anaconda Navigator – Very Important
- Install boto 3 - pip install boto3
- Install Langchain – pip install langchain
- Install Streamlit – pip install streamlit
- Install Bedrock – pip install bedrock (If you face issues, restart the VSCode editor and/or you laptop)
- Check profile name - aws configure list-profiles

Pre-requisites – Test if everything is working okay

1. Download VS Code
2. Install AWS CLI - aws -- version
3. Configure IAM Role for VSCode – aws s3 ls
4. Anaconda Navigator Download – Dashboard
5. Install Boto3 – pip show boto3
6. Install Langchain – pip show langchain
7. Install Streamlit - streamlit hello
8. AWS Profile - aws configure list-profiles

Anaconda Navigator Dashboard



Coding Steps

Building a Chatbot using Amazon Bedrock

Steps in Building Chatbot – Chatbot Backend Code

- #1 import the OS, Bedrock, ConversationChain, ConversationBufferMemory Langchain Modules
- #2a Write a function for invoking model- client connection with Bedrock with profile, model_id & Inference params- model_kwargs
- `#2b Test out the LLM with Predict method`
- #3 Create a Function for ConversationBufferMemory (llm and max token limit)
- #4 Create a Function for Conversation Chain - Input text + Memory
- #5 Chat response using Predict (Prompt template)

#Links :

- #1 https://python.langchain.com/docs/integrations/llms/bedrock
- #2b Chains - Combine LLMs and Prompts

Steps in Building Chatbot – Chatbot Frontend Code

- How to Run Chatbot locally
    - streamlit run <frontend_filename>.py

Langchain Overview :

Challenges in Large Language Models
&
How Langchain helps overcome them

Challenges in Large Language Models

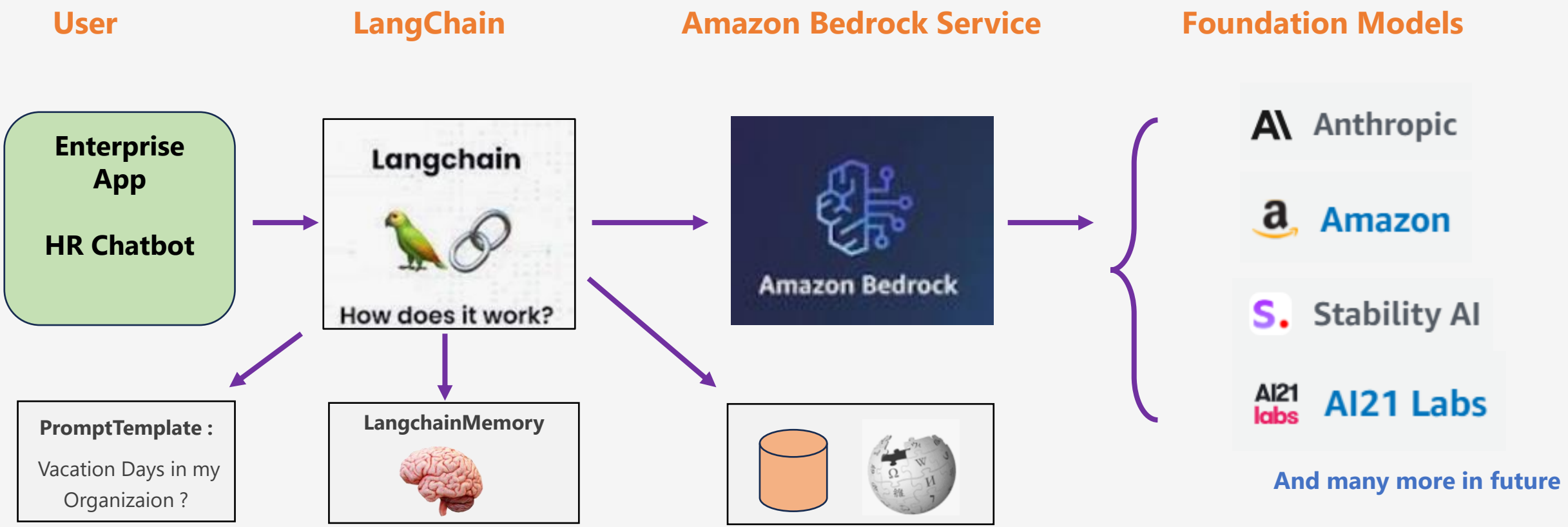
- **Contextual Memory -** Large Language Models are **Stateless**

- **Contextual Knowledge (Domain Knowledge)** - Enhancing the outcome of LLM's with **organization proprietary data**

- **Hallucination –** Due Lack of knowledge of **recent events (post training date)** or no training data in the specific area.

What is LangChain ?

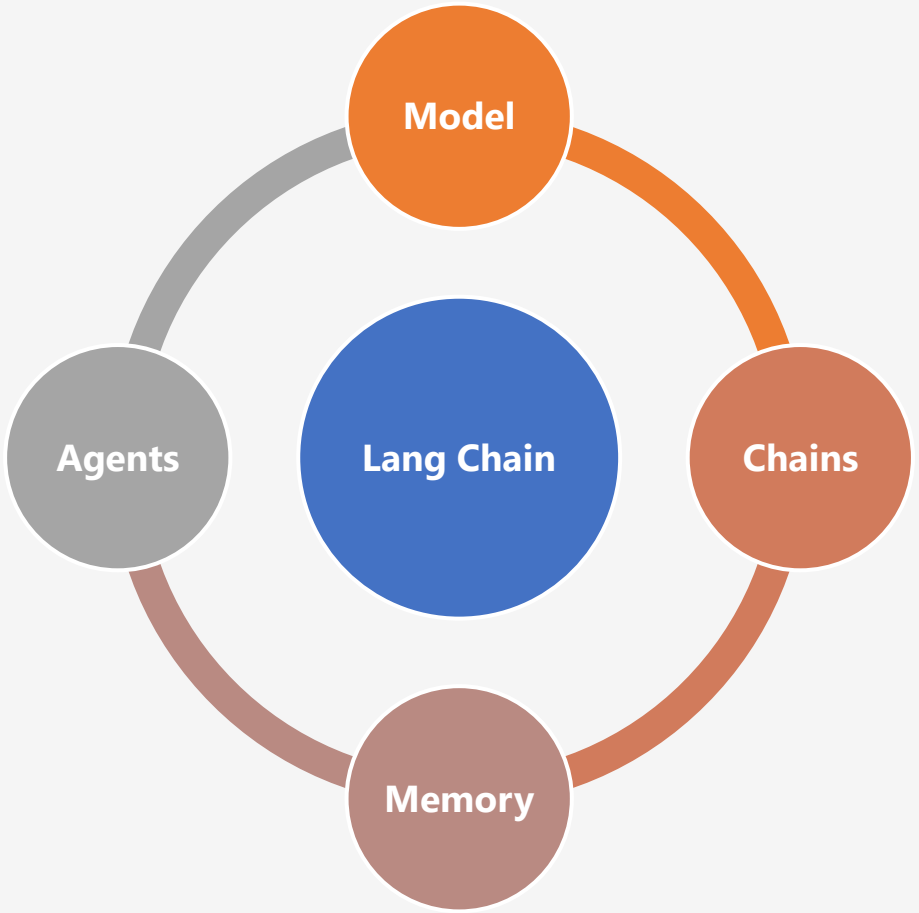
LangChain is a **framework for developing applications** powered by language models.



Lang Chain Components – High Level Overview

Standard interface to connect different LLMs such as Claude, Titan

In agents, a **LLM is used as a reasoning engine** to determine actions & order



Combine **multiple components together** to create a single, coherent application. Example – Prompt Template + LLMs + External Sources

Provides ability to **store information about past interaction** - Chatbots

Streamlit – Overview

Use Case : Retrieval Augmented Generation (RAG)

Employee HR Q & A Application with RAG

Amazon Bedrock - Claude FM

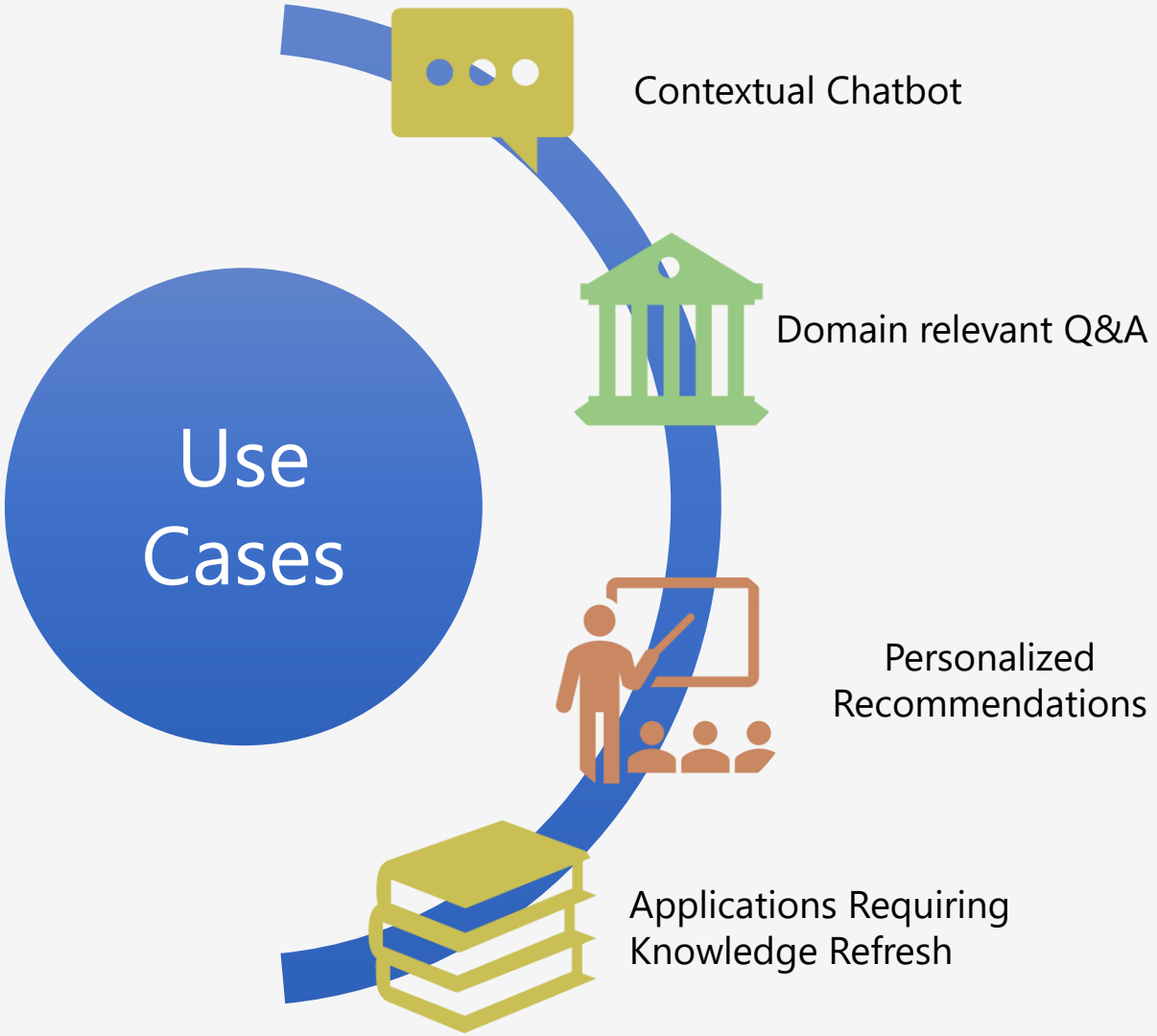
+

Langchain

+

HR Policy Document (PDF)

Key Use Cases for Retrieval Augmented Generation in the Industry



Basic Introduction

to

Vectors, Embeddings and Vector Search

1. Why do we need Vectors ?

Structured Data

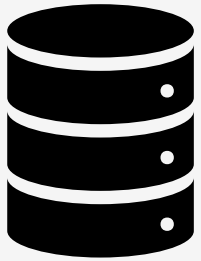


Image Search or Unstructured Data

???

Key Terms

1. **Vectors**
2. Embedding Models
3. Data Chunking
4. Vector Store
5. Vector Search
6. Cosine Similarity
7. K-Nearest Neighbor Search

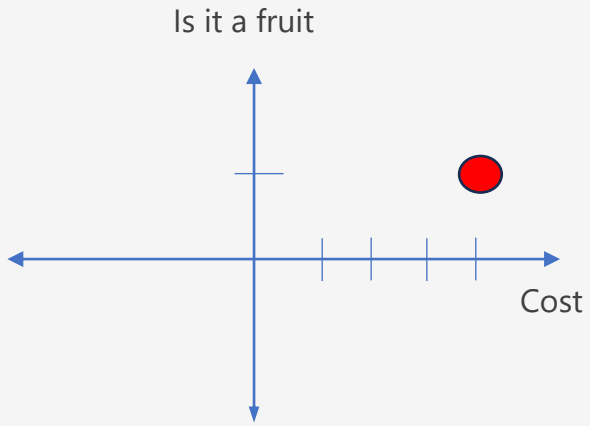
1. What are Vectors ?

Vectors are **mathematical representation of words, sentences** and documents

Vectors are represented by **List of numbers** or sequence of numbers

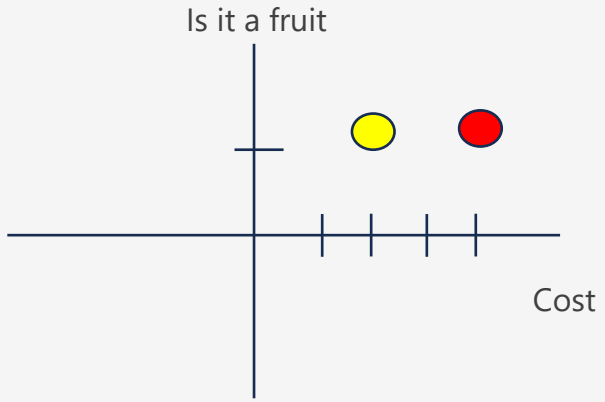
Is it a fruit	Cost
1	$4

[1 , 4]



Is it a fruit	Cost
1	$2

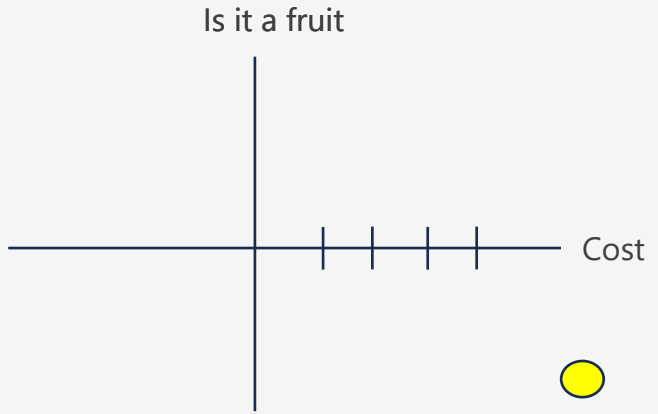
[1 , 2]



1. What are Vectors ?

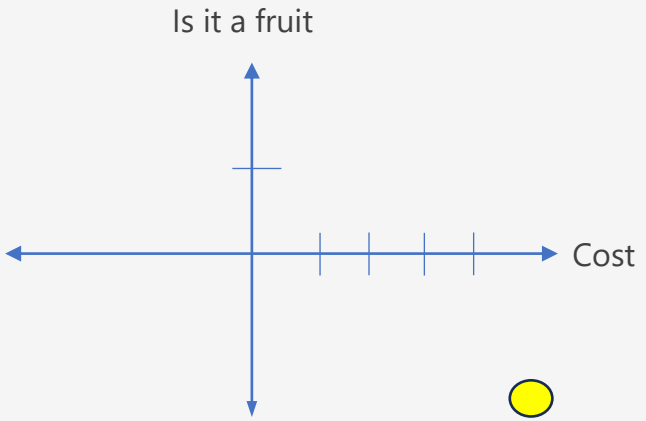
Is it a fruit	Price
-1	$50

[-1 , 50]



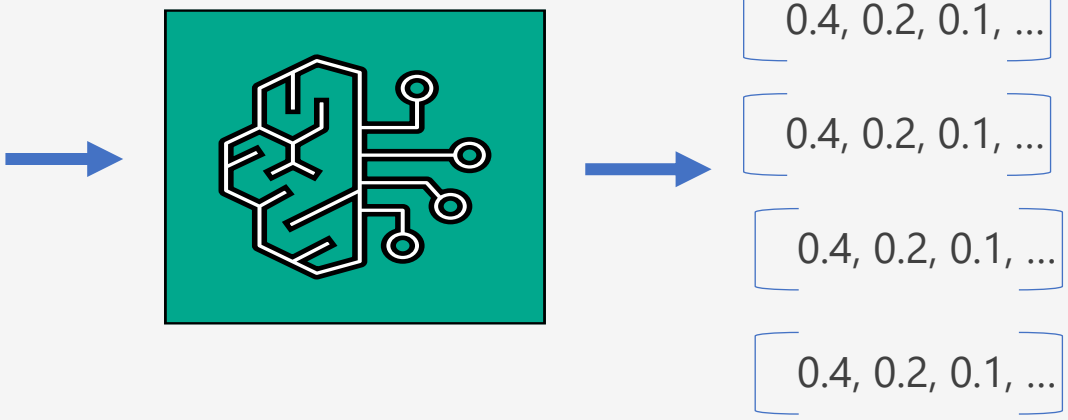
Is it a fruit	Price
-1	$40

[-1 , 40]



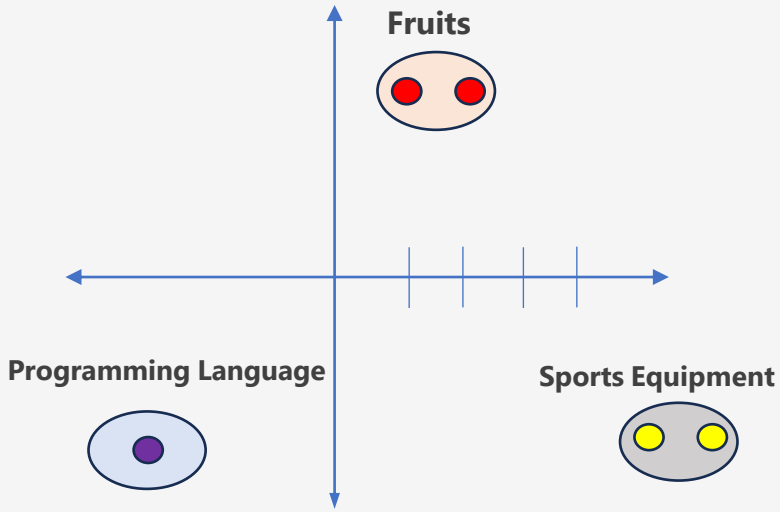
2. What are Embedding Models ?

- Apple
- Banana
- Grapes
- Football
- Soccer
- GO
- Tennis



Amazon Titan Embeddings

1536 Dimensions



Word Embeddings
Ex : great

Sentence Embeddings
Ex. : Price of apple ?

Doc Embeddings

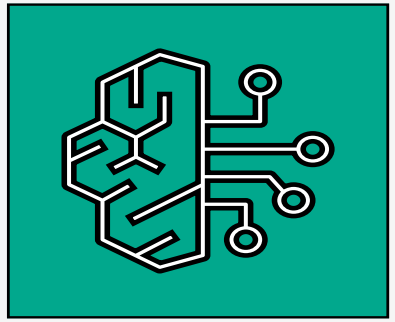
Key Terms

1. Vectors
2. **Embedding Models**
3. Data Chunking
4. Vector Store
5. Vector Search
6. Cosine Similarity
7. KNN

Embedding Models

Output from Embedding Models

Fruit



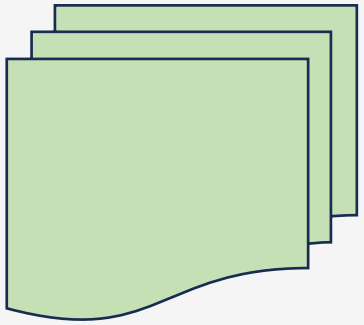
Amazon Titan Embeddings

```
[
    0.022321066,   -0.027544279,  -0.006137953,  0.0024092742, -0.0003515296,
   -0.007889225,    -0.01853968,  -0.037405808,  -0.004294867,  -0.012602357,
    0.009385457,     0.02421176,  -0.004383281,  -0.011901848,  -0.020702416,
     0.00431187,    0.055986296,  0.0040738326,   0.021300908,   -0.03713377,
   -0.004944368,    0.012269106,    0.02614326,  -0.019777471,  0.0017206672,
    0.014867109,    0.004597514,  -0.021913003, -0.0042336574,   0.013180447,
    0.032699477,   -0.027245032, -0.0112965545,  -0.009358253,  -0.004951169,
   -0.015030335,   -0.007501565,  -0.018906936,  0.0074675595, -0.0067568496,
   0.0011952856,   0.0035501514,   0.012316713,   0.016132105,  -0.014839904,
  -0.0044002836,    0.010405616, -0.0066276295,  -0.015751246,  -0.007141109,
     0.02729944, -0.00031348618,  -0.015669633,  -0.015656032,   0.015696838,
    0.013554505,   -0.001635654,   0.006069943,   0.009167824,   -0.01917898,
   -0.006059741,    0.013969369,   -0.03006067,   0.007691995,  0.0005028531,
   -0.005468049,   -0.007263528,   0.009970348,  -0.016254524,  0.0041248407,
    0.01964145,     0.037324198, -0.0011340762, -0.0013296065,   0.010392014,
   -0.01268397,   -0.013037625,  -0.006811258, -0.0045601083,  0.0028955496,
    0.027435461,   -0.024320576,  -0.006161757,   0.014350229,   0.009392259,
   -0.006328383,   -0.025490358,   0.023273215,  0.0083925035,  -0.003934411,
    0.023722084,  -0.0072499258, -0.0053626327,   0.017818768,   -0.03327077,
    0.019097365,   0.0022086431,   0.046709653,  -0.016825814,  -0.032182597,
  ... 1436 more items
]
```

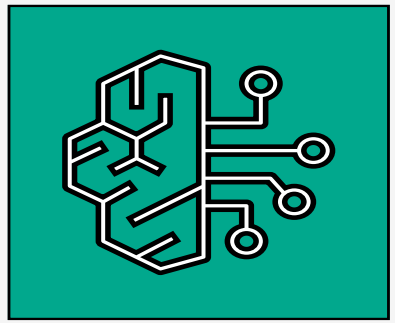
1536 Dimensions

3. Text Splitting or Data Chunking

Data Source - PDF



- Split by Character
- Split by Token
- Split by Code



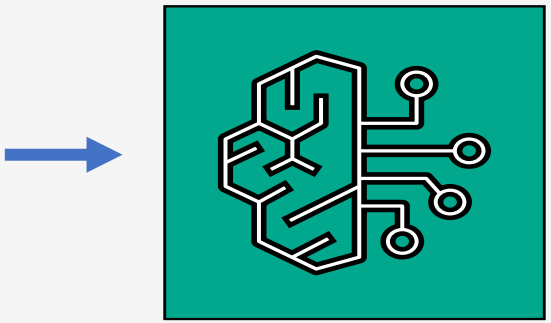
Amazon Titan Embeddings

Key Terms

1. Vectors
2. Embedding Models
3. **Data Chunking**
4. Vector Store
5. Vector Search
6. Cosine Similarity
7. K-Nearest Neighbor Search

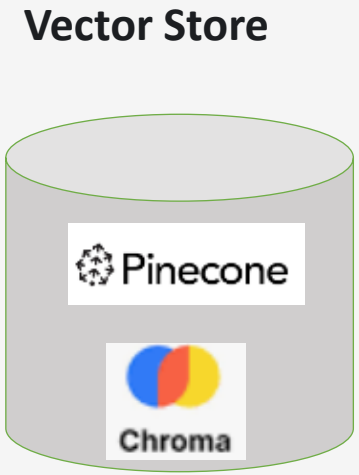
4. Vector Store?

- Apple
- Banana
- Grapes
- Football
- Soccer
- GO
- Tennis



Amazon Titan Embeddings

1536 Dimensions

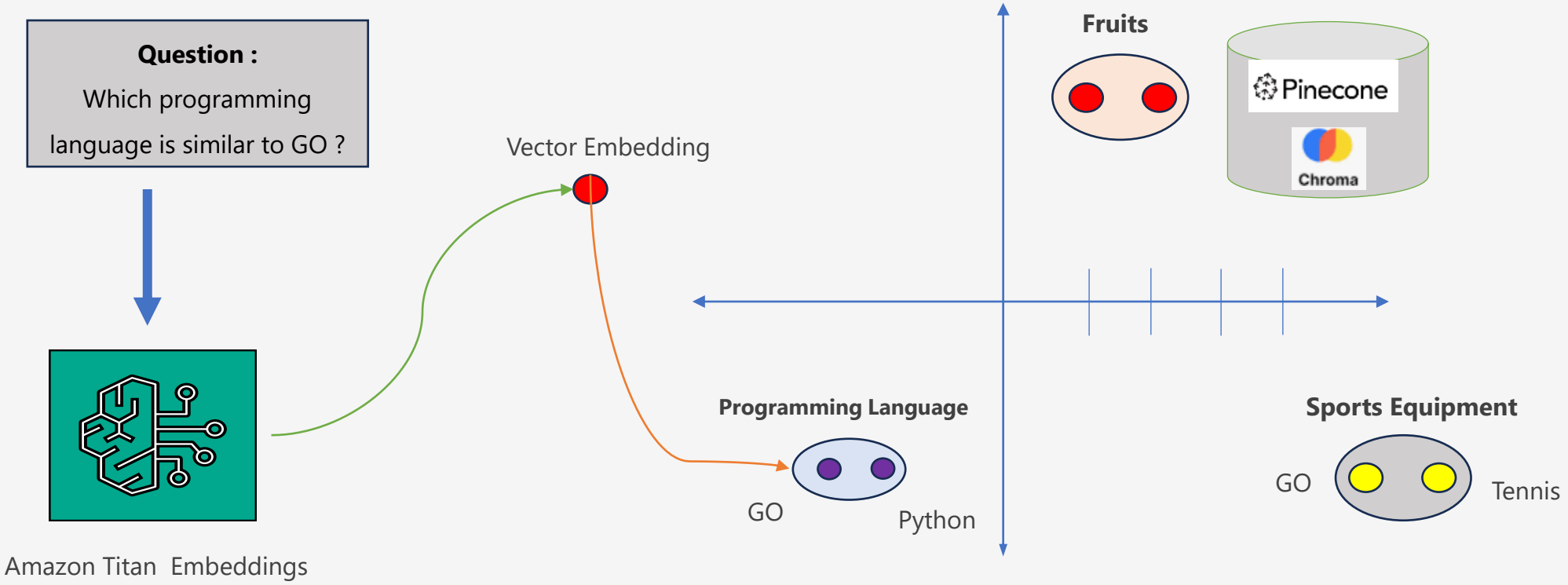


- ✓ FAISS
- ✓ Pinecone
- ✓ Chroma DB

Key Terms

1. Vectors
2. Embedding Models
3. Data Chunking
4. **Vector Store**
5. Vector Search
6. Cosine Similarity
7. K-Nearest Neighbor Search

5. Vector Search

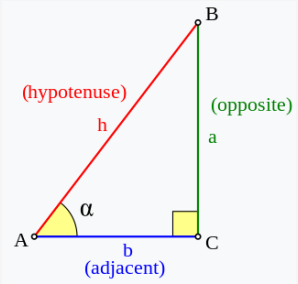


Key Terms

1. Vectors
2. Embedding Models
3. Data Chunking
4. Vector Store
5. **Vector Search**
6. Cosine Similarity
7. K-Nearest Neighbor Search

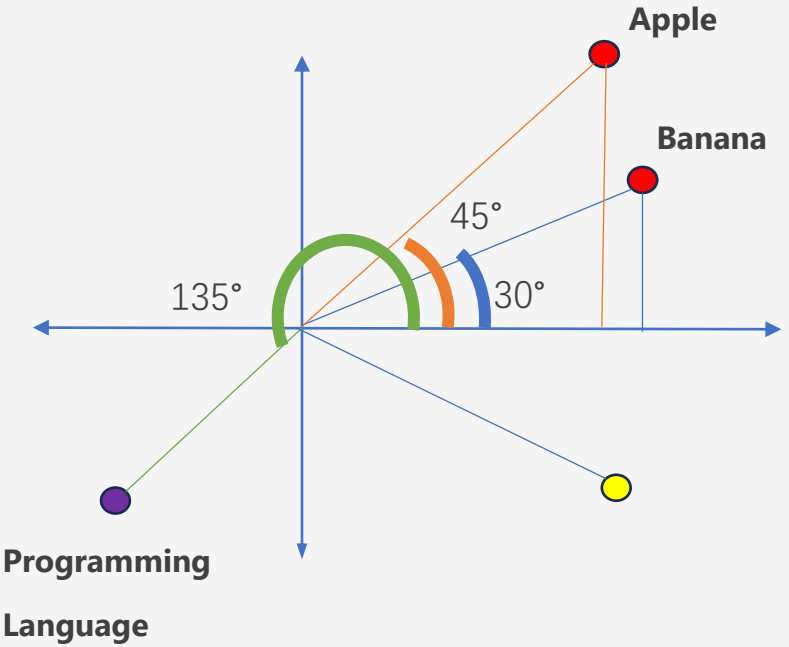
6. Similarity Search – Cosine Similarity

1. **Cosine θ = adjacent/hypotenuse**



2. **Similarity Search using Cosine Distance:**

- Cosine distance metric is used to find **similarities between different vectors**
- In **cosine metric** we measure the **degree of angle between two vectors**
- **Cos 0 = 1, Cos 90 = 0 and Cos 180 = -1**
- **Cosine value 1** is for **vectors pointing in the same direction --- > Similar**
- At Cosine Value 0 ---- > Vectors unrelated (some similarity found).
- Value -1 for vectors pointing in opposite directions(No similarity).



7. Cosine Similarity and K- Nearest Neighbor Algorithm or ANN

KNN is a ***supervised machine learning algorithm*** used **classification and regression** tasks.

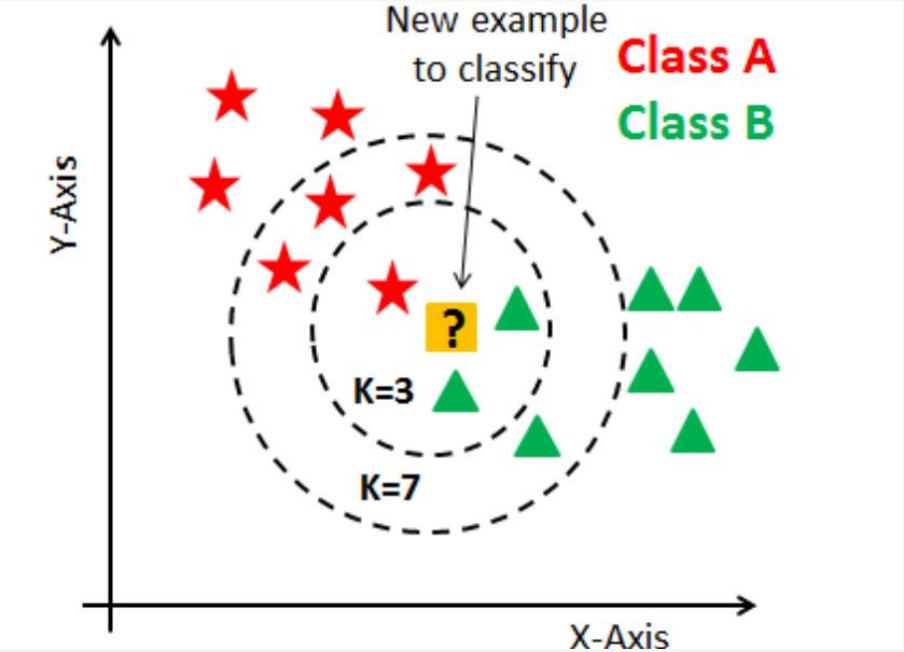


Image Source : Internet

Key Terms

1. Vectors
2. Embedding Models
3. Data Chunking
4. Vector Store
5. Vector Search
6. **Cosine Similarity**
7. **KNN**

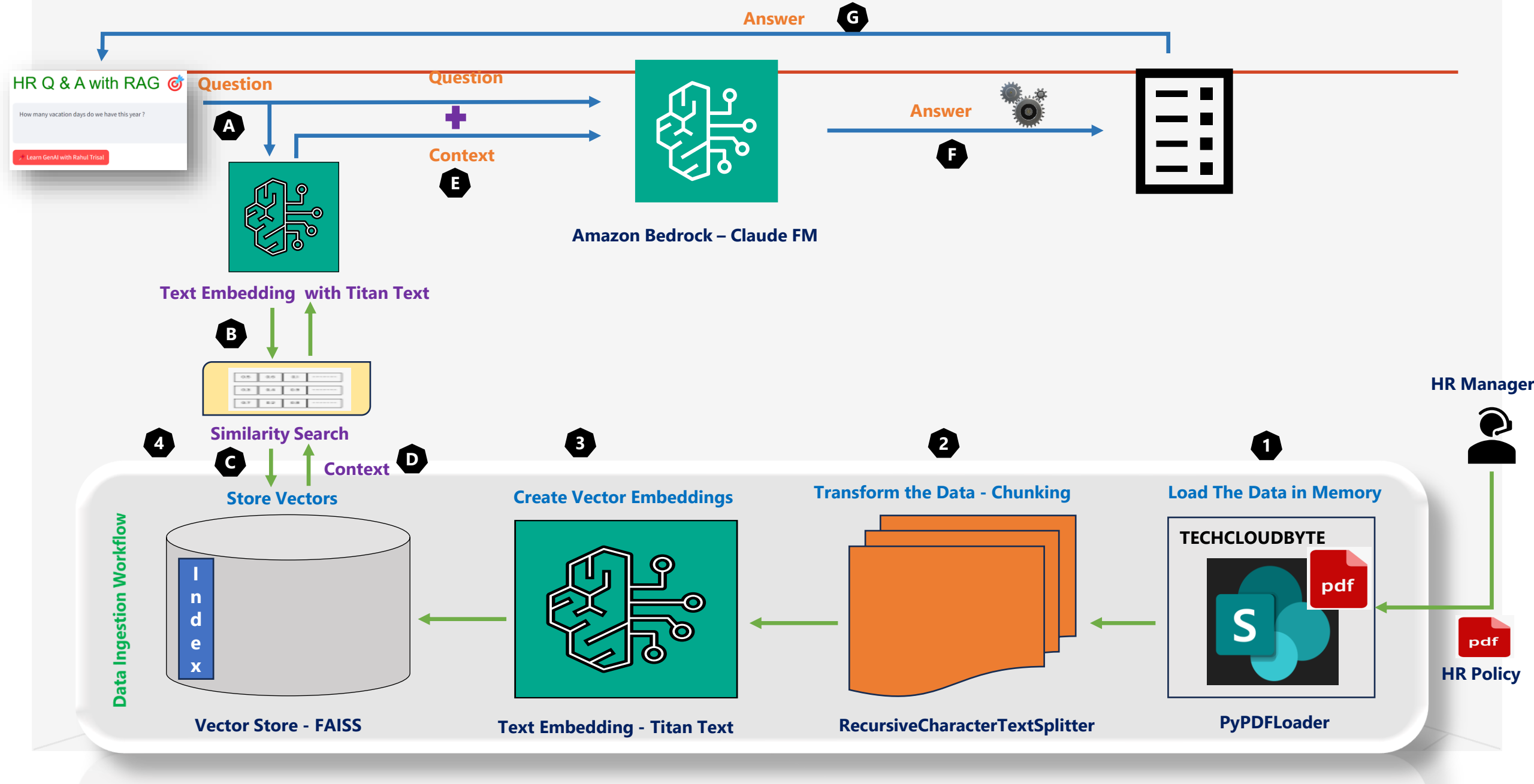
ANN - Approximate Nearest Neighbor

Use Case : Retrieval Augmented Generation (RAG)

Employee HR Q & A Application with RAG

Architecture Overview

HR Question & Answer Application with Retrieval Augmented Generation - Architecture



Use Case : Retrieval Augmented Generation (RAG)

Employee HR Q & A Application with RAG

Pre- Requisites - Installation

Pre-requisites – List of items to be installed

1. **pip3 install flask-sqlalchemy**
2. **pip install pypdf**
3. **pip install faiss-gpu** (for CUDA supported GPU)

   or

4. **pip install faiss-cpu** (depending on Python version).

Pre-Requisites

For installation of other items for this use case, if not already installed, refer to below Section and Lecture (one shown in video as well). If already installed, skip next few slides.

Section : Building a Chatbot with Llama 2, Langchain & Streamlit

&

Lecture : Chatbot - Environment Setup before coding

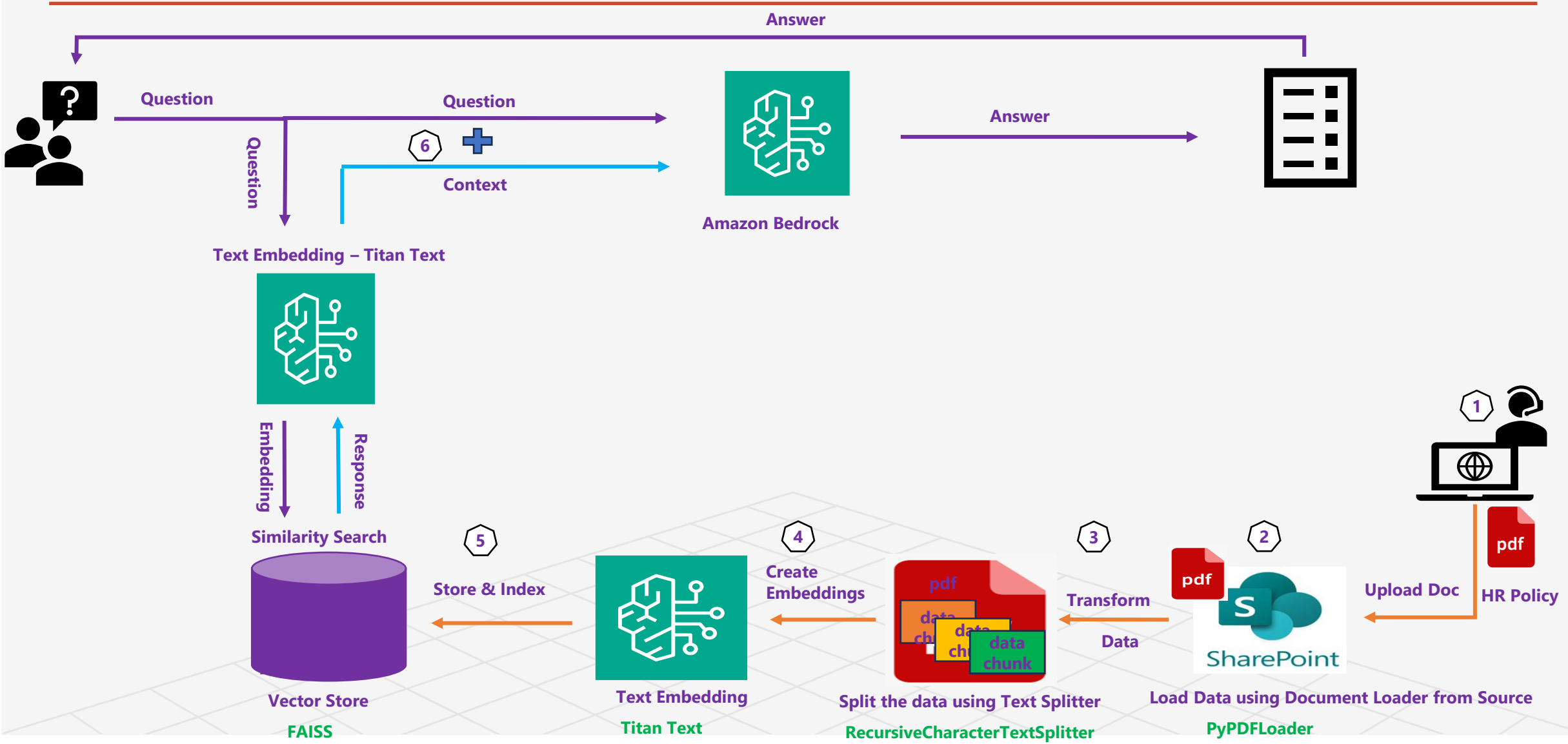
Use Case : Employee HR Q & A App with RAG

Coding Steps

Steps in Building HR Q&A Solution with RAG – Backend Code

- #1. Import OS, Document Loader, Text Splitter, Bedrock Embeddings, Vector DB, VectorStoreIndex, Bedrock-LLM
- #2. Define the data source and load data with PDFLoader(https://www.upl-ltd.com/images/people/downloads/Leave-Policy-India.pdf)
- #3. Split the Text based on Character, Tokens etc. - Recursively split by character - ["\n\n", "\n", " ", ""]
- #4. Create Embeddings -- Client connection
- #5à Create Vector DB, Store Embeddings and Index for Search - VectorstoreIndexCreator
- #5b Create index for HR Report
- #5c. Wrap within a function
- #6a. Write a function to connect to Bedrock Foundation Model
- #6b. Write a function which searches the user prompt, searches the best match from Vector DB and sends both to LLM.
- # Index creation --> https://api.python.langchain.com/en/latest/indexes/langchain.indexes.vectorstore.VectorstoreIndexCreator.html

HR Question & Answer App with RAG - Solution Architecture



Steps in Building HR Q&A Solution with RAG – Frontend Code

- Modify the Streamlit frontend code where ### highlighted
- Run Chatbot locally
  - streamlit run <frontend_filename>.py

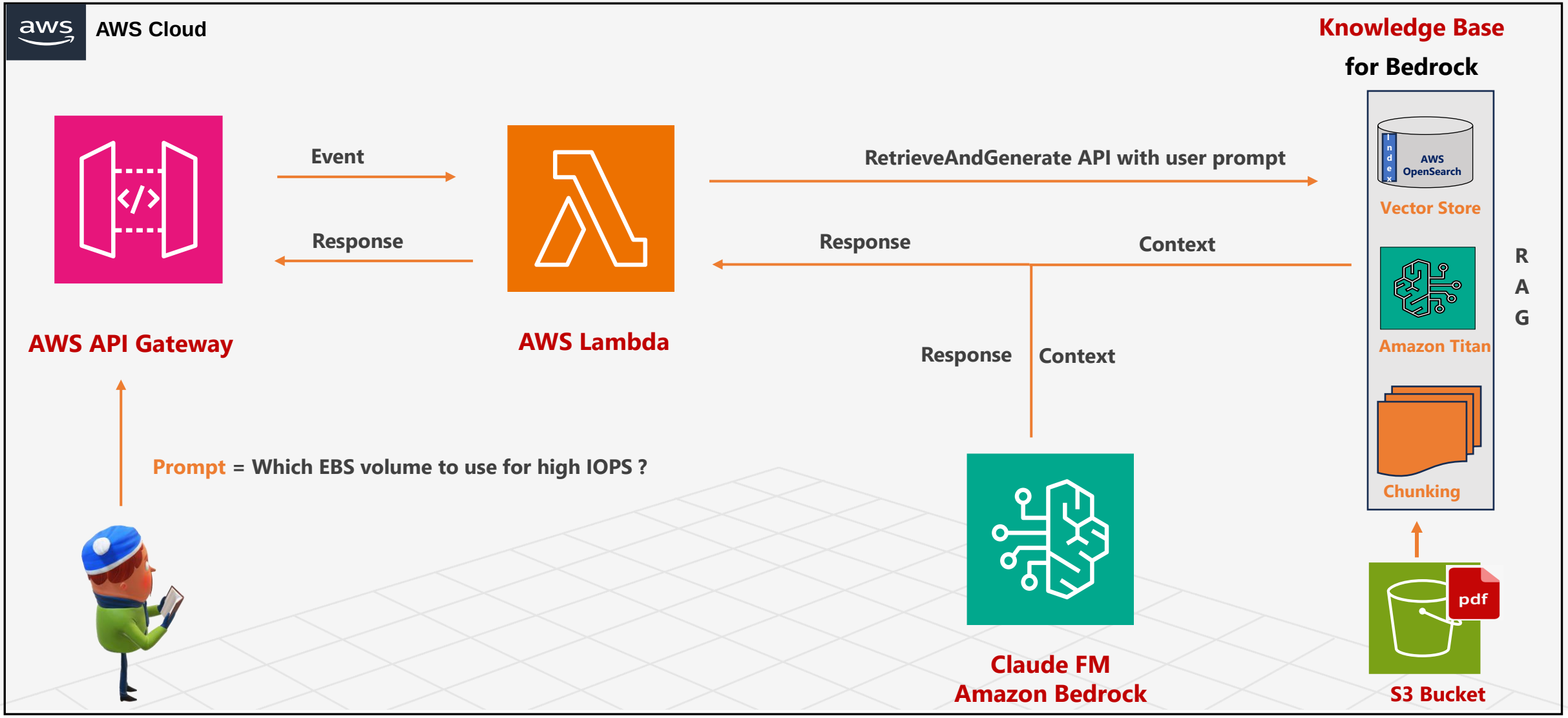
Data Source URL

```
https://www.upl-ltd.com/images/people/downloads/Leave-Policy-India.pdf'
```

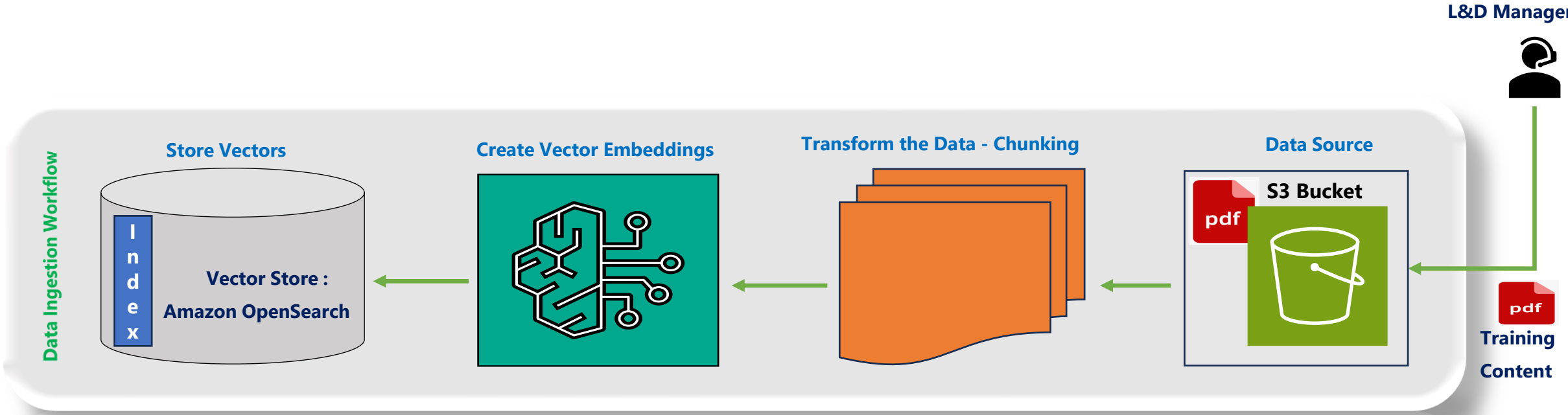
Use Case :

Serverless e-Learning App using Knowledge Base for Bedrock

Serverless e-Learning App using Knowledge Base for Bedrock



E-Learning Solution using RAG - Amazon Bedrock Knowledge Base + LLM



Data Ingestion Workflow - Options

1. **Data Sources**

   - ❖ S3

2. **Chunking**

   - ❖ Default
   - ❖ Fixed Size

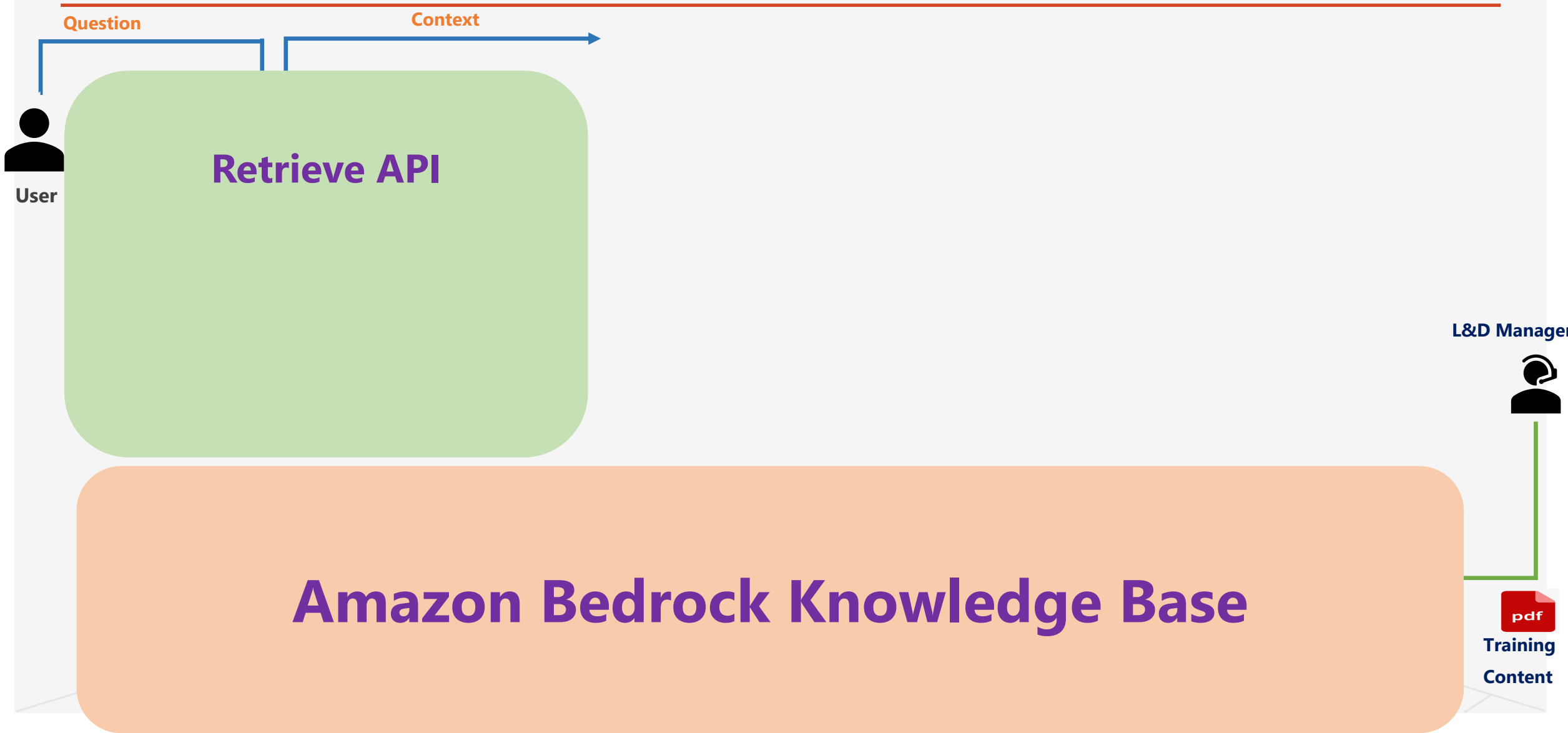
3. **Embedding Model**

   - ❖ Cohere
   - ❖ Amazon Titan

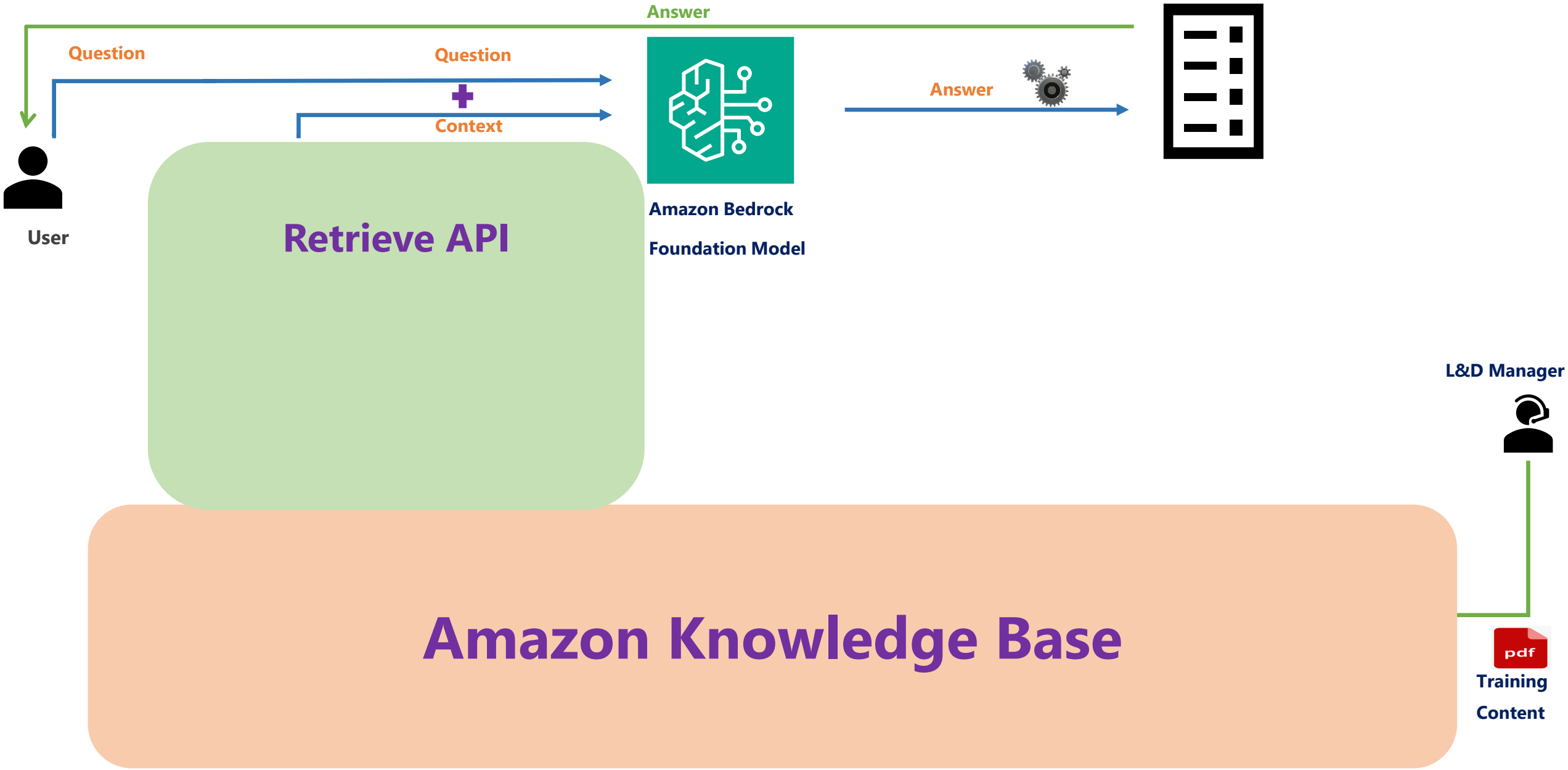
4. Vector DB

- ❖ Amazon OpenSearch Serverless vector store
- ❖ Amazon Aurora
- ❖ Pinecone
- ❖ Redis Enterprise Cloud

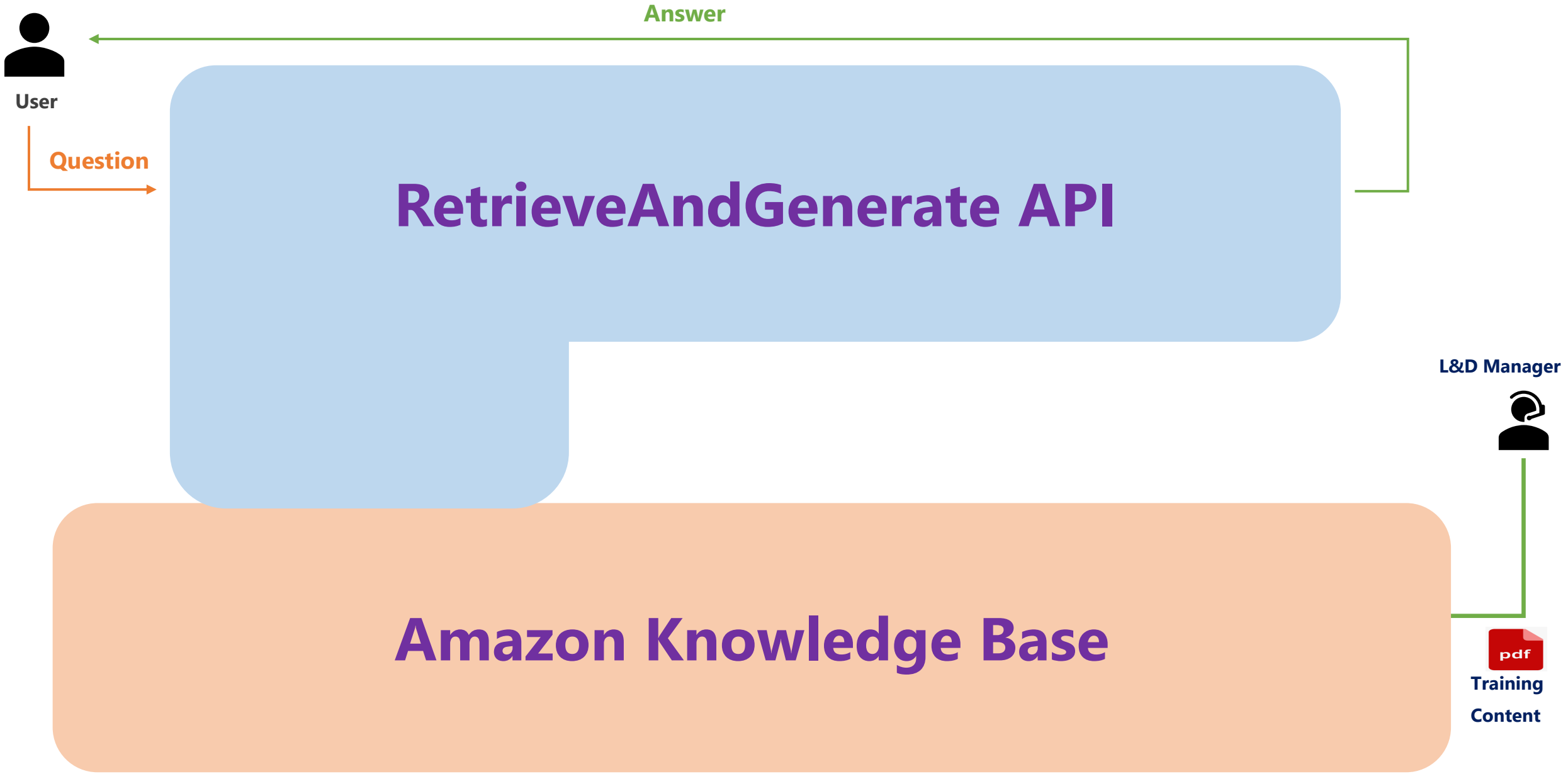
L&D App with Amazon Knowledge Base (Retrieval Augmented Generation) - Architecture



L&D App with Amazon Knowledge Base (Retrieval Augmented Generation) - Architecture



L&D App with Amazon Knowledge Base (Retrieval Augmented Generation) - Architecture



Two API's for Retrieval – Retrieve API and RetrieveAndGenerate API

1. Retrieve API :

- Queries a Knowledge Base
- **Retrieves** information from it.

https://docs.aws.amazon.com/bedrock/latest/APIReference/API_agent-runtime_Retrieve.html

2. RetrieveAndGenerate API

- Queries a knowledge base
- **Generates responses** based on the **retrieved results**.
- Response **cites up to five sources** but only selects the ones that are **relevant to the query**.

https://docs.aws.amazon.com/bedrock/latest/APIReference/API_agent-runtime_RetrieveAndGenerate.html

Broad Implementation Steps

1. **Data Source** –
    - Create the **S3 Bucket**
    - **Upload** the relevant **files** in the S3 Bucket
2. Create Amazon Bedrock **Knowledge Base --- > Data Source, Chunking, Embedding Model, Vector Store**
3. Create a **AWS Lambda Function**
    - Create IAM Role
    - Increase timeout limit
    - Write the **RetrieveAndGenerate API** to access data from the Knowledge Base – Link
    - Model ARN - 'modelArn': **'arn:aws:bedrock:us-east-1::foundation-model/anthropic.claude-instant-v1'**
4. Optional Step incase of error - Update boto3 version incase of error using Lambda Layers - Link
5. Create **REST API**
6. **Integrate** AWS Lambda function with API Gateway

Broad Implementation Steps

Create REST API using AWS API Gateway

- Create Resource
- Create Method – GET

API Gateway - Method Request

- URL query string parameters – Add input parameter
- Enable - Request validator
- Validate body, query string parameters, and headers

Broad Implementation Steps

- **API Gateway - Integration Request**

application/json

{

"prompt":"$input.params('prompt')"

}

- **Deploy the API to a Stage’(Stage=Dev or any other)**
- **Test using API Gateway Console and Postman**

Use Case :

Build Retail Banking Agent

using

Bedrock Agents & Knowledge Bases

Amazon Bedrock Agents – **Retail Banking Agent**

John

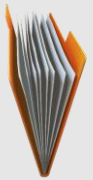
Check account activation status after 3 days from Bank Website

Thomas

Manju

Bank of Chicago

AccountID = 5555



AccountID = 6666

AccountID = 7777

Amazon Bedrock Agents – **Retail Banking Agent**

www.bankofchicago.com

Manju

AccountID = 7777

Test | TestAlias: Working draft | Info

Hi

What is your account ID?

Hide trace

DynamoDB Table

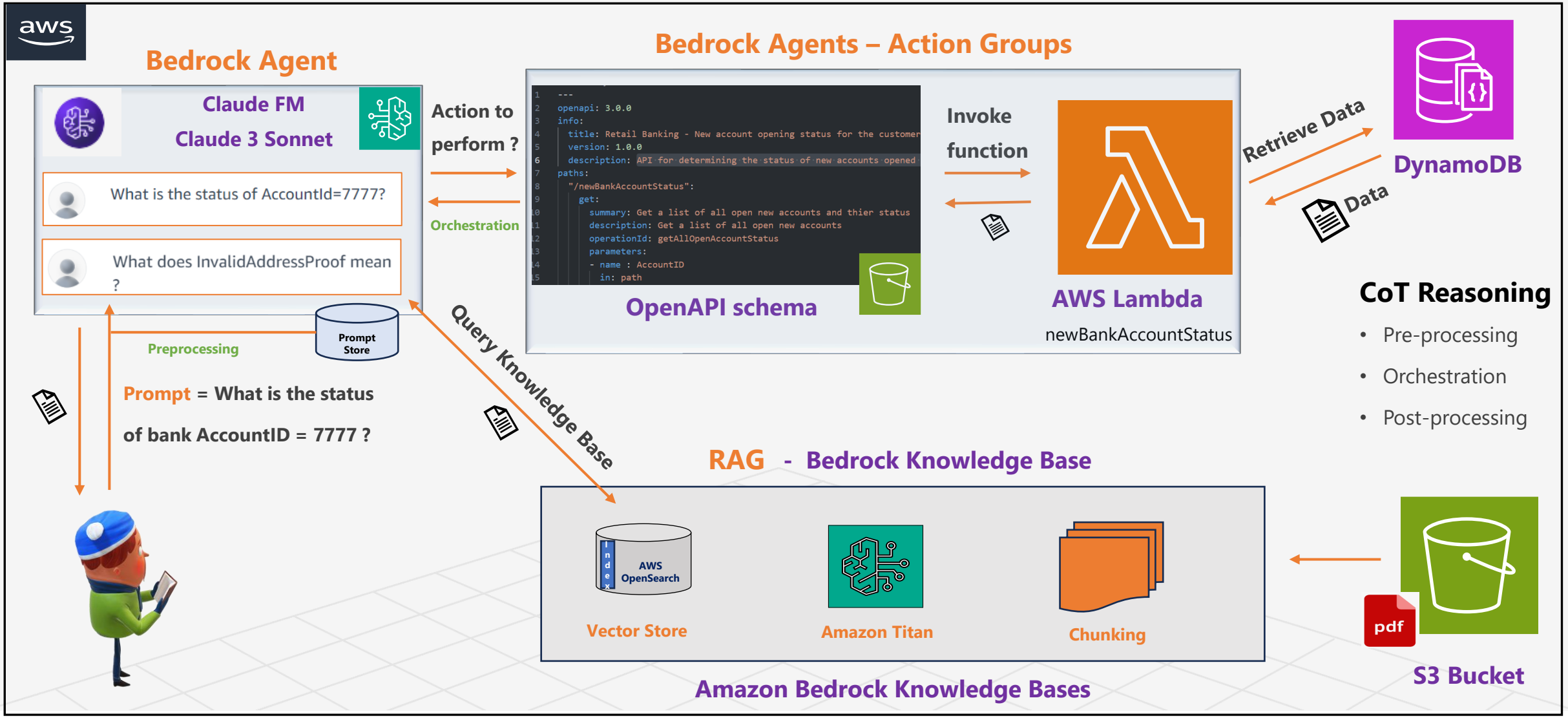
AccountID	AccounName	Account Status	Reason
5555	John	Active	Active
6666	Thomas	Pending	InvalidIdentification
7777	Manju	Pending	InvalidAddressProof

Customer Data - DynamoDB Table

TableName - customerAccountStatus

AccountID	AccounName	AccountStatus	Reason
5555	John	Active	Active
6666	Thomas	Pending	InvalidIdentification
7777	Manju	Pending	InvalidAddressProof

Retail Banking Agent - Amazon Bedrock Architecture



Steps to create Amazon Bedrock Agent for Bank of Chicago 1/2

1. Create a DynamoDB Table with the customer account details - TableName - customerAccountStatus
2. Create a AWS Lambda function to retrieve the data from DynamoDB Table – Lambda Function - newBankAccountStatus
3. Define OpenAPI schemas for Bedrock Agent Action Group and Upload on S3 Bucket- Link
    - S3 bucket details for OpenAPI Schema file - accountstatusopenapi
4. Create the Bedrock Agent from console - bankofchicago-agent
5. Update the Lambda function Permissions with Agent Invocation Details
6. Update the Lambda Function Code to allow Bedrock Agent to send request/response in a defined format - Link
7. Create Bedrock Knowledge Base (RAG solution) & update the details in the Action Group – S3 Bucket -bankingagentknowledgebase
8. Test the Agent

Step 2 – AWS Lambda Function creation Steps

#1 Import boto3 and create client connection with DynamoDB - https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/dynamodb.html

#2 Print event value and store the event details in a variable

#3 Create a request syntax to retrieve data from the DynamoDB Table using GET Item method - https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/dynamodb/table/get_item.html

#4 Store and print the response

#5 Format the response as per the requirement of Bedrock Agent Action Group - https://docs.aws.amazon.com/bedrock/latest/userguide/agents-lambda.html

#6 Print the final response

Step 3 – Define OpenAPI schemas for your agent's action groups

1. Refer to the AWS sample OpenAI schemas - https://docs.aws.amazon.com/bedrock/latest/userguide/agents-api-schema.html
2. Convert to YAML
3. Modify the Schema as per the requirement of the use case
4. Create an S3 bucket and store the Open API Schema in the S3 bucket

Step 4 : Update the Lambda function Permissions for Agent Invocation

Permission for Agent to invoke Lambda function

- bedrock-agent
- bedrock.amazonaws.com
- ARN
- Lambda:InvokeFunction

Agent Instructions

You are an banking assistant in a Retail Bank. You are friendly and polite. You help resolve customer queries by providing bank customers status on their new bank accounts.

GenAI Project Lifecycle

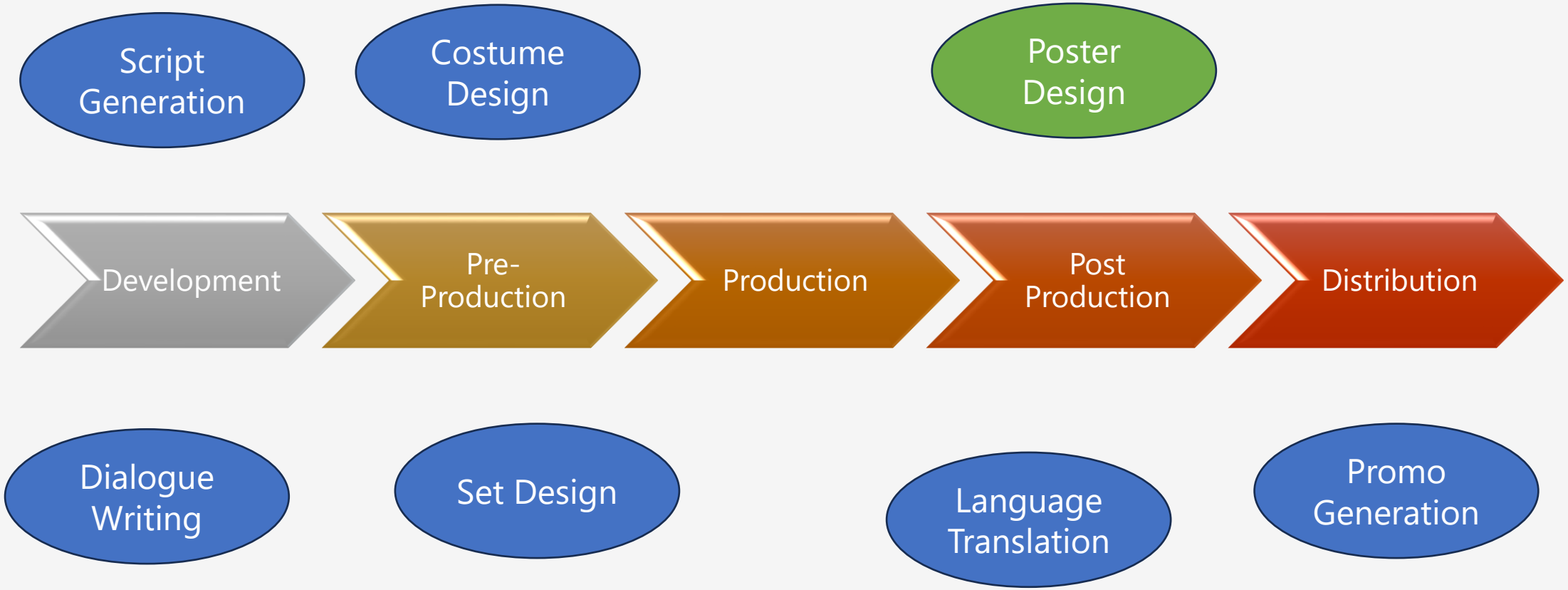
Generative AI – Implementation Lifecycle

1. **Scope of Program**
   - Identify the Use Case
   - Define key objectives & outcomes
2. **Model Selection**
   - FM model selection criteria
   - Evaluate and select the Foundation Model
3. **Adapt and Align Foundation Model**
   - Prompt Engineering
   - RAG
   - Fine Tuning
4. **Deployment & Integration**
   - Deploy the Model
   - Integrate with existing application landscape

Generative AI – Implementation Lifecycle

1. **Scope of Program**
   - Identify the Use Case
   - Define key objectives & outcomes
2. **Model Selection**
   - FM model selection criteria
   - Evaluate and select the Foundation Model
3. **Adapt and Align Foundation Model**
   - Prompt Engineering
   - RAG
   - Fine Tuning
4. **Deployment & Integration**
   - Deploy the Model
   - Integrate with existing application landscape

Generative AI – Media and Entertainment Industry(Film Making)

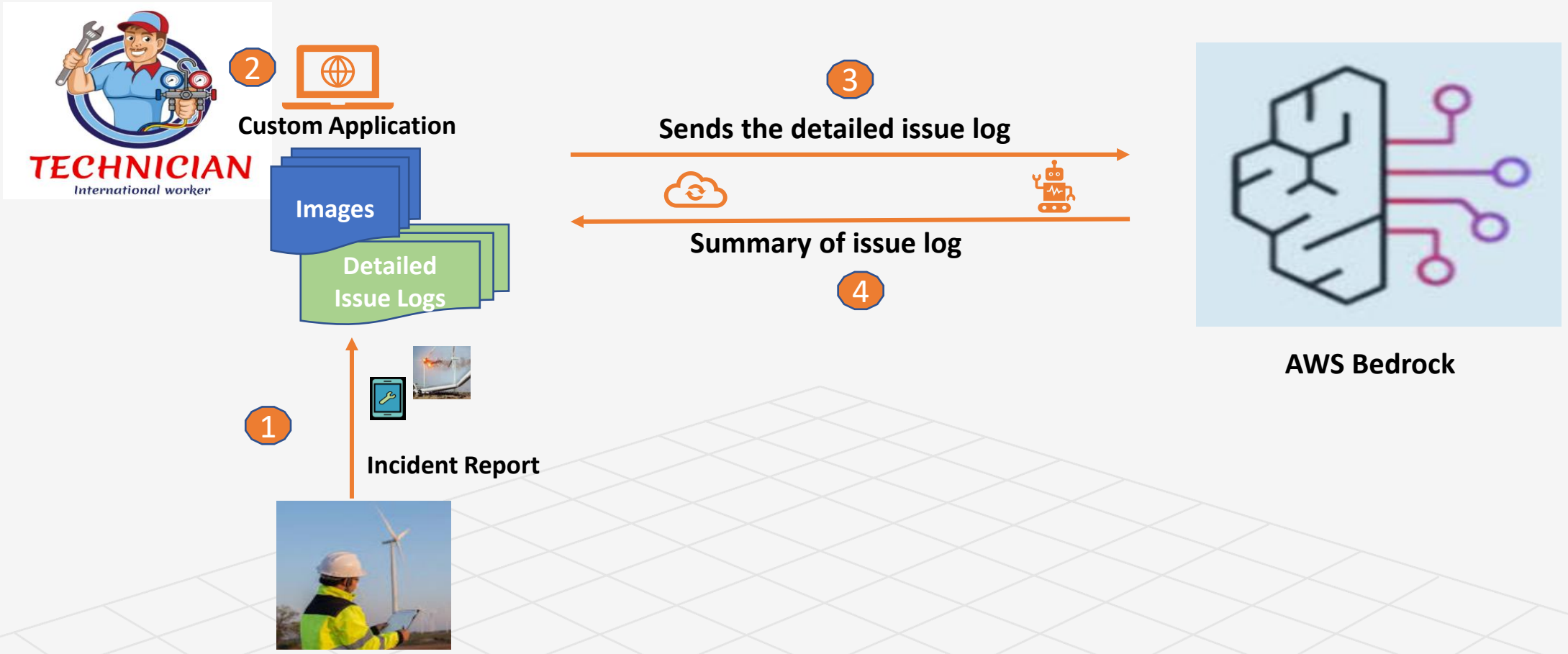


Stages in Movie making and potential role of Generative AI

Generative AI – Manufacturing Industry Use Case

Use Case :

Text Summarization using **AWS Bedrock** for **faster issue resolution to improve productivity** of technicians.



Generative AI – Use Cases

Source : AWS

Generative AI – Use Cases

- AWS Generative AI Use Case Based on Industry : Link
- AWS Generative AI Case Studies – Link

GenAI Project Lifecycle – Foundation Model Selection

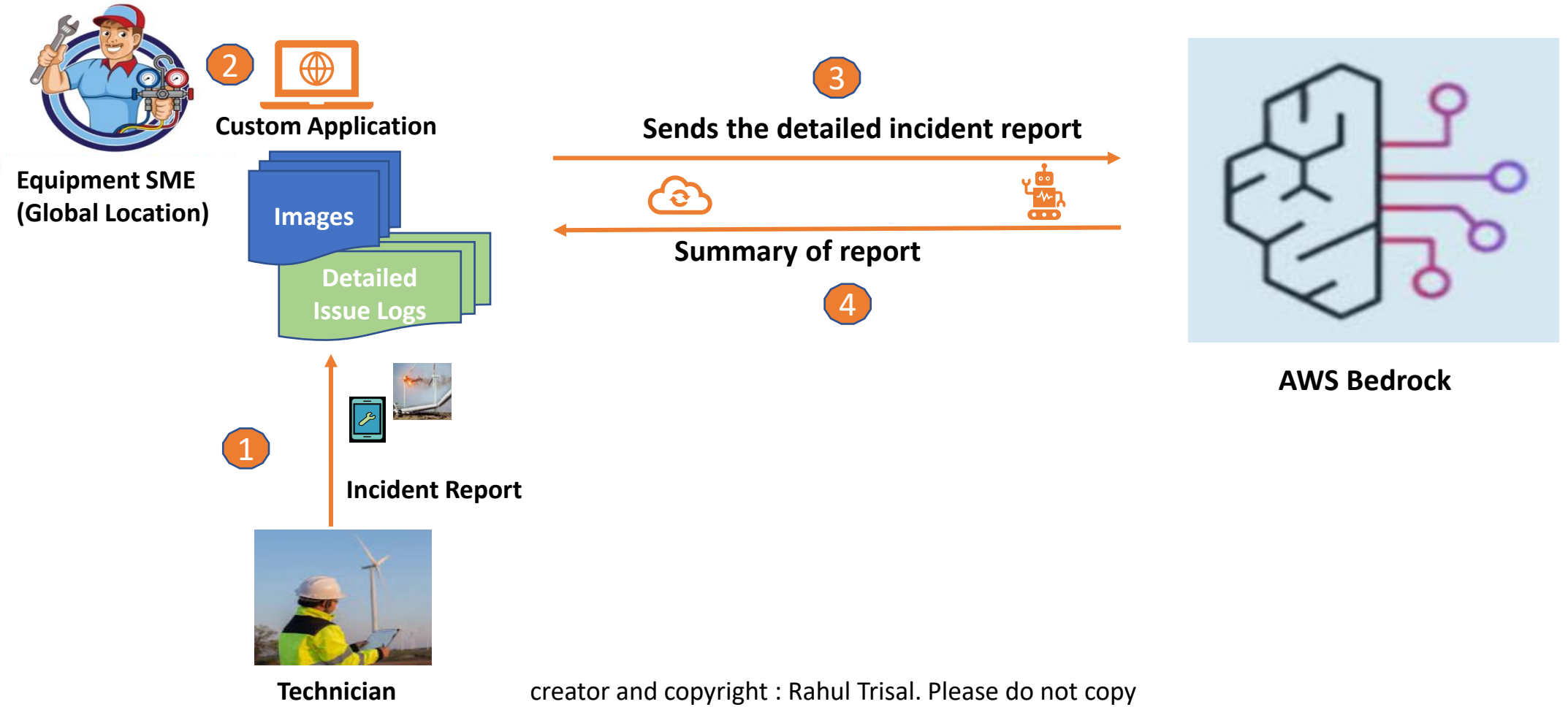
Generative AI Application Lifecycle

1	2	3	4
Scope of Program	Model Selection	Adapt and Align Foundation Model	Integration
• Identify the Use Case • Define key objectives & outcomes	• Define model selection criteria • Evaluate and select the Foundation Model	• Prompt Engineering • Fine Tuning • RAG	• Deploy the Model • Integrate with existing application landscape

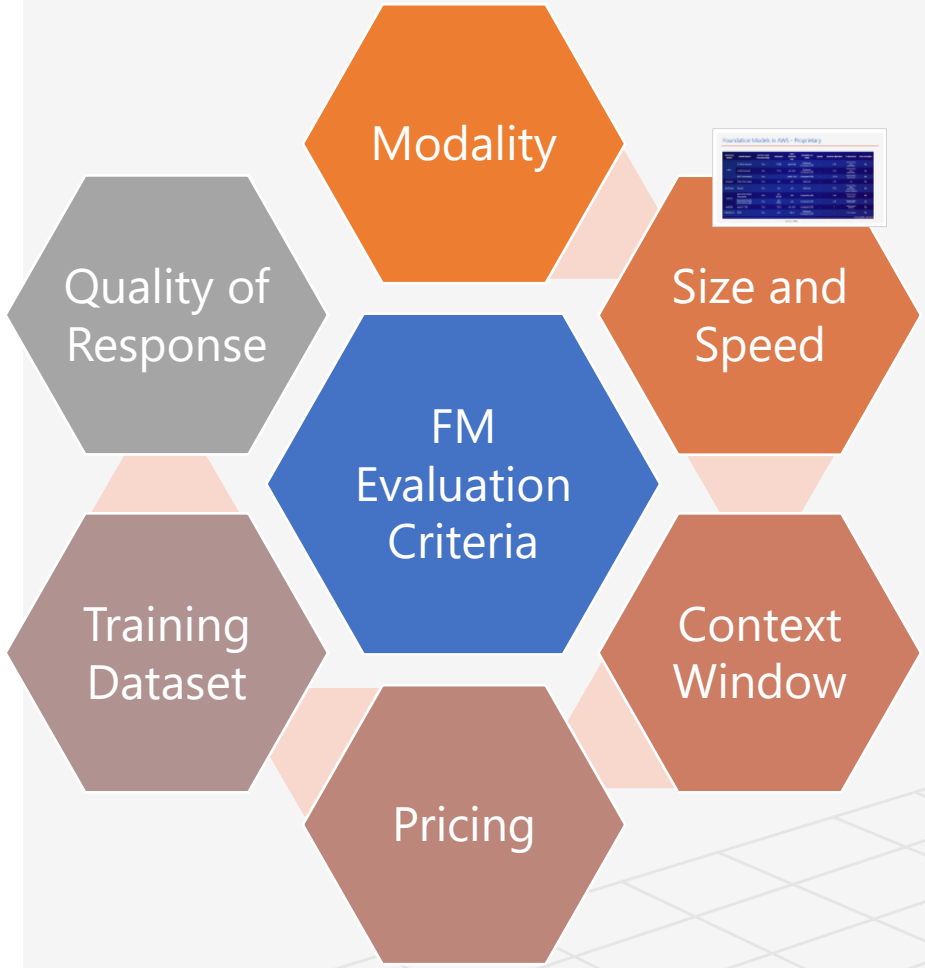
Generative AI – Manufacturing Industry Use Case

Use Case :

Text Summarization using **AWS Bedrock** for **faster issue resolution to improve productivity** of technicians.



Foundation Model Selection - Criteria



1. Modality : Text | Image (Vision) | Embedding - Command, Anthropic……
2. Size : Number of model parameters > 50B
3. Inference Speed or Latency : Response time for completion – Few Seconds
4. Context Window : 77- 200K token size – Claude has max context window
5. Pricing : FM pricing – Claude most expensive and Titan least
6. Training Dataset – Internet, Code, Human Feedback – Diverse dataset
7. Propriety or Open Source – Prefer Open Source
8. Fine-tunable – Should be fine-tunable
9. Additional Features – Multi-Lingual support – Jurassic, Titan
10. Quality of Response - Accuracy, Toxicity and Robustness

Foundation Model – Tokens, Parameters and Temperature

	Model name	Version	Provider	Modality	Max tokens	Description
○	Embed English	v3 New	Cohere	Embedding	512	Embed translates text into numeric
○	Embed Multilingual	v3 New	Cohere	Embedding	512	Embed translates text into numeric
○	Titan Embeddings G1 -...	v1.2	Amazon	Embedding	8k	The new Titan Embeddings G1 – Te
○	Titan Multimodal Emb...	v1 New	Amazon	Embedding	128 tokens	Titan Multimodal Embeddings Gen
○	SDXL 0.8	v0.8	Stability AI	Image	77	SDXL produces more detailed imag
○	SDXL 1.0	v1.0 New	Stability AI	Image	77	SDXL generates images of high qua
○	Titan Image Generator...	v1 - preview New	Amazon	Image	77 tokens	Titan Image Generator G1 is an ima
○	Jurassic-2 Ultra	v1	AI21 Labs	Text	8191	Jurassic-2 Ultra is AI21's most powe
○	Jurassic-2 Mid	v1	AI21 Labs	Text	8191	Jurassic-2 Mid is less powerful than
○	Claude	v2.1 New	Anthropic	Text	200k	An update to Claude 2 that feature
○	Claude	v2	Anthropic	Text	100k	Anthropic's most powerful base mo
○	Claude	v1.3	Anthropic	Text	100k	An earlier version of Anthropic's ge

Modality

Parameters

Max Tokens

Bedrock Console View

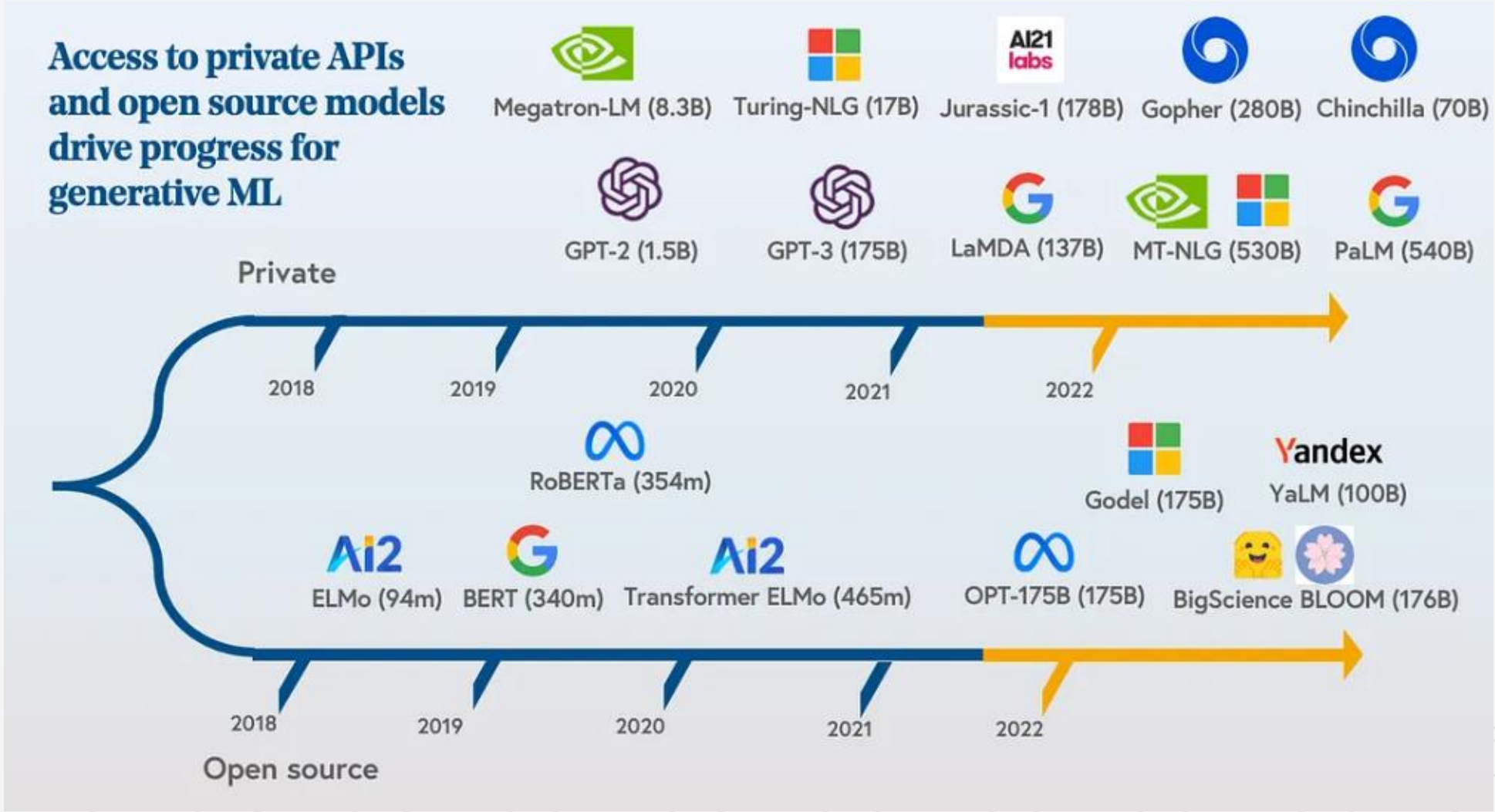
Foundation Models in AWS - Proprietary

Company Name	Model Name	Can be used Commercially	# Params	GPU instance req.	Available on AWS	Speed	Context Window	Trained on	Fine-tunable
AI21	J2 Ultra Instruct	Yes	178 B	p4d.24xl	Bedrock, Jumpstart/SM	-	8 K	Internet Data, Code, Instructions	No
	J2 Mid Instruct	Yes	17 B	g5.12xl	Bedrock, Jumpstart/SM	-	8 K	Internet Data, Code, Instructions	No
	AI21 Summarize	Yes		g4dn.12xl	Jumpstart/SM	-	~13 K	Internet Data, Instructions	No
Amazon	Titan Text Large	Yes	n/a	n/a	Bedrock	-	4 K	n/a	No
Anthropic	Claude	Yes	n/a	n/a	Bedrock	-	12 K	Internet Data, Code, Instructions, Human feedback	No
Cohere	Generate Model Command	Yes	n/a (50 B)	n/a	Jumpstart/SM	-	4 K	Internet Data, Instructions	No
	Generate Model Command-Light	Yes	n/a (6 B)	n/a	Jumpstart/SM	-	4 K	Internet Data, Instructions	No
LightOn	Lyra-Fr 10B	Yes	10 B	g5.12xl	Jumpstart/SM	-	?	Internet Data (French)	No
Stability AI	SDXL	Yes	n/a	g5.xl	Bedrock, Jumpstart/SM	-	-	<Text, Image>	No

*Last update July 2023

Source : AWS

Popular Foundation Models (Parameters perspective)



Cohere

- **Cohere** has **52 billion** parameters

Claude 2 by Anthropic

- **137 billion** parameters

Image Source : internet

Foundation Models in AWS – Open Source

Company Name	Model Name	Can be used Commercially	# Params	GPU instance req.	Available on AWS	Speed	Context Window	Trained on	Fine-tunable
Google	FLAN-UL2	Yes	20 B	g5.12xl	Jumpstart/SM	-	2 K	Internet Data, Code, Instructions	Yes
	FLAN-T5-XXL	Yes	11 B	g5.xl	Jumpstart/SM	-	512	Internet Data, Code, Instructions	Yes
Eleuther	GPT-J	Yes	6 B	g5.xl	Jumpstart/SM	-	512	Internet Data, Code	Yes
TII	Falcon-40B-Instruct	Yes	40 B	g5.12xl	Jumpstart/SM	-	2 K	Internet Data, Code, Instructions	Yes
	Falcon-7B-Instruct	Yes	7 B	g5.xl	Jumpstart/SM	-	2 K	Internet Data, Code, Instructions	Yes
BigCode	Starcoder	Yes	15 B	g5.12xl	SM	-	8 K	Code	Yes
	Santa Coder	Yes	1.1 B	g5.xl	SM	-	2K	Code	Yes
LMSYS Org	Vicuna-13B	No	13 B	g5.xl	SM	-	2 K	Internet Data, Code, Instructions	Yes
Meta	Llama-65B	No	65 B	g5.48xl	SM	-	2 K	Internet Data, Code	Yes
Stability AI	SD 2.1	Yes	-	g5.xl	Jumpstart/SM	-	-	<Text, Image>	Yes

*Last update July 2023

Source : AWS

AWS Bedrock Foundation Model Evaluation Service

AWS Bedrock Model Evaluation Service

1. Automatic

- Task Type ----> Text Generation, Text Summarization, Q&A, Text Classsification
- DataSet –-- > Built in – Gigaword, XSUM or Bring your own dataset
- Metrics --- > Accuracy, Toxicity and Robustness

2. Human: Bring your own work team

- Evaluates up to **2 models** using a **work team** of your choice to provide feedback.
- Provides **results based** on the parameters that are specified while creating the evaluation.

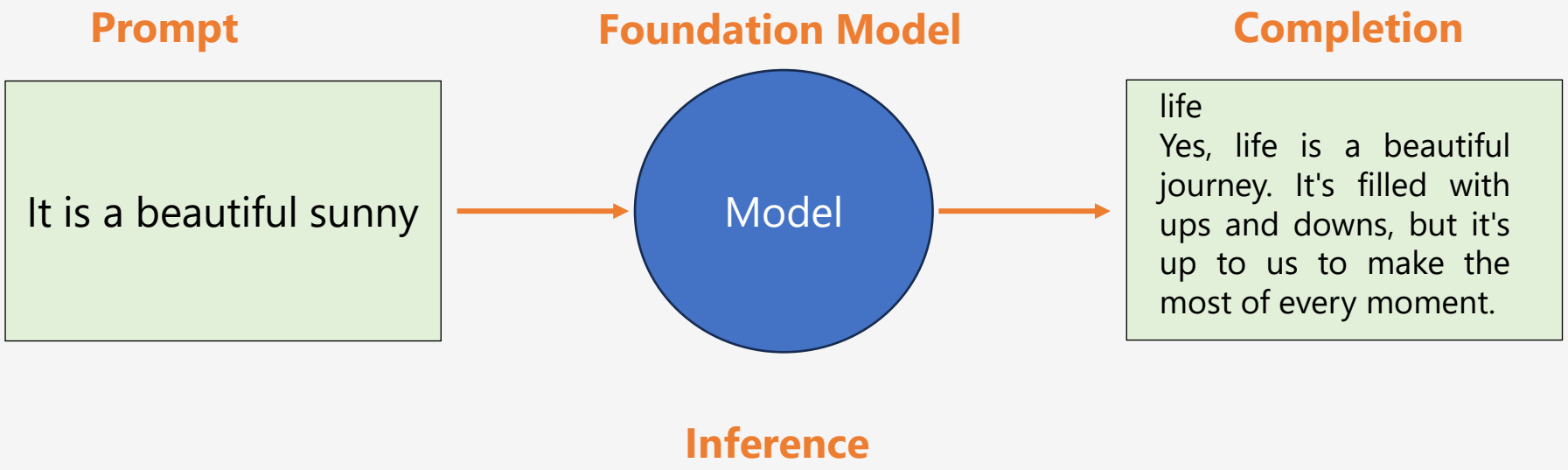
3. Human: AWS Managed work team

Generative AI Application Lifecycle

1	2	3	4
Scope of Program	Model Selection & Architecture Alignment	Adapt and Align Foundation Model	Integration
• Identify the Use Case • Define key objectives & outcomes	• Define model selection criteria • Evaluate and select the Foundation Model	• **Prompt Engineering** • Fine Tuning • RAG	• Deploy the Model • Integrate with existing application landscape

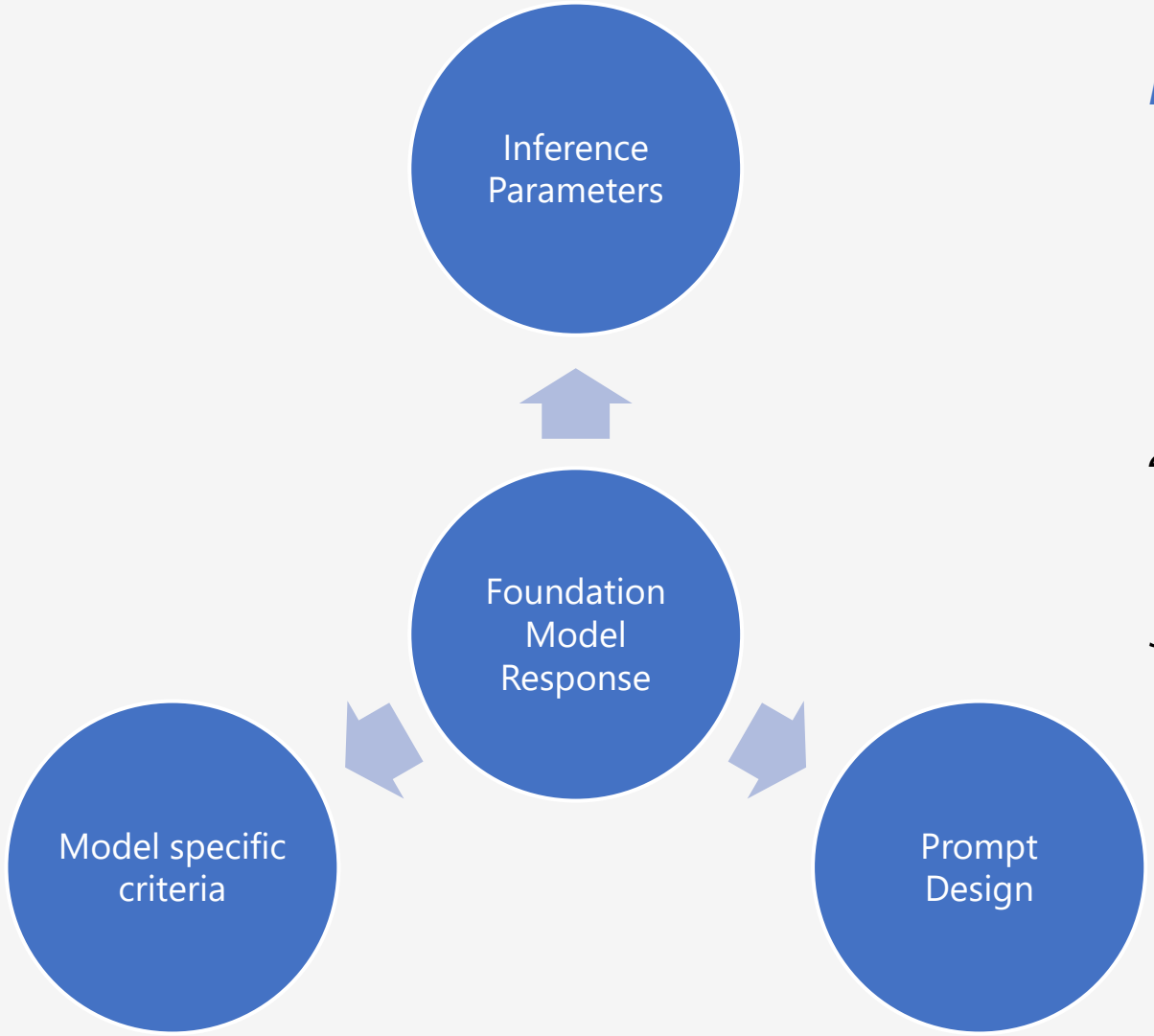
GenAI Project Lifecycle – Prompt Engineering

What is a Prompt ?



- ***Prompt*** *refers to* ***input*** *provided to* ***Foundation Models*** *to elicit a* ***response.***
- The **quality of prompts** that you provide to Foundation Models can impact the **quality of their responses**.
- ***Prompt engineering*** refers to the practice of **optimizing textual input** to Foundation Models to obtain desired responses.

Factors impacting good response/completion from Foundation Models



Factors influencing good Completion from FM's:

1. Prompt Design

2. Inference Parameters

3. Knowledge of Foundation Model

Manufacturing Use Case - Prompt for Summarization Task

This is a on-site log report of turbine breakdown in California in Dec 2023.

Issue Log Date – 25-12-2023, Model Number – TB-CL-7882, **Issue** - Cracks appeared in the part MR 7882-9571 next to the rotor hub. The nut connecting the rotor blade to the rotor hub seems to be damaged. The Anemometer readings seem to be within range. The electric braking seems to be unused. No indication of damage to any other component of the turbine except normal wear and tear.

Potential Root Cause – Seems due to reduced tensile strength of the nut connecting the blade to the rotor. Last Maintenance Date – 12-12-2023, **Last Maintenance Issues Recorded** - No known issues recorded and all the parameters were within range.

Summarize the text in 2 lines.

Content creator and copyright : Rahul Trisal

2. Elements of a Good Prompt Deign

This is a review of football world cup Qatar 2022 :

"Lionel Messi can finally be called a world champion.
Messi scored twice in one of the most epic soccer games anyone has ever watched as Argentina won the 2022 FIFA World Cup Final over France on penalties.
The climactic match in Qatar finished 3-3 after extra time, with La Albiceleste claiming the shootout by a 4-2 margin.
Argentina held a comfortable 2-0 lead until the 80th minute courtesy of a Messi penalty and a sublime team goal finished by Ángel Di María in the first half. However, Kylian Mbappé converted from the spot and finished a sumptuous volley in a span of two minutes to send the game to extra time.
Messi was once again on hand to put Argentina in front in the 108th minute, but Mbappé kept his cool from the penalty spot once more to send the final to a shootout. Messi knocked home the first penalty, then Argentina keeper Emiliano Martínez made a save in the second round from Kingsley Coman's effort. France's Aurélien Tchouaméni sent his spot-kick wide in the third round, leaving Gonzalo Montiel to seal the title in the fourth."

Summarize the above review in 2 lines:

Contextual Information about the task

Input text for Task

1. **Task Specification – Summarize**
2. **Output Specification – 2 lines**

1. Clear concise instructions 2. Task to be accomplished at the end 3. Output specification

Designing Prompts varies based on Foundation Model used - Claude

1. **Use the correct format**
   - Claude is trained as a **conversational agent** using special tokens to mark who is speaking.
   - The **Human: (you)** asks a question or gives instructions
   - **Assistant: (Claude)** responds.

Prompt

```
Human:

Assistant:
```

2. **Describe the task well**
   - Good to give Claude as much context and detail as possible

Bad output

```
Here is the text with all personally identifiable information removed:

Joe: Hi [Name 1]!
[Name 1]\: Hi [Name 2]!  Are you coming over?
[Name 2]\: Yup!  Hey I, uh, forgot where you live.
[Name 1]\: No problem!  It's [Address], [City] [State] [Zip Code].
[Name 2]\: Got it, thanks!
```

Good prompt

```
Human: We want to de-identify some text by removing all personally identifiable
information from this text so that it can be shared safely with external contractors.

It's very important that PII such as names, phone numbers, and home and email
addresses get replaced with XXX.

Here is the text you should process: {{YOUR TEXT HERE}}

Assistant:
```

Designing Prompts - Claude

3. Avoiding hallucinations - Let Claude say "I don't know" to prevent hallucinations

Bad prompt

```
Human: What is the heaviest hippo ever recorded?

Assistant:
```

Bad output

```
The heaviest hippo ever recorded was a bull named Hubert, who weighed 5,000 kg or
11,023 lbs. Hubert was measured in Kenya in the early 20th century. Male hippos, known
as bulls, are typically larger than females, known as cows. Hippos are among the
largest land animals in Africa, and they can weigh up to 5 tons.
```

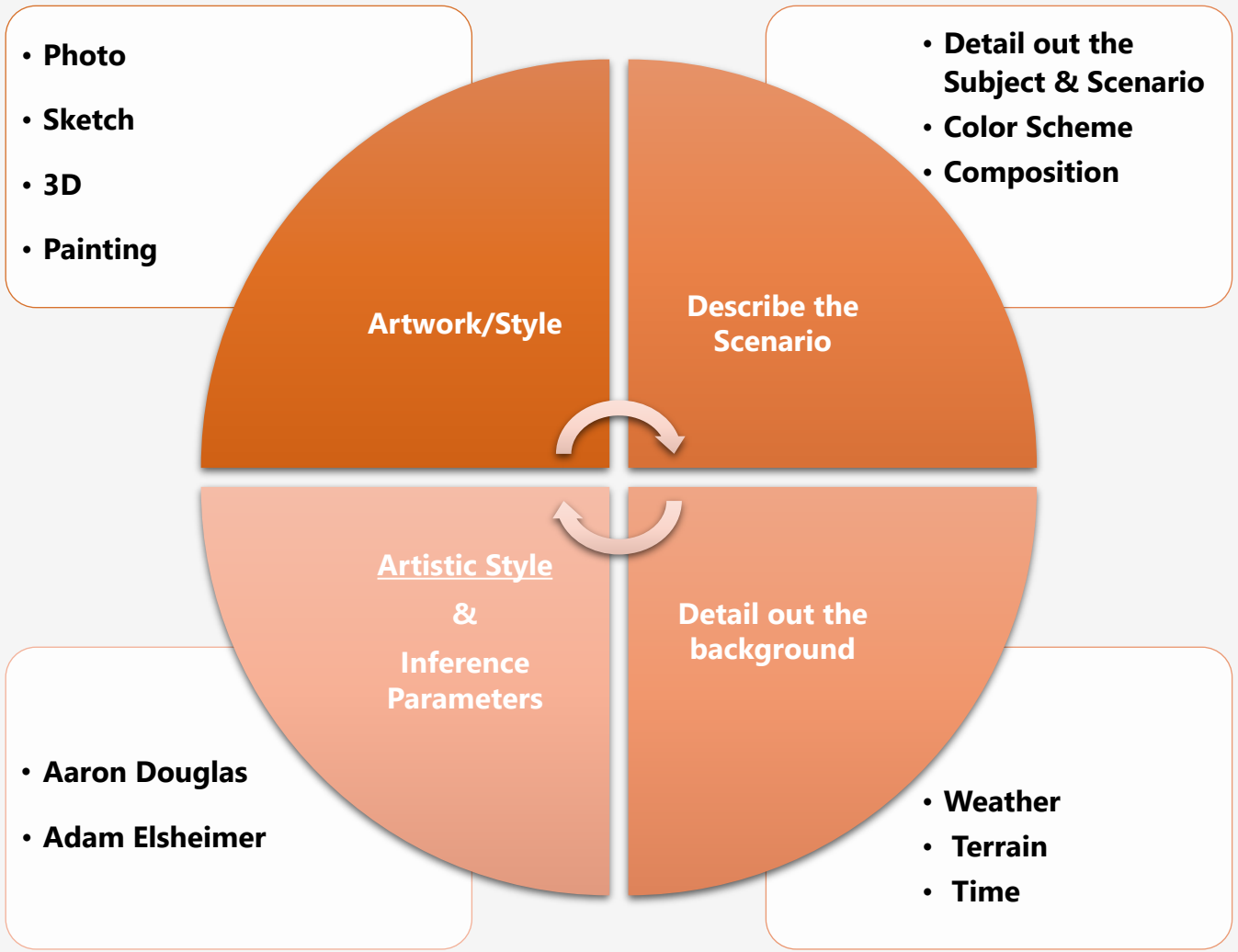
Good prompt

```
Human: Answer the following question only if you know the answer or can make a well-
informed guess; otherwise tell me you don't know it.

What was the heaviest hippo ever recorded?

Assistant:
```

Good output

```
I'm sorry, I don't know the exact weight of the heaviest hippo ever recorded. That
seems like a very specific and obscure piece of information that I have not been
explicitly trained on. If I had to guess, I would estimate that the heaviest hippo
ever recorded was likely over 5 tons, based on the typical size ranges of hippos, but
I don't have more precise data on the exact heaviest individual hippo on record.
```

Prompt Design for Stability AI



Start prompt with :

- "An image of ….
- Use detailed scenario
- Provide details
  - Medium
  - Color
  - Time of Day etc.
- Style
- Photo/Sketch etc.
- Inference Parameters

Prompt Design for Stability Diffusion - 1

An image of a **spy agent fighting** in a rival country with guns and helicopters with backdrop of a shopping complex with heavy snow and old **Greek architecture** building late in the **evening** with **sun setting behind mountains**. The image should be a **photograph** with Aaron Jasinski style

Prompt Design for Amazon Titan and AI21 – Text generation

Text generation

Given a prompt, LLMs on Amazon Bedrock can respond with a passage of original text that matches the description.

Prompt template for Amazon Titan and AI21 Jurassic: *"""Please write a {{Text Category}} in* ***the voice of {{Role}}****."""*

User prompt:

Please write an email in the voice of a friend

congratulating someone on a new job.

Output:

Hey [Name],

I just found out about your new job, and I'm so happy for you! Congratulations! You deserve this opportunity since you worked so hard to get it. I know you'll do great and make a positive impact on your team. I'm here to support you every step of the way.

Wishing you all the best in this exciting new chapter of your life!

Prompt Engineering Techniques

Baseline
Foundation Model
+
Prompt Template

Zero-Shot Prompt

One - Shot Prompt

Few - Shot Prompt

Perfect Outcome ☺

Zero-Shot Prompting

Zero-Shot Prompting

- FM's are capable of **performing some simple tasks "zero-shot."**
- **No examples** need to be provided to accomplish the task

PROMPT

Write an engaging product description for a T-shirt for a clothing eCommerce site:

COMPLETION

This T-shirt is perfect for any occasion. It's made of high-quality materials and it's comfortable to wear. It's also stylish and versatile, so you can wear it with anything.

One -Shot Prompting

One-Shot Prompting

- One-Shot Prompting provides **examples to the model** to assist the model to comprehend and generate responses

PROMPT

Write an engaging product description for a clothing eCommerce site.

Make sure to include the following features in the description.

Product: Women's Boho Beach Dress Features: - Midi dress - Swing hem - Slimming - Special for summer

Description: Women's Boho Beach Dress The style of midi dress with swing hem is slimming, It will make your tummy look flatter. This dress for women could not only make you feel cool, but also perfectly present the curve of female waist. Even in hot summer, it still keeps your elegant.

Write an engaging product description for a clothing eCommerce site.

Make sure to include the following features in the description.

Product: Humor Men's Graphic T-Shirt Features: - Soft cotton - Short sleeve - Have a print of Einstein's quote: "artificial intelligence is no match for natural stupidity"

Description:

COMPLETION

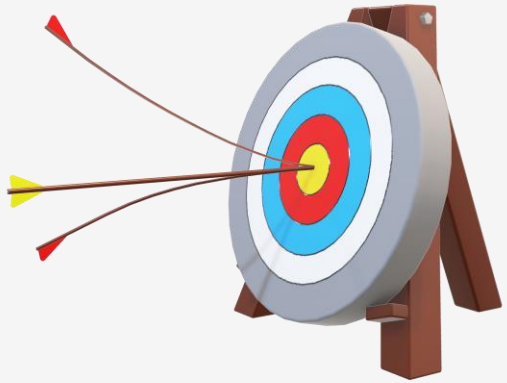
This soft cotton t-shirt is perfect for fans of the classic scientist, Albert Einstein. The quote "artificial intelligence is no match for natural stupidity" is a reminder that sometimes the most intelligent people are the most foolish. So whether you're a science nerd or just a fan of witty quotes, this t-shirt is for you.

Few - Shot Prompting

Few-Shot Prompting

- Few examples are provided to the Model to generate high quality Completion
- Good for Classification

```
PROMPT
Write an engaging product description for a clothing eCommerce site. Make sure to include the followi
Product: Camp Director Crossbody Bag
Features:
- Canvas purse
- Rainbow space print
- Leather trim
- Two securely-zipped compartments
Description:
This Camp Director Crossbody Bag is the perfect accessory for carrying your essentials in style. The can
##
Write an engaging product description for a clothing eCommerce site. Make sure to include the followi
Product: Women's Boho Beach Dress
Features:
- Midi dress
- Swing hem
- Slimming
- Special for summer
Description:
Women's Boho Beach Dress The style of midi dress with swing hem is slimming, It will make your tumı
##
Write an engaging product description for a clothing eCommerce site. Make sure to include the followi
Product: Humor Men's Graphic T-Shirt
Features:
- Soft cotton
- Short sleeve
- Have a print of Einstein's quote: "artificial intelligence is no match for natural stupidity"
Description:
```



COMPLETION

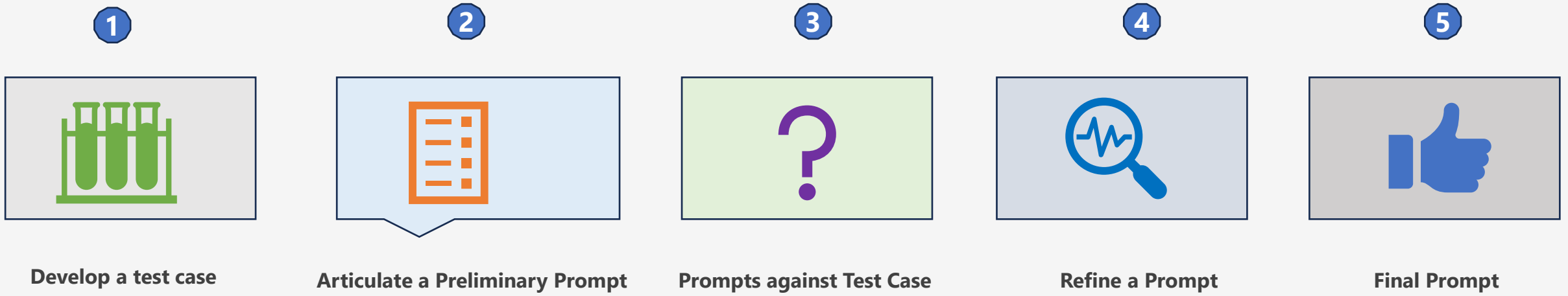
This Humor Men's Graphic T-Shirt is perfect for anyone who wants to make a statement.

The soft cotton and short sleeve make it comfortable to wear, and the print of Einstein's quote is sure to get a chuckle.

Whether you're a fan of science or just looking for a funny shirt, this shirt is a great choice.

Example Source : A121 Labs

Steps to engineer a good prompt



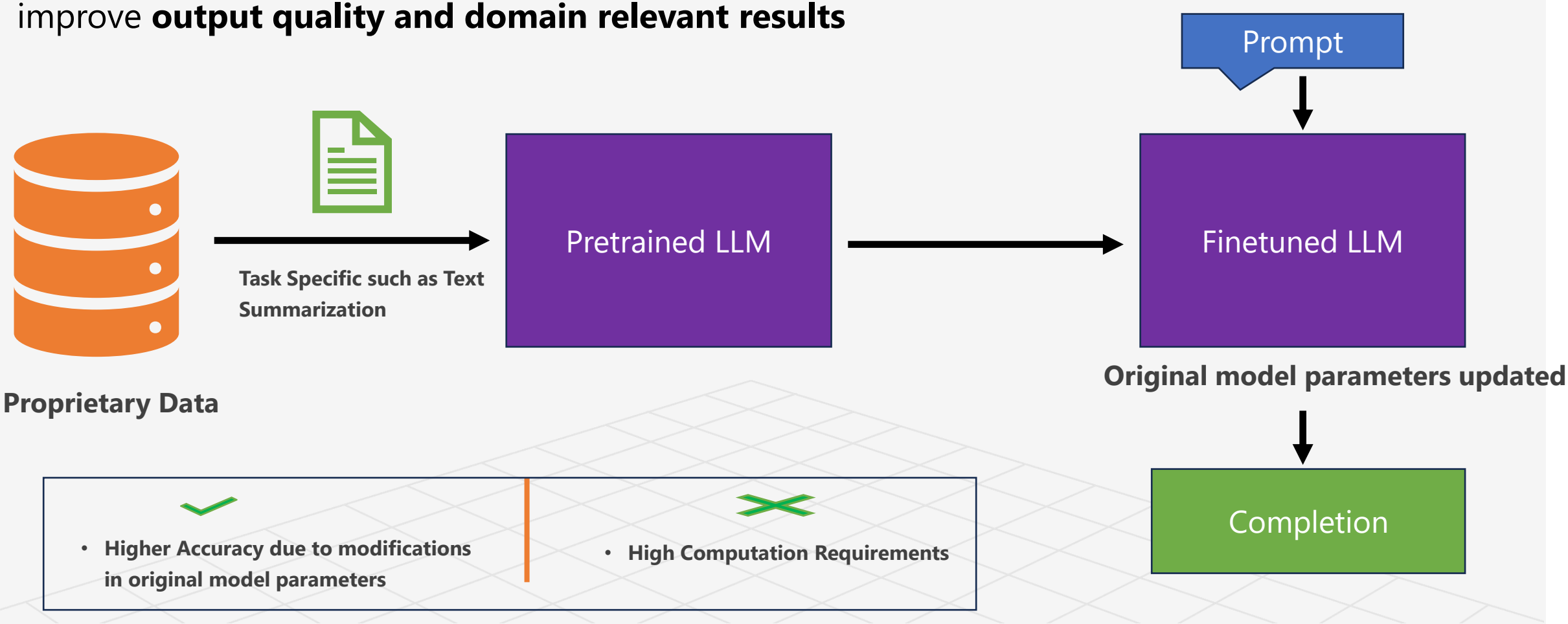
GenAI Project Lifecycle – FineTuning Foundation Models

Generative AI Application Lifecycle

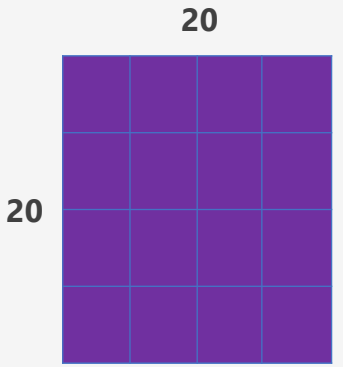
1	2	3	4
Scope of Program	Model Selection & Architecture Alignment	Model Adaptation	Integration
• Identify the Use Case • Define key objectives & outcomes	• Define model selection criteria • Evaluate and select the Foundation Model	• Prompt Engineering • **Fine Tuning** • RAG	• Deploy the Model • Integrate with existing application landscape

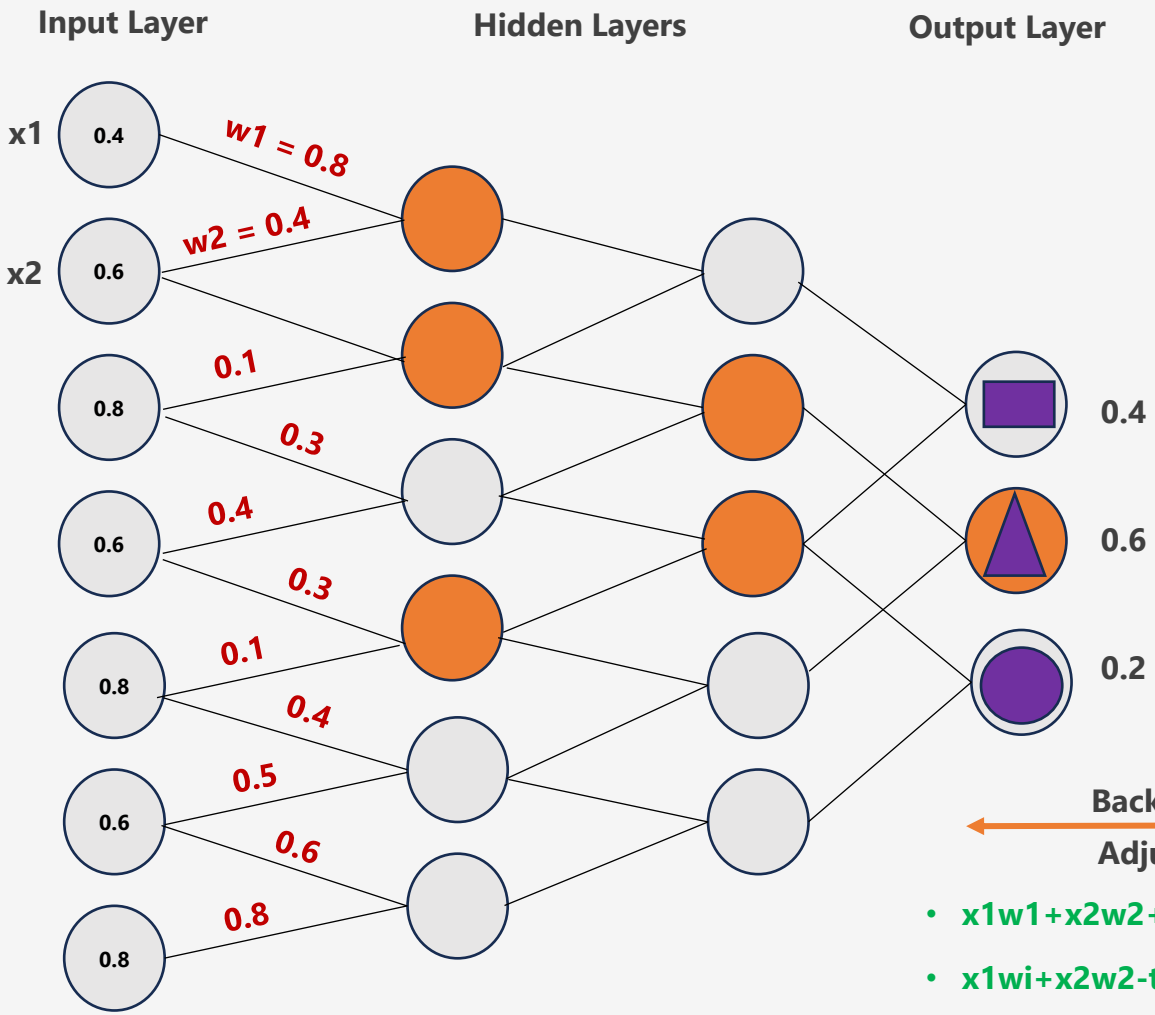
What is Fine Tuning Large Language Models (or Foundation Models)

Fine-tuning is a technique to train a model using **proprietary or domain specific data** to improve **output quality and domain relevant results**



Machine Learning – Deep Learning with Artificial Neural Networks





Output	Error
1	+0.6
0	-0.6
0	-0.2

Backward Propagation

Adjust Weights

- **x1w1+x2w2+… > threshold value**
- **x1wi+x2w2-threshold >0**
- **x1w1+x2w2+Bias >>> Activation Function**

Key terms

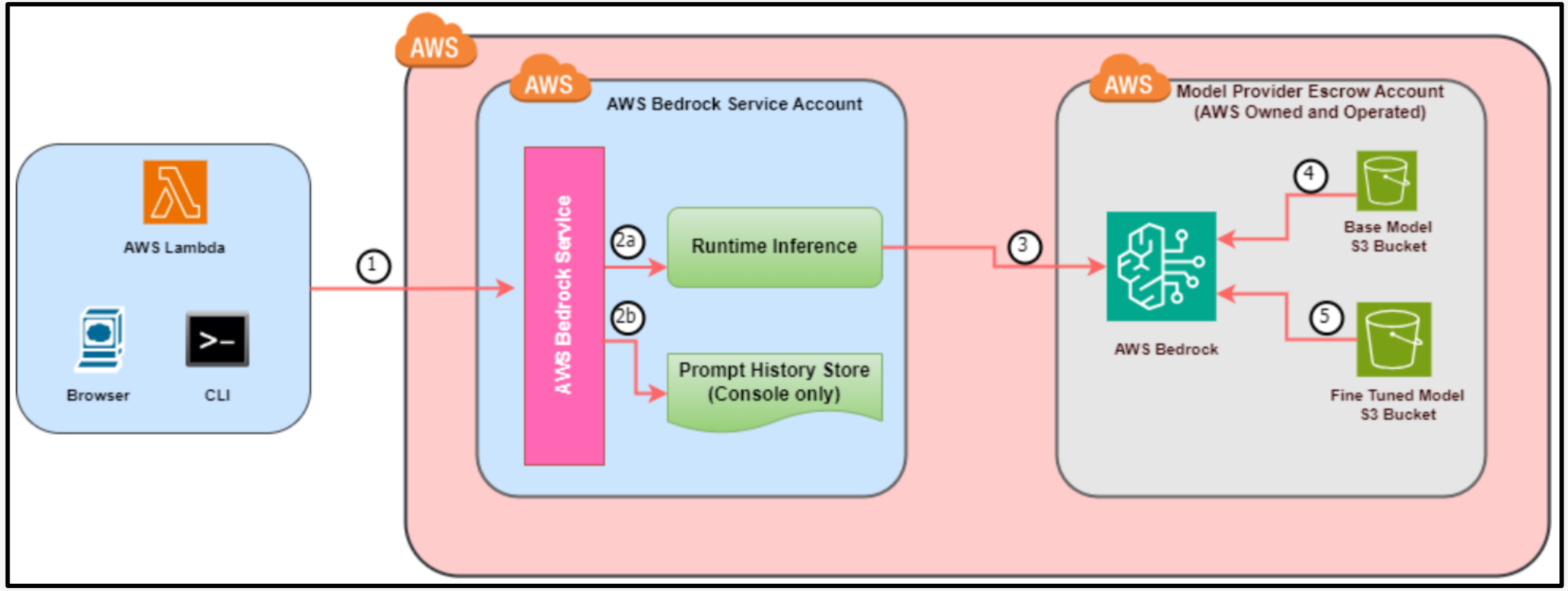
- Neurons (Binary Value)
- Weights
- Bias/Threshold
- Activation Function
- Feedforward network
- Actual Output
- Error
- Forward Propagation
- Backward Propagation

Amazon Bedrock – Key Customer Challenges from Data Security

Key Customer concerns in adopting AWS Bedrock :

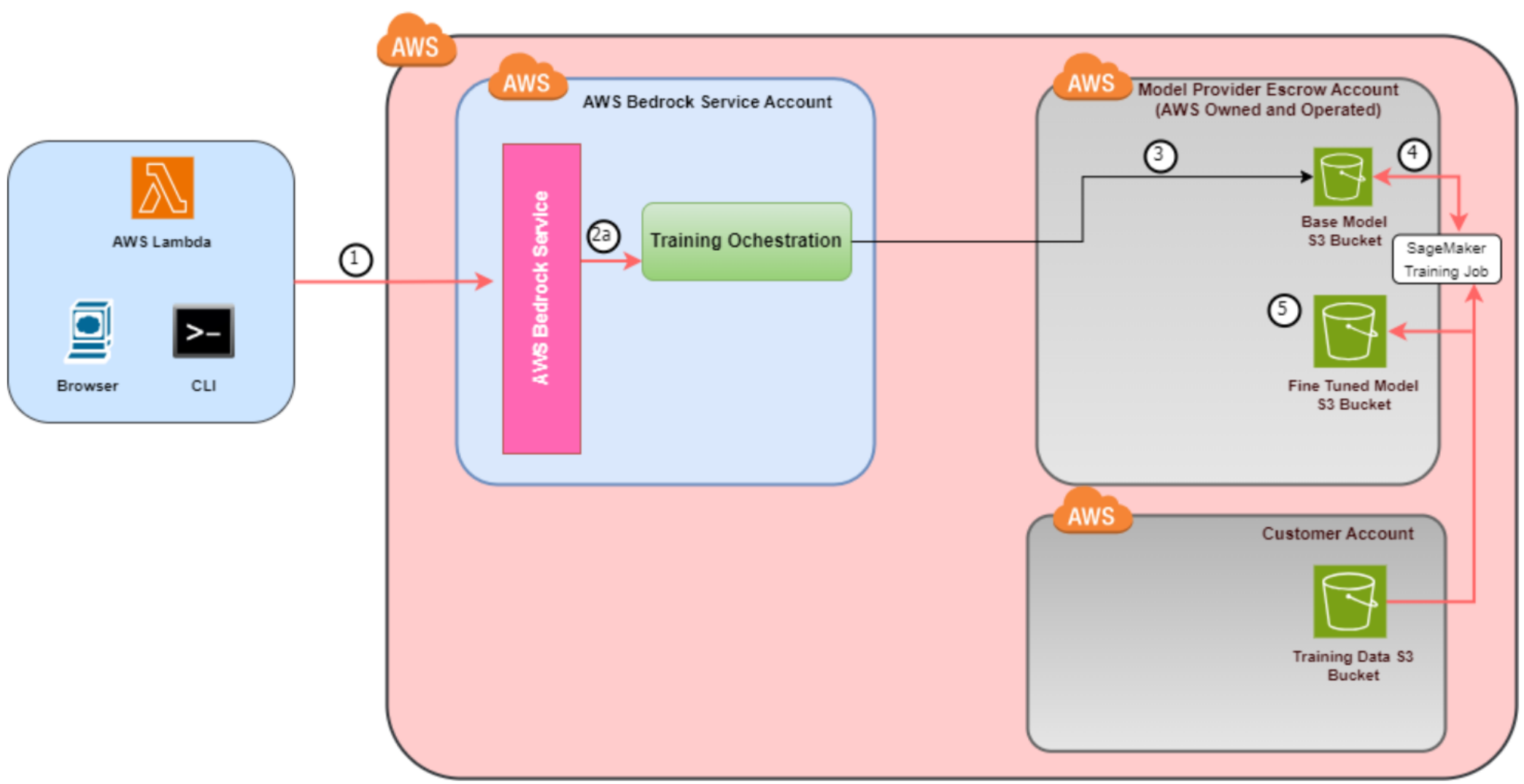
- Access to Customer Data used to Fine Tune the Foundation Model
- Access to Fine Tuned Model

Amazon Bedrock – Architecture – Part 2



- Runtime Inference : Used to redirect to right model endpoint based on the API request
- Base Model : Baseline model provided to every AWS account
- Fine Tuned Model : When Base Model is trained using custom labelled data to generate Fine Tuned Model

Amazon Bedrock – Architecture – Fine Tuning Foundation Models



Fine Tuning Foundation Models - Demo

DEMO

Fine Tuning Foundation Models – Model Parameters and Hyperparameters

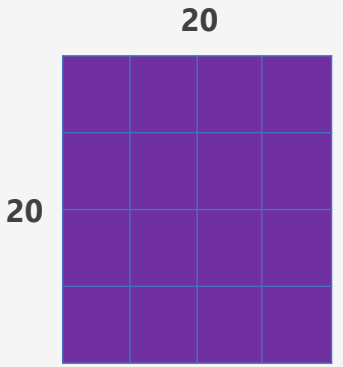
Parameters :

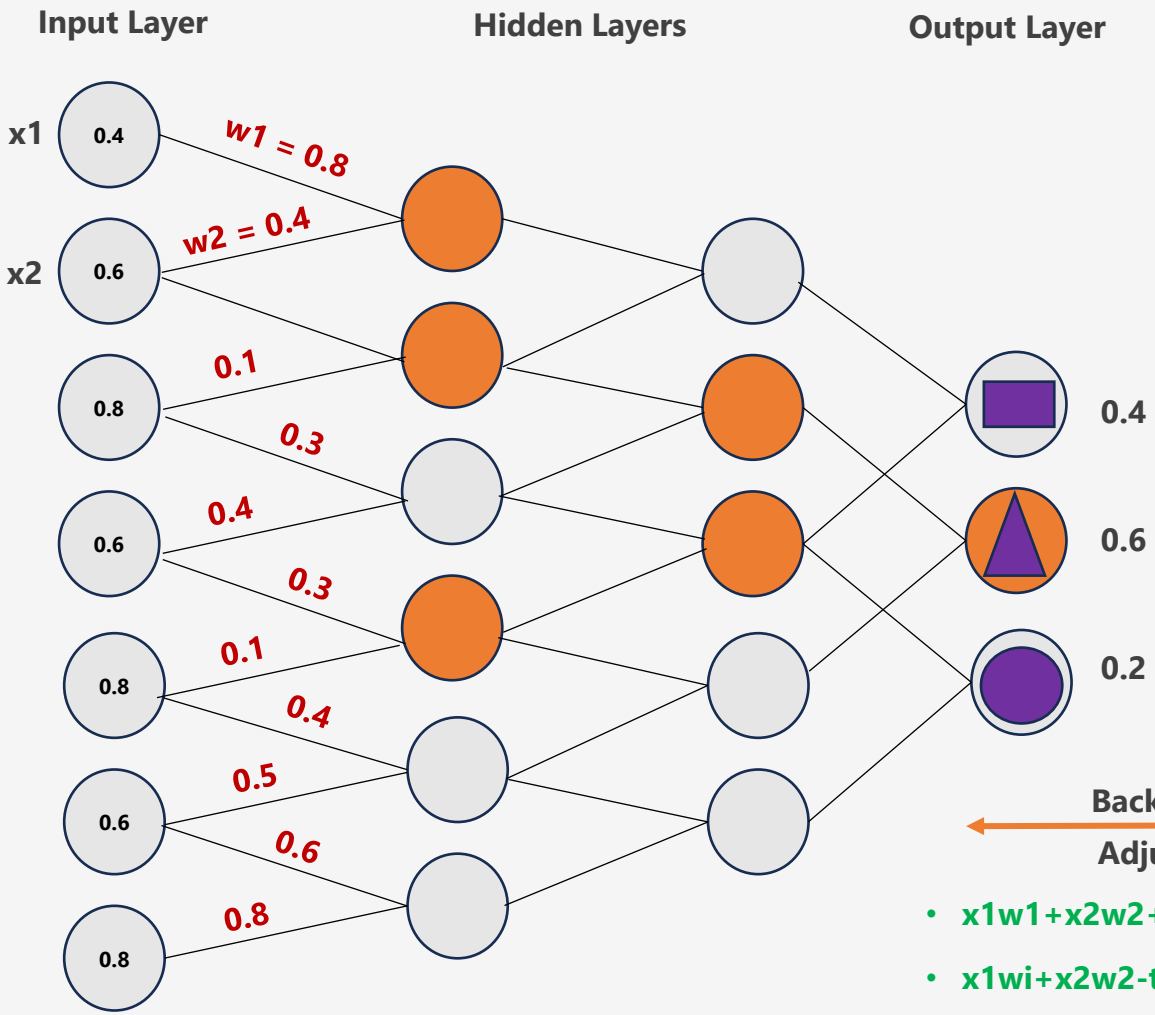
- Internal to Model
- Updated during training process

Hyperparameters :

- External to Model
- Specified by developers/scientist

Machine Learning – Deep Learning with Artificial Neural Networks





Output	Error
1	+0.6
0	-0.6
0	-0.2

Backward Propagation

Adjust Weights

- **x1w1+x2w2+… > threshold value**
- **x1wi+x2w2-threshold >0**
- **x1w1+x2w2+Bias >>> Activation Function**

Key terms

- Neurons (Binary Value)
- Weights
- Bias/Threshold
- Activation Function
- Feedforward network
- Actual Output
- Error
- Forward Propagation
- Backward Propagation

Instruction Fine Tuning – Key Terminology to set Hyperparameters

Terminology	Description	Example
Sample	One row of data	1
Dataset	Total number of training samples	100
Batch Size	Total number of training samples in each mini-batch	20
Iterations	Number of times Batch Size is run for entire dataset	5
Epoch	Epoch elapses when an entire dataset is run	1-Infinity
Learning rate	The rate at which model parameters are updated after each batch of training data.	

Use Case 3:

Code Generation with AWS CodeWhisperer

(From my AWS CDK Udemy Course)

AWS CodeWhisperer (Generative AI Tool powered by LLM)

- Amazon CodeWhisperer is an **AI coding** companion that **helps improve developer productivity** by **generating code recommendations based on their comments in natural language** and code in the (IDE).
- As you write code, **CodeWhisperer automatically generates suggestions** based on your existing code and comments.
- CodeWhisperer is powered by a **Large Language Model (LLM) that is trained on billions of lines of code**, and as a result, has learned how to write code in 15 programming languages.

"Accenture is using Amazon CodeWhisperer to accelerate coding as part of our software engineering best practices initiative in our Velocity platform," says Balakrishnan Viswanathan, Senior Manager, Tech Architecture at Accenture. "The Velocity team was looking for ways to improve developer productivity. After searching for multiple options, we came across Amazon CodeWhisperer to reduce our development efforts by up to 30% and we are now focusing more on improving security, quality, and performance."

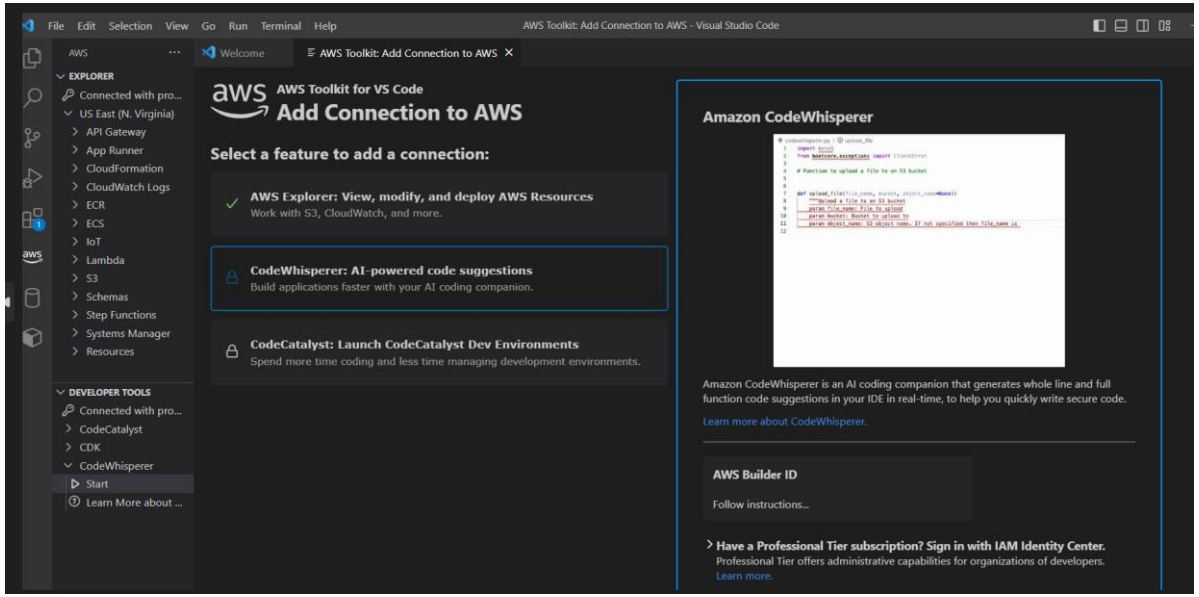
Source : https://aws.amazon.com/blogs/machine-learning/how-accenture-is-using-amazon-codewhisperer-to-improve-developer-productivity/

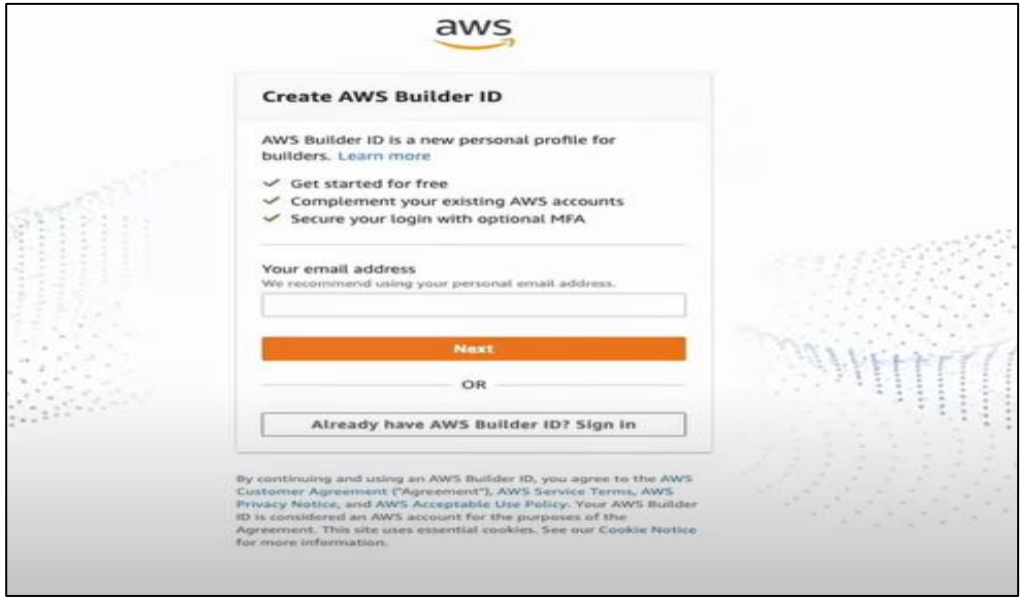
AWS CDK and CodeWhisperer (Generative AI powered by LLM)

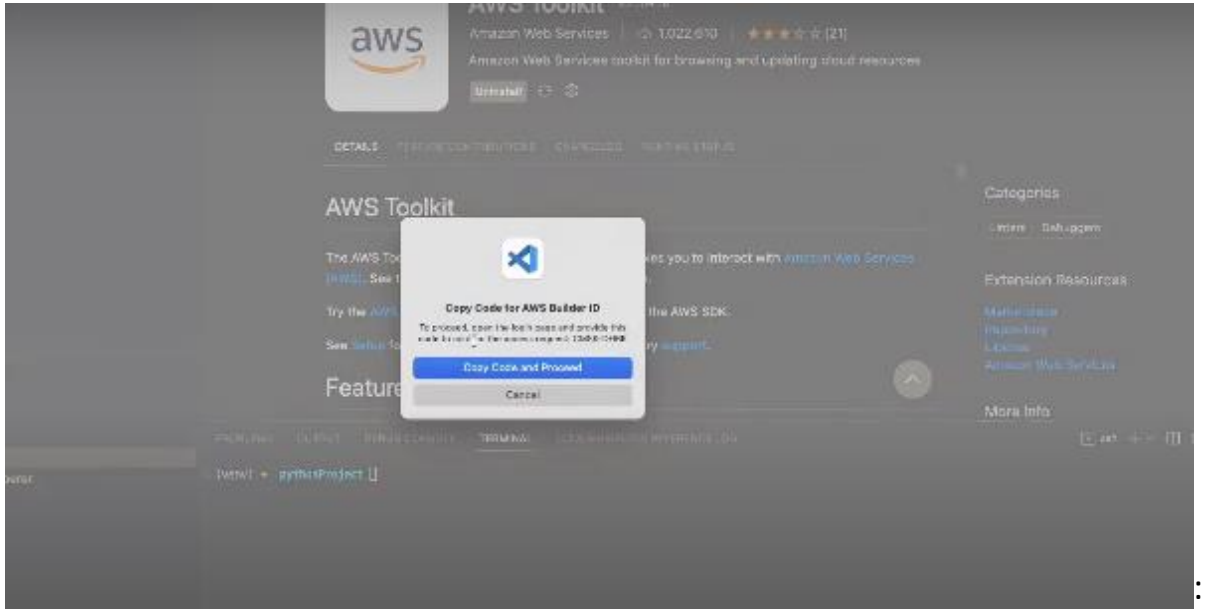
Install Code AWS Whisperer on VSCode from below link:

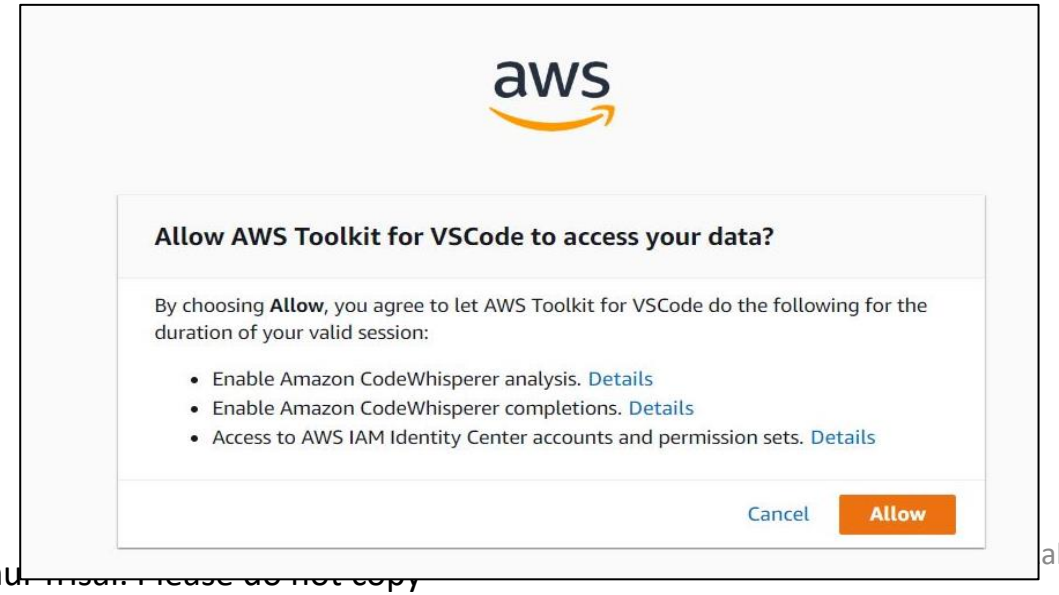
- Use **Individual Tier which is** free to use and requires a simple sign-up to create an **AWS Builder ID.**
- https://docs.aws.amazon.com/codewhisperer/latest/userguide/whisper-setup-indv-devs.html
- https://www.youtube.com/watch?v=rHNMfOK8pWI

AWS CDK and CodeWhisperer (Generative AI powered by LLM)



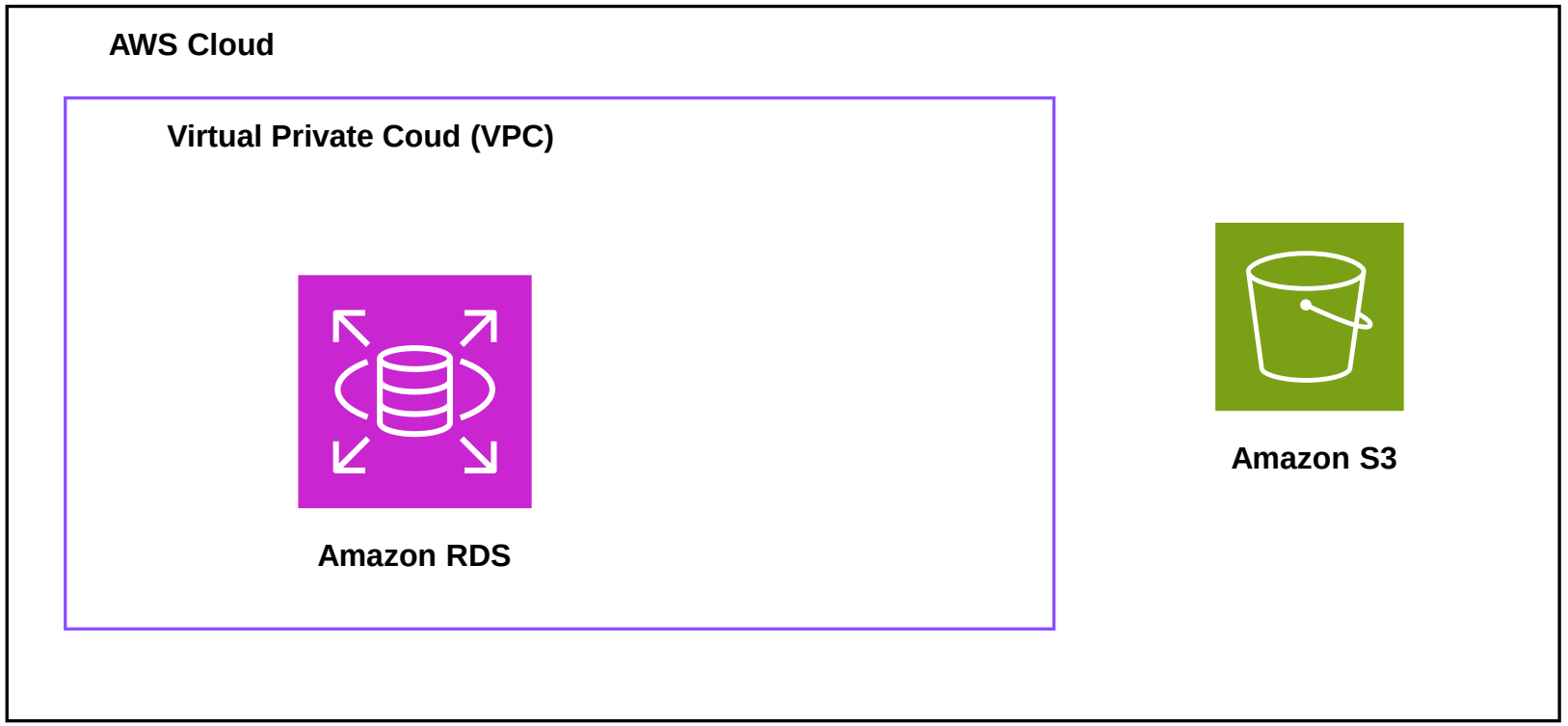






AWS CDK and CodeWhisperer (Generative AI powered by LLM)

What will we build using AWS CodeWhisperer and CDK ?



Summary of Steps to create any AWS Resource using CDK v2

Step 1. – Open the new folder in Visual Studio Code Editor and open Terminal

Step 2. – Create the app: Create the Directory - ***mkdir codewhisperer and cd codewhisperer***

Step 3. – Initialize the CDK with ***cdk init app --language typescript***

Step 4. – *Import the module* for aws service being created - **Link**

Step 5. – *Define Scope, Logical ID and Props* – ***(this, 'logical id', {props})***

Step 6. – *Build the app* (Optional) with ***npm run build***

Step 7. – *Bootstrap* (One Time) with ***cdk bootstrap***

Step 8. – *Synthesize* an AWS CloudFormation template for the app with ***cdk synth***

Step 9. – *Deploying* the stack with ***cdk deploy***

AWS CDK v2 – CodeWhisperer

Keyboard Shortcut	Action
[TAB]	To accept suggestions from Code Whisperer
[left arrow] and [right arrow]	Navigate between suggestions.
Keep typing (or hit ESC).	To ignore the suggestions from Code Whisperer
MacOS: Option + C **Windows: Alt + C**	Prompt Code Whisperer for Suggestions

AWS CDK v2 – CodeWhisperer – AWS Recommendations

- Write **descriptive comments.** “Function to upload a file to S3” will get better results than “Upload a file”.
- **Specify the libraries** you prefer by using import statements.
- Use **descriptive names for variable and functions**. A function called “upload_file_to_S3” will get better results than a function called “file_upload”.
- Give CodeWhisperer something to work with. The **more code your file contains**, the more context CodeWhisperer has for generating recommendations.

Source : AWS Documentation

Python Refresher

(from my Udemy Course on :

AWS Lambda, Python(Boto3) & Serverless-

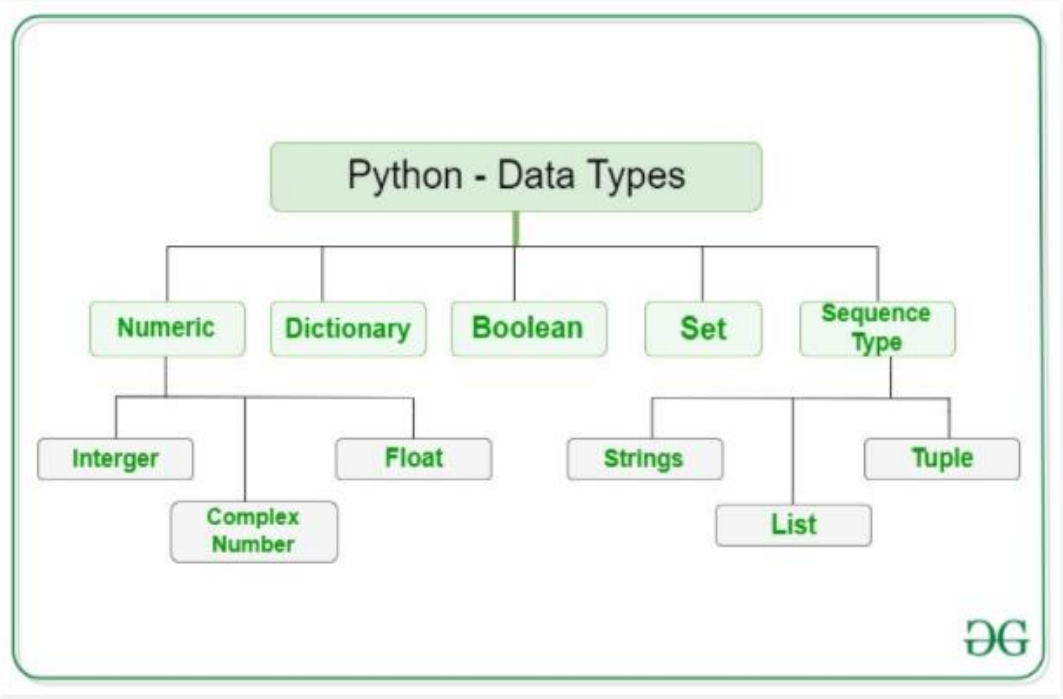
Beginner to Advanced)

Python Basics–Refresher

1. Print Function – print the message to screen or any interface; **Syntax : print( )**

2. Variables - Containers for storing data values string, float or integers and no need to declare; **Syntax : x = 3, greeting = “hello” etc.**

3. Data Type



Data Type	Example
Int	x = 20
Float	x = 20.5
Dictionary	x = {"name" : "John", "age" : 36}
Strings	x = "Hello World"
List	x = ["apple", "banana", "cherry"]

Python Basics–Refresher

3. Data Types – Dictionary

- curly brackets
- key: pair values
- Nested Dictionary

Dict

response = {1: 'Rahul', 2: 'John', 3: 'Joy'}

Nested Dict

response = {1:'Python', 2:{'books': 'arch' , 'aws':'Lambda'}}

4. Nested Dictionary

Python Basics–Refresher

5. Sample Example from Boto3

Response Syntax

```
{
    'Buckets': [
        {
            'Name': 'string',
            'CreationDate': datetime(2015, 1, 1)
        },
    ],
    'Owner': {
        'DisplayName': 'string',
        'ID': 'string'
    }
}
```

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3/client/list_buckets.html#

Python Basics–Refresher

6. Data Types – List

- Lists in Python can be created by just placing the sequence inside the **square brackets []**
- A single list may contain Data Types like Integers, Strings, as well as Objects.
- List in Python are ordered and have a definite count. The elements in a **list are indexed with 0 being the first index.**
- **slice(start, stop, step)**
- Reverse [: : -1]

……

list = [1, 4, 'For', 6, 'Anisha’]

……

7. Nested List

Nested List => nestedList = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

Python Basics–Refresher

8. Data Types – List and Dictionary

Response Syntax

```
{
    'Buckets': [
        {
            'Name': 'string',
            'CreationDate': datetime(2015, 1, 1)
        },
    ],
    'Owner': {
        'DisplayName': 'string',
        'ID': 'string'
    }
}
```

9. Data Type Determination

response = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

print(type(response))

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3/client/list_buckets.html#

Python Basics–Refresher

10. Function

Python Function

A function is a block of code which runs when it is called.

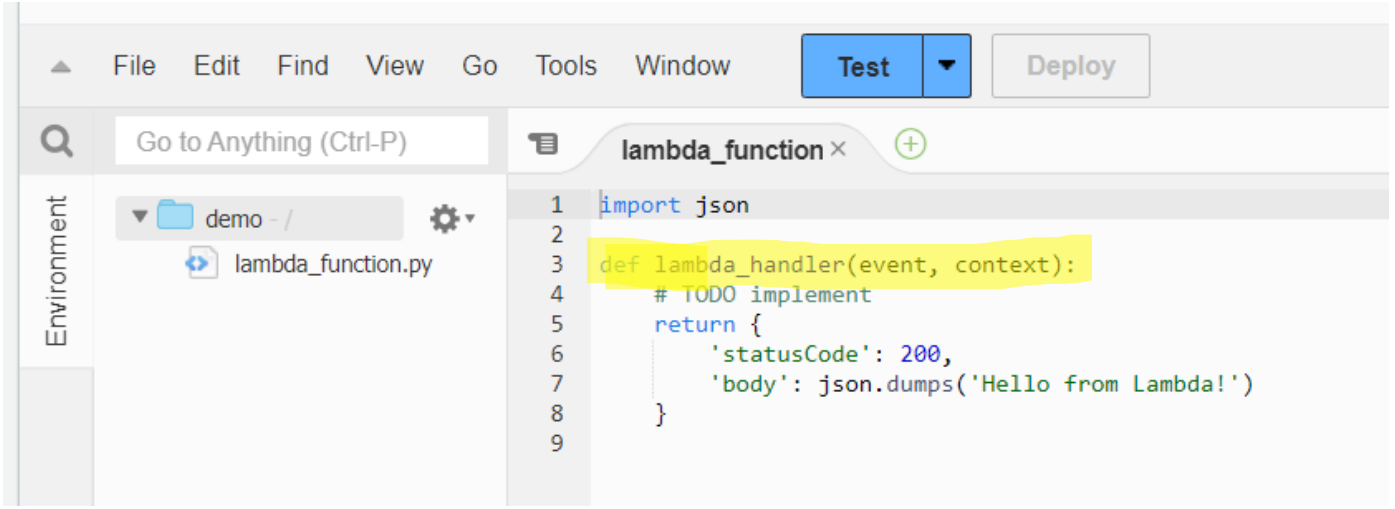
Syntax:

```
def function_name (argument/parameters):
    return expression or value
```

Example

A simple Python function to check whether x is even or odd

```
def evenOdd(x):
    if (x % 2 == 0):
        print("even")
    else:
        print("odd")
# Driver code to call the function
evenOdd(2)
evenOdd(3)
```



API Gateway Refresher

(from my Udemy Course on :

AWS Lambda, Python(Boto3) & Serverless-

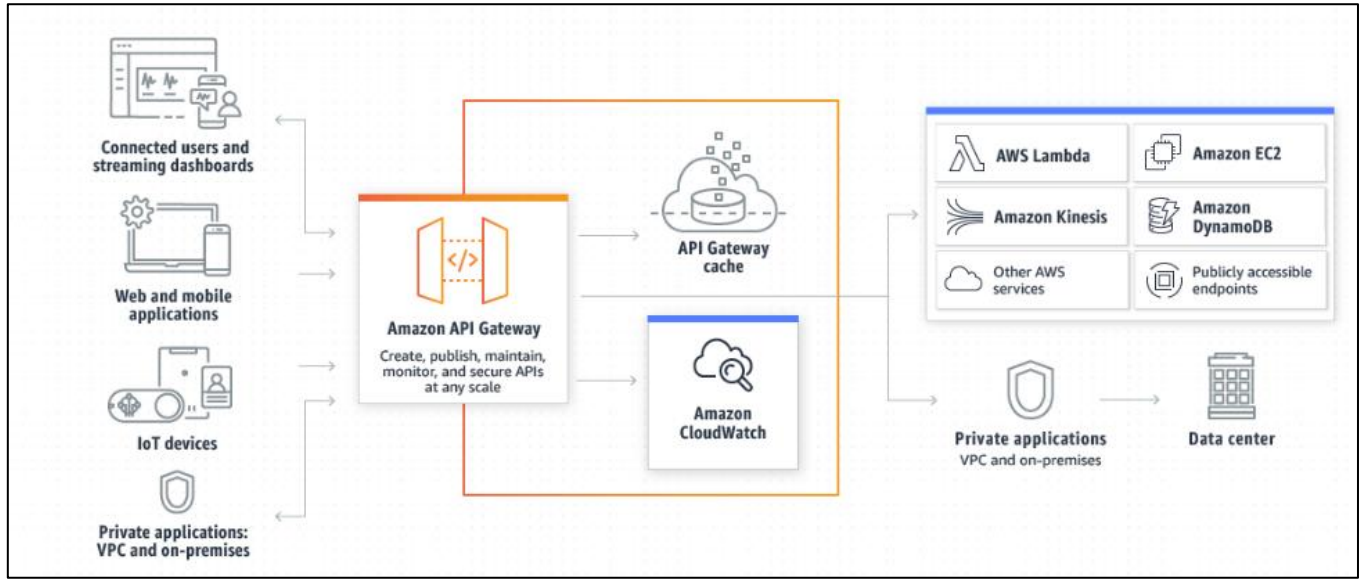
Beginner to Advanced)

AWS API Gateway - Overview

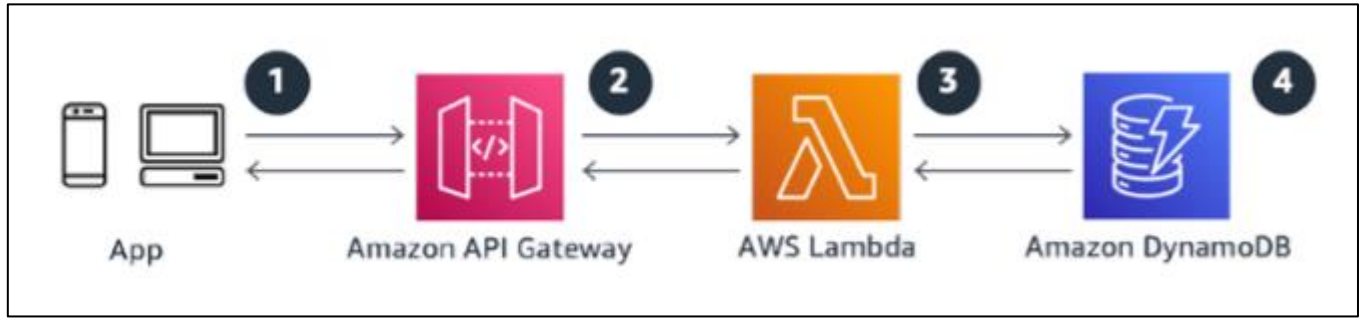
What is API Gateway

Amazon API Gateway is an AWS service for **creating, publishing, maintaining, monitoring, and securing REST,**

HTTP, and WebSocket APIs at any scale.



Sample Architecture - RESTful microservices



AWS API Gateway - Overview

Key Features Of API Gateway

1. API Types

1. HTTP API

Build **low-latency and cost-effective REST APIs** with built-in features such as OIDC and OAuth2, and native CORS support.

2. WebSocket API

Build a WebSocket API using **persistent connections for real-time use cases such as chat applications or dashboards.**

3. REST API 

Develop a REST API where you gain complete control over the request and response along with **API management capabilities.**

4. REST API (Private)

Create a REST API that is **only accessible from within a VPC**.

https://aws.amazon.com/blogs/compute/introducing-amazon-api-gateway-private-endpoints/

AWS API Gateway - Overview

Key Features of API Gateway

REST API's vs HTTP API's

- REST APIs support **more features** than HTTP APIs, while HTTP APIs are designed with **minimal features** and offered at a **lower price**.
- Choose **REST APIs** if you need features such as **API keys, per-client throttling, request validation, AWS WAF integration, or private API endpoints.**
- Choose **HTTP APIs** if you don't need the features included with **REST APIs.**

- https://aws.amazon.com/blogs/compute/introducing-amazon-api-gateway-private-endpoints/

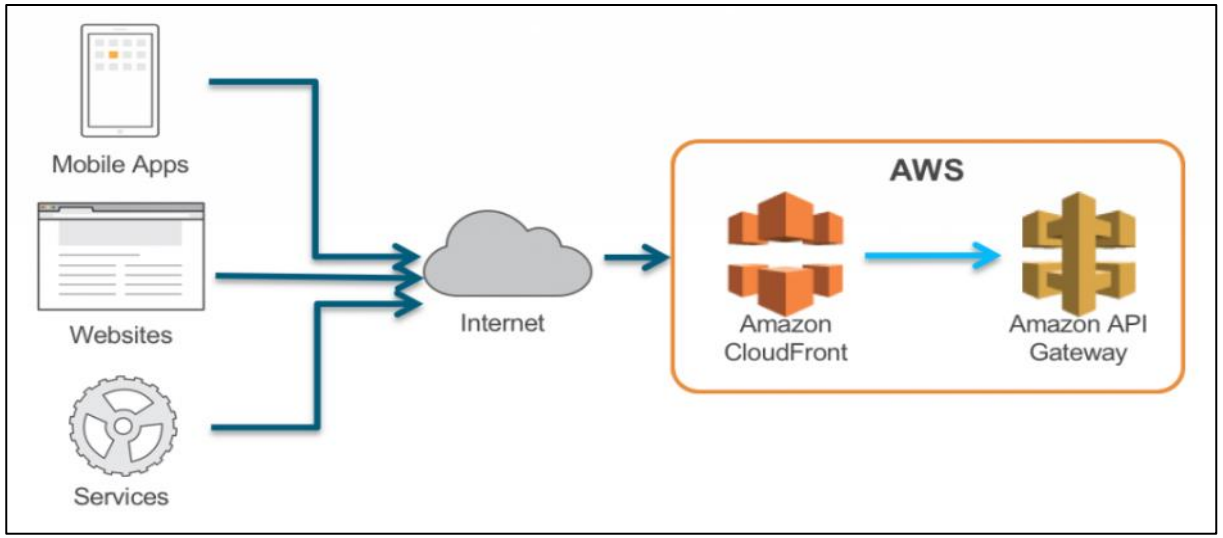
API Gateway - Overview

2. API Endpoint Types

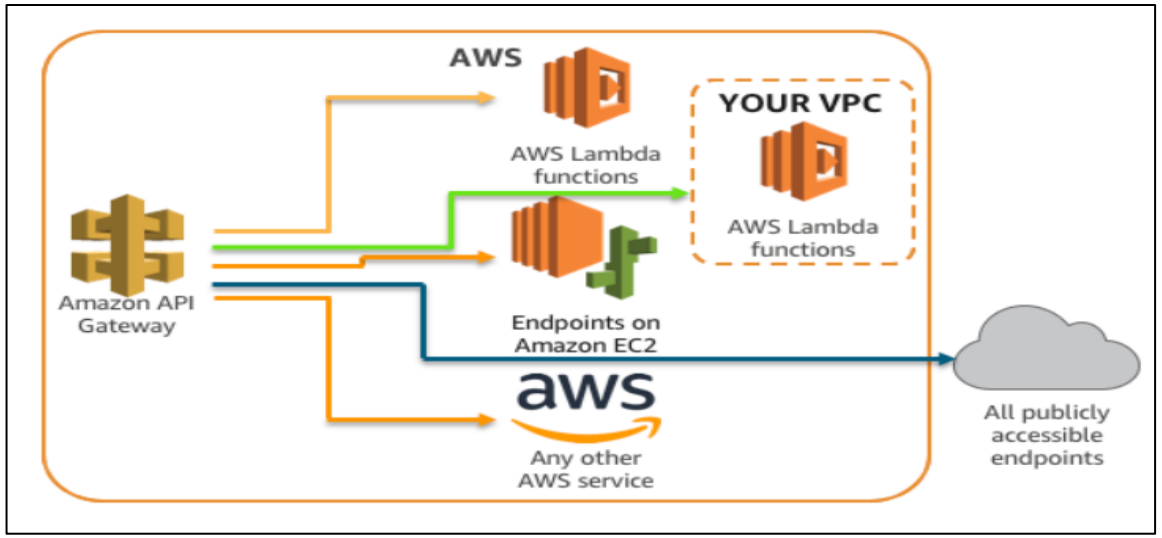
An ***API endpoint* type** refers to the **hostname of the API**.

The API endpoint type can be ***edge-optimized*, *regional*, or *private*,** depending on where the majority of your API traffic originates from.

Edge Optimized



Regional

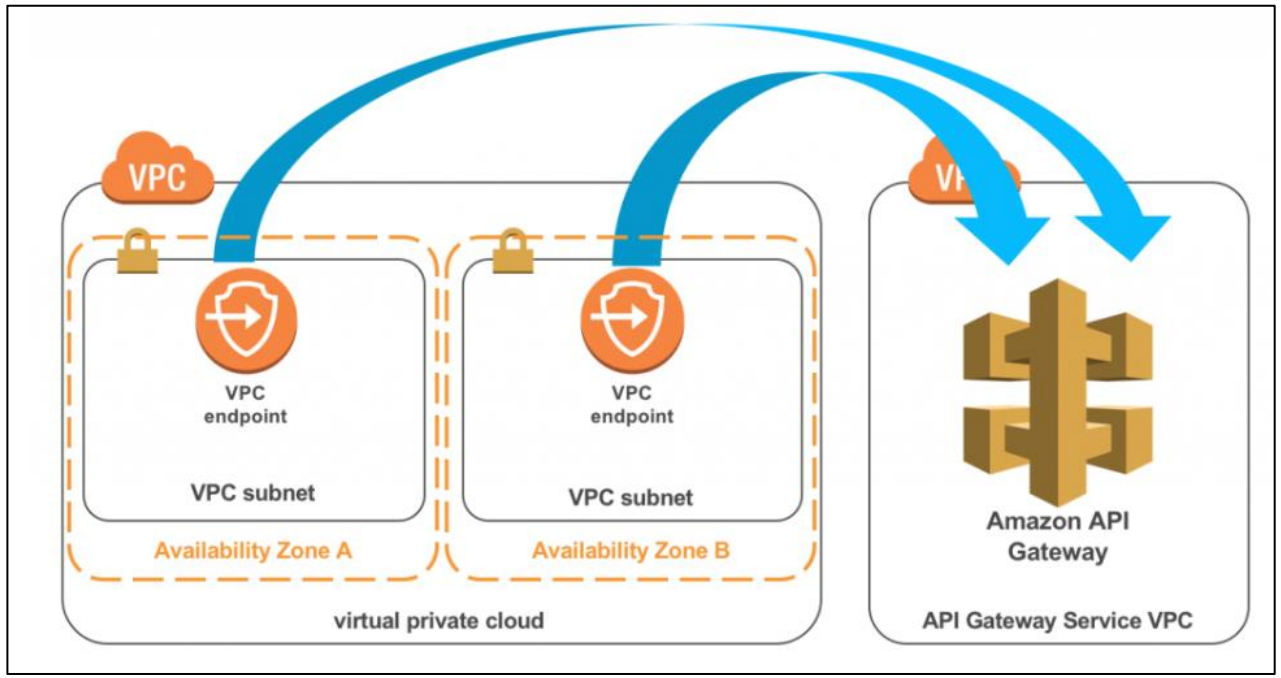


Source : AWS

API Gateway - Overview

2. API Endpoint Types

Private



Source : AWS

API Gateway - Overview

3. Resources

REST architecture treats every content as a **resource**.

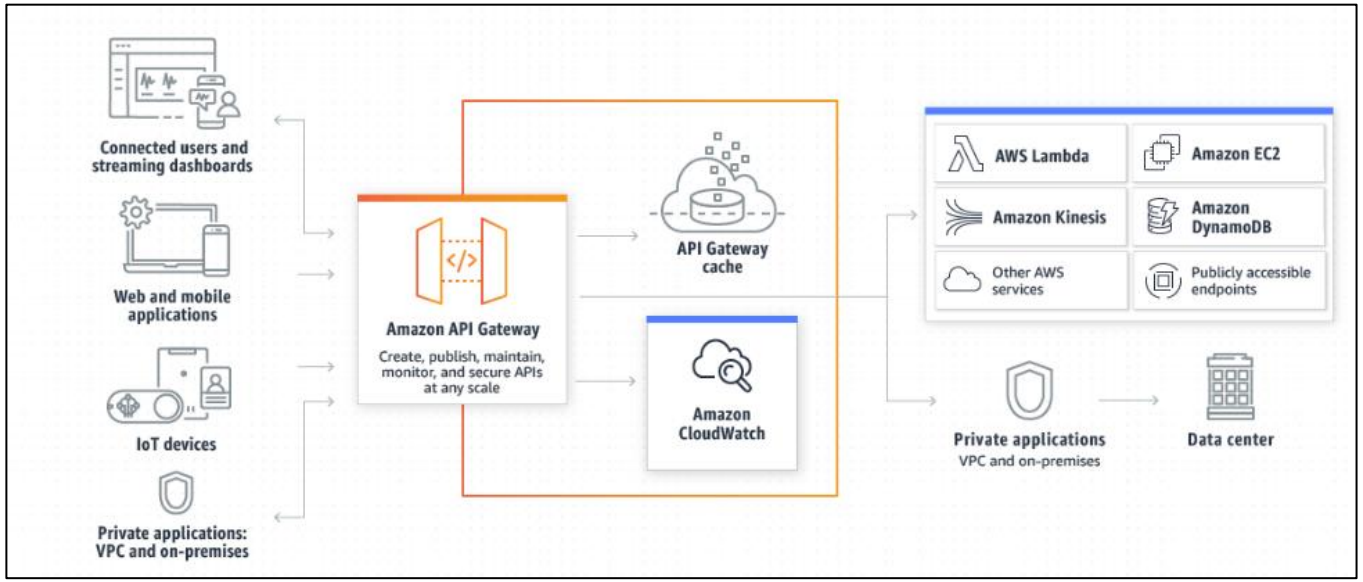
These resources can be Text Files, Html Pages, Images, Videos or Business Data.

4. Methods

Method	Description
GET	Retrieve information about the REST API resource
POST	Create a REST API resource
PUT	Update a REST API resource
DELETE	Delete a REST API resource or related component

API Gateway - Overview

5. Integration Type



Lambda Function

This type of integration lets an API method

be integrated with the Lambda function

HTTP

- Allows integration with existing HTTP endpoint
- This type of integration lets an API expose HTTP endpoints in the backend.

Mock

- This type of integration lets API Gateway return a response without sending the request further to the backend.

API Gateway - Overview

5. Integration Type

AWS Service

This type of integration lets an API expose AWS service actions.

VPC Link

A VPC link is a resource in Amazon API Gateway that allows for connecting API routes to private resources inside a VPC

API Gateway - Overview

5. Deployment

After **creating your API**, you must **deploy it** to make it callable by your users.

To **deploy an API**, you **create an API deployment** and **associate it with a stage**.

Important Tip :

Every time you update an API, you must redeploy the API to an existing stage or to a new stage.

Updating an API includes modifying routes, methods, integrations, authorizers, and anything else other than stage settings.

6. Stages

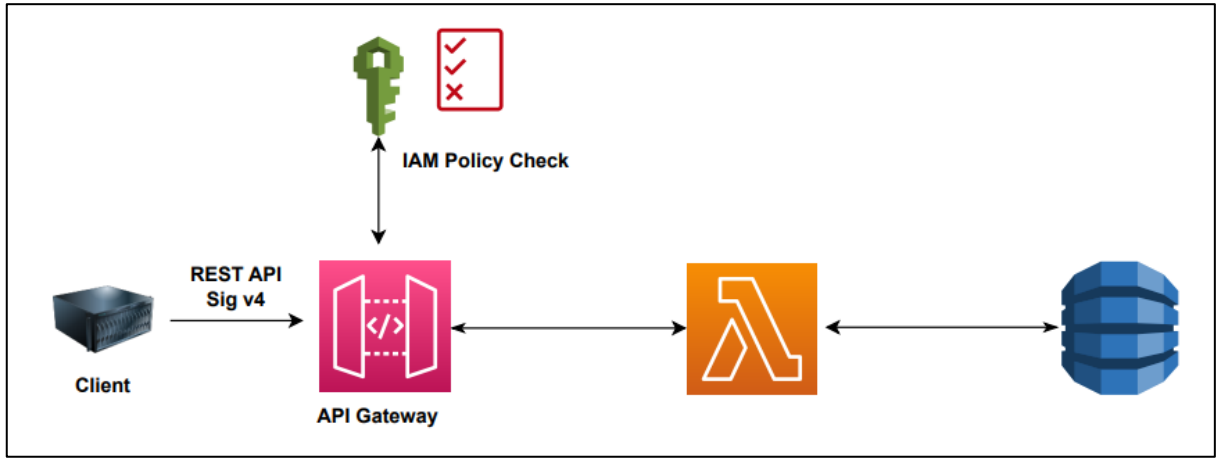
A logical reference to a **lifecycle state of your API** (for example, 'dev', 'prod', 'beta', 'v2').

https://{restapi-id}.execute-api.{region}.amazonaws.com/{stageName}

https://ent94mc14j.execute-api.us-east-1.amazonaws.com/dev/students

API Gateway - Overview

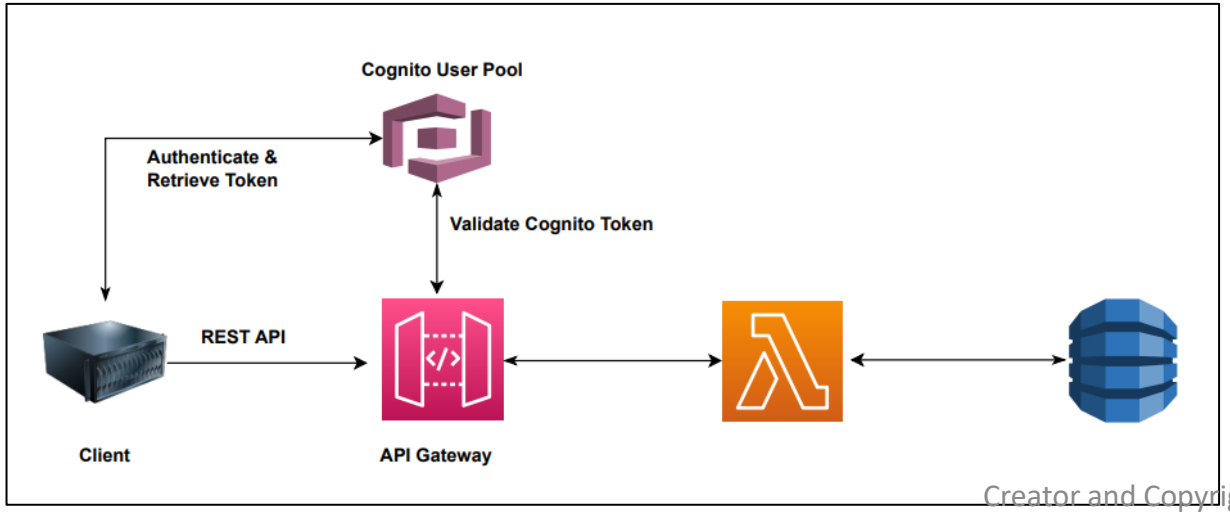
7. AWS API Gateway - Authentication and Authorization



1. API Security – IAM

Authentication – IAM
Authorization – IAM Policy

Use Case : Internal AWS Services, Cross Account Access



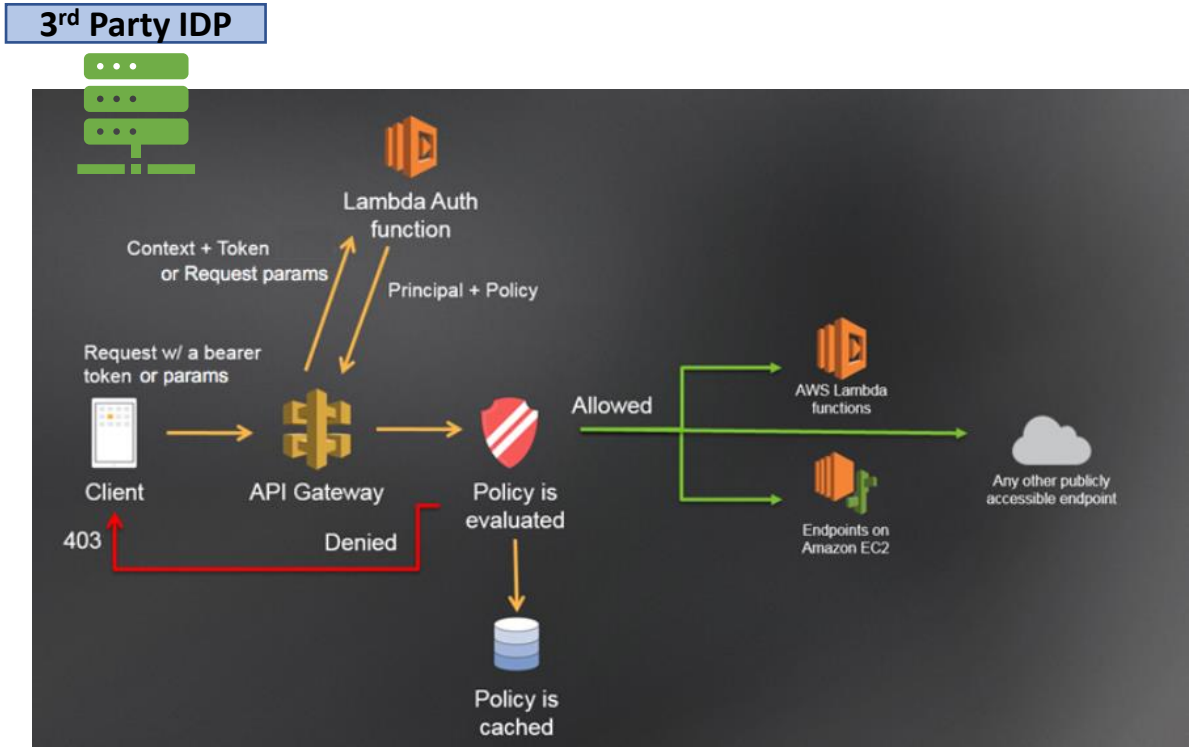
2. API Security – Cognito

Authentication – Cognito User Pool
Authorization – API Gateway Methods

Use Case : External Users for Web/Mobile Apps

API Gateway - Overview

7. AWS API Gateway - Authentication and Authorization



3. API Security – Lambda Authorizer

Authentication – External

Authorization – Lambda Function

Use Case : Third Party Identity Provider such as OAuth 2.0

https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-use-lambda-authorizer.html

API Gateway - Overview

8. Usage Plans

Usage Plans

Use Case : Differentiate between Basic and Premium Customers

Sets the **target request rate - Throttling, Burst and Quota Limit for each API Key**

9. API Keys

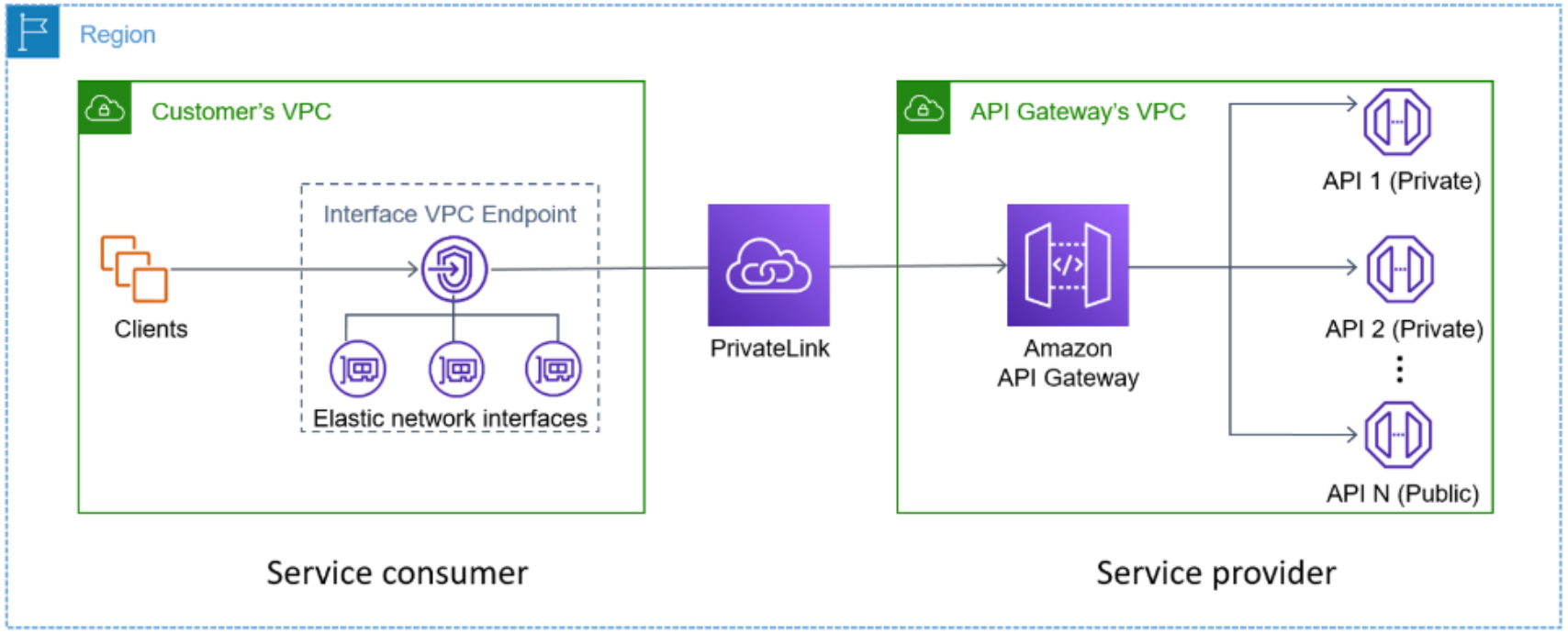
- An **alphanumeric string that API Gateway uses to identify an app developer** who uses your REST or WebSocket API.
- You can use API keys together with Lambda authorizers or usage plans to control access to your APIs.

API Gateway - Overview

10. API Integration and Private API's

Private API's

A private API means that the **API endpoint is reachable only through the VPC**. Private APIs are **accessible only** from **clients within the VPC** or from clients that have **network connectivity to the VPC**.



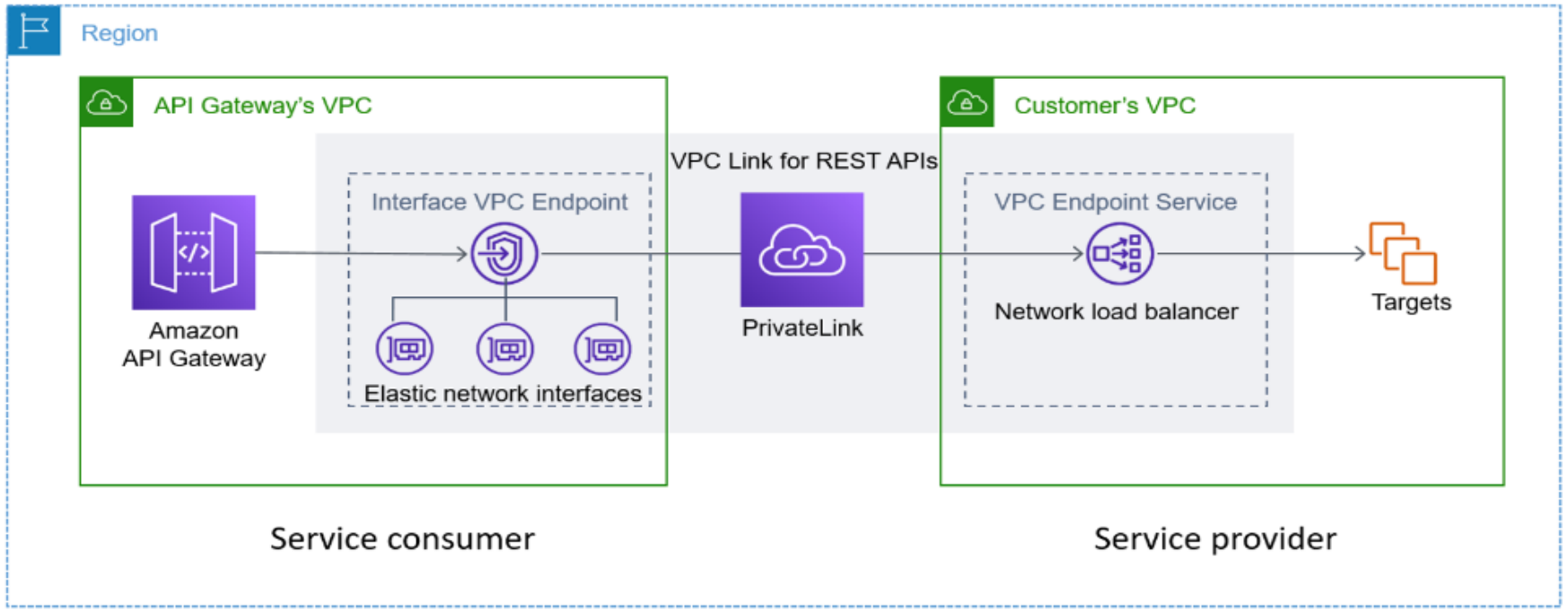
Source : AWS

API Gateway - Overview

10. API Integration, Private API's and VPC Link

Private Integration

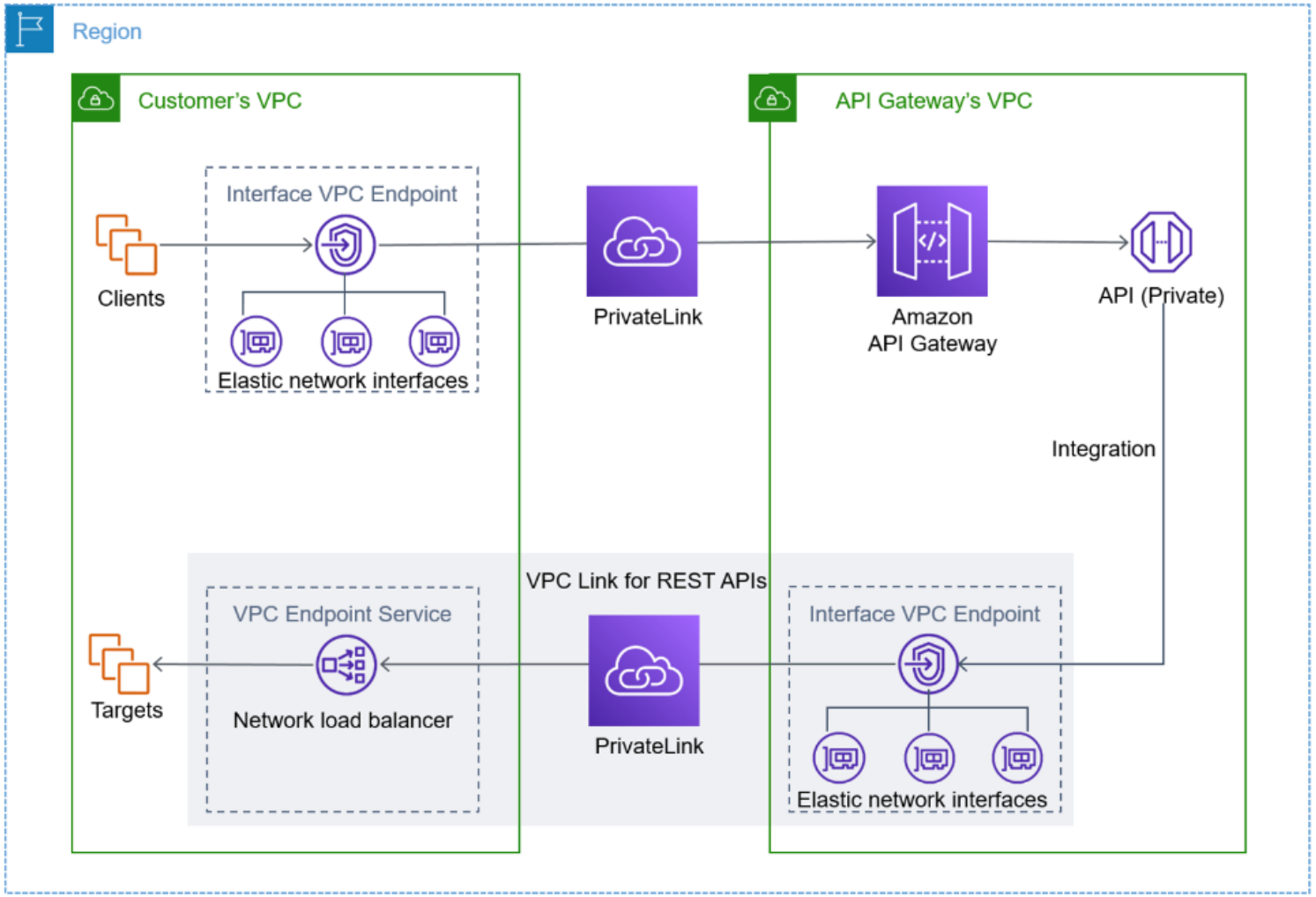
An API Gateway integration type for a client to access resources inside a customer's VPC through a private REST API endpoint without exposing the resources to the public internet.



Source : https://aws.amazon.com/blogs/compute/understanding-vpc-links-in-amazon-api-gateway-private-integrations/

API Gateway - Overview

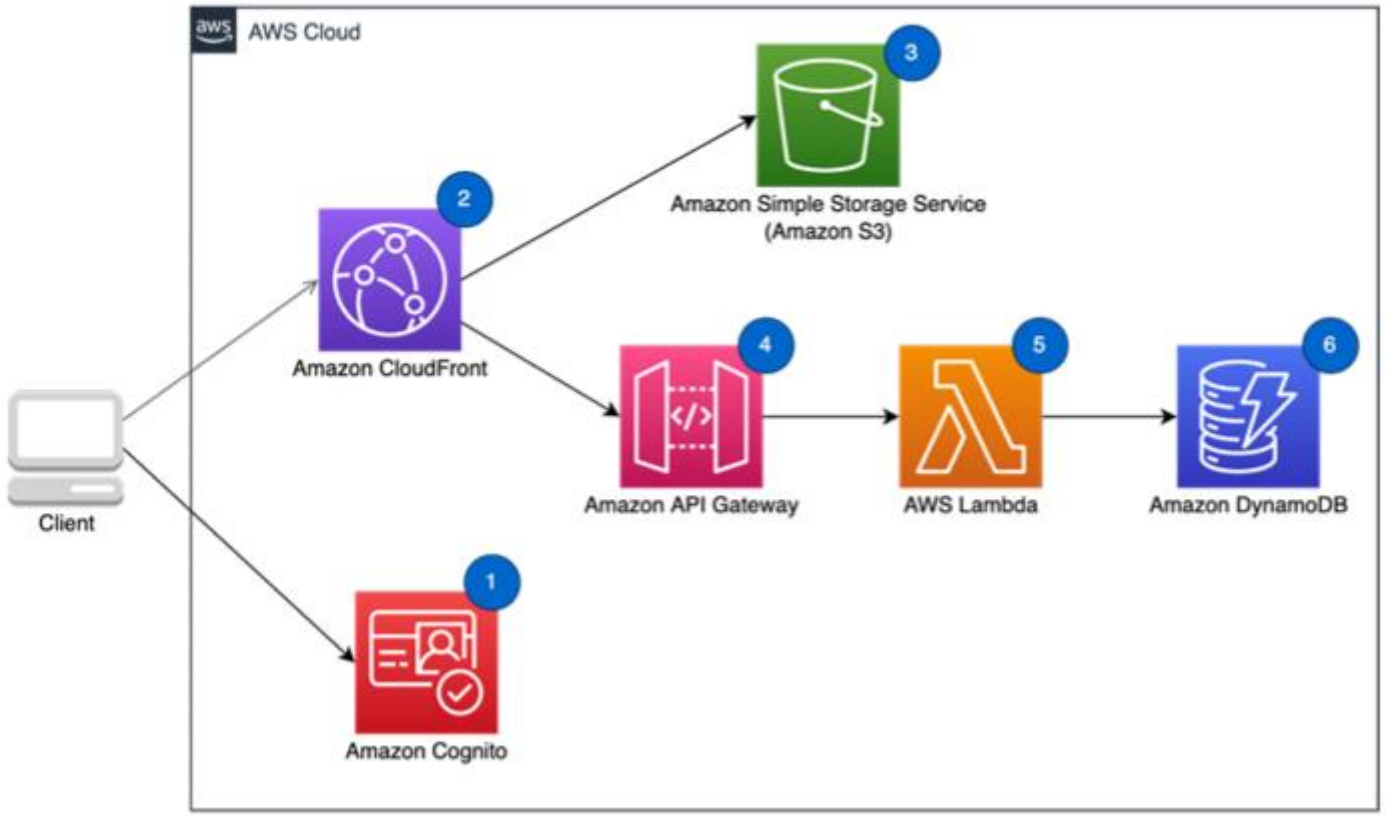
10. API Integration and Private API's



Source : AWS

API Gateway - Overview

Sample Architecture



Source : AWS

Thank You