

Artificial Intelligence Integration: Using Spring AI with OpenAI (ChatGPT) and AWS Bedrock to Deliver Efficient APIs



Integração de inteligência artificial: usando Spring AI com OpenAI (ChatGPT) e AWS Bedrock para fornecer APIs eficientes



About the Speaker

Weder Mariano de Sousa

Specialist Senior Java - GFT

Technician **System Development**

Graduated **Computer Science**

Post Graduate in **Midias UFG**

Post Graduate in **Information Security**



<https://www.linkedin.com/in/wedermarianodesousa/>

<https://github.com/weder96>

<https://twitter.com/weder96>

<https://dev.to/weder96>



GOJava



AWS User Group Goiânia



Agenda

Introduction

Topic 1: Generative AI – Concepts and Advantages

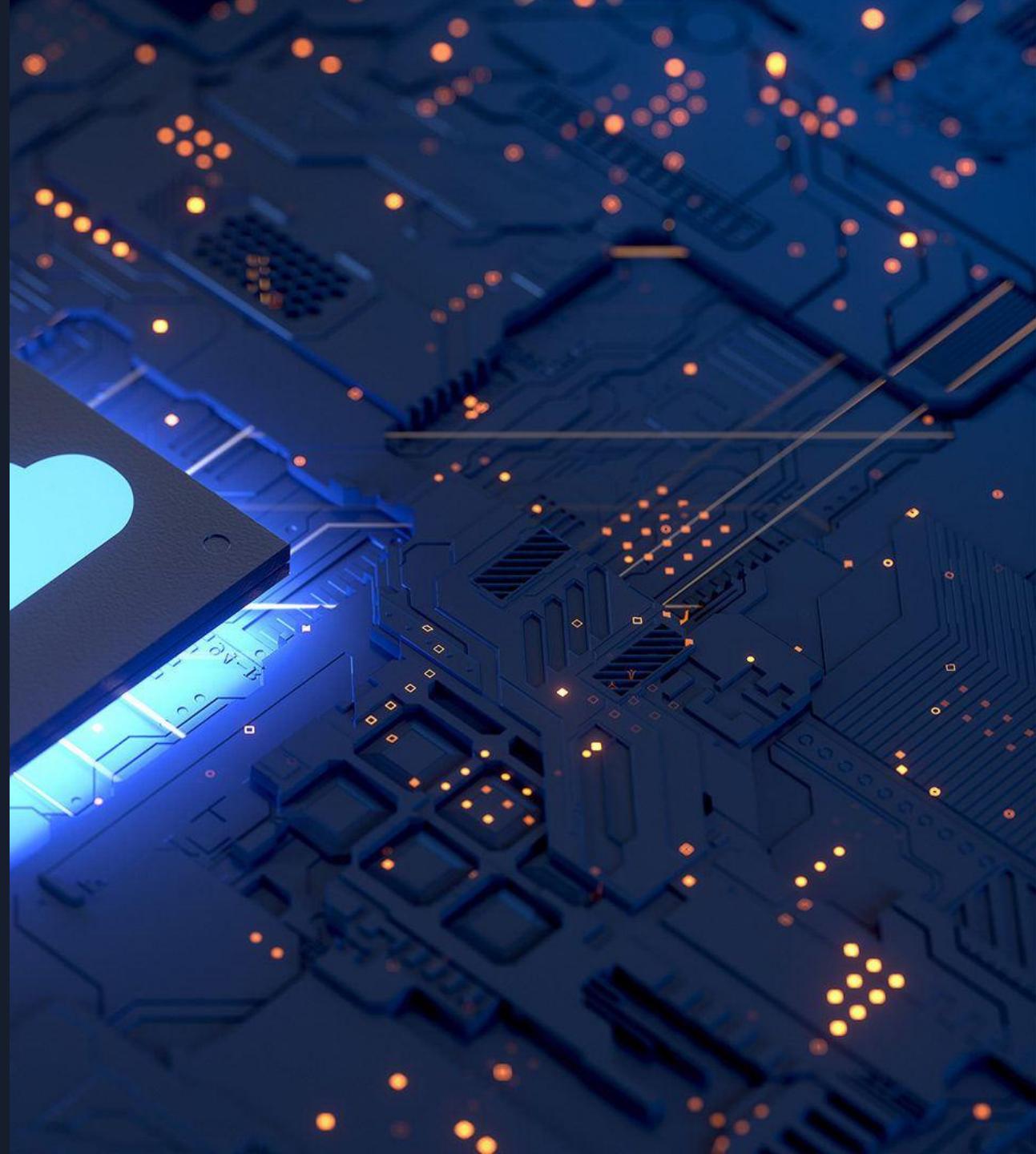
Topic 2: Getting to Know OpenAI and AWS Bedrock

**Topic 3: Integration: Spring AI with OpenAI
(ChatGPT)**

Topic 4: Integration: Spring AI with AWS (Bedrock)

**Topic 5: Presentation of the integration architecture
AWS Bedrock**

Conclusion

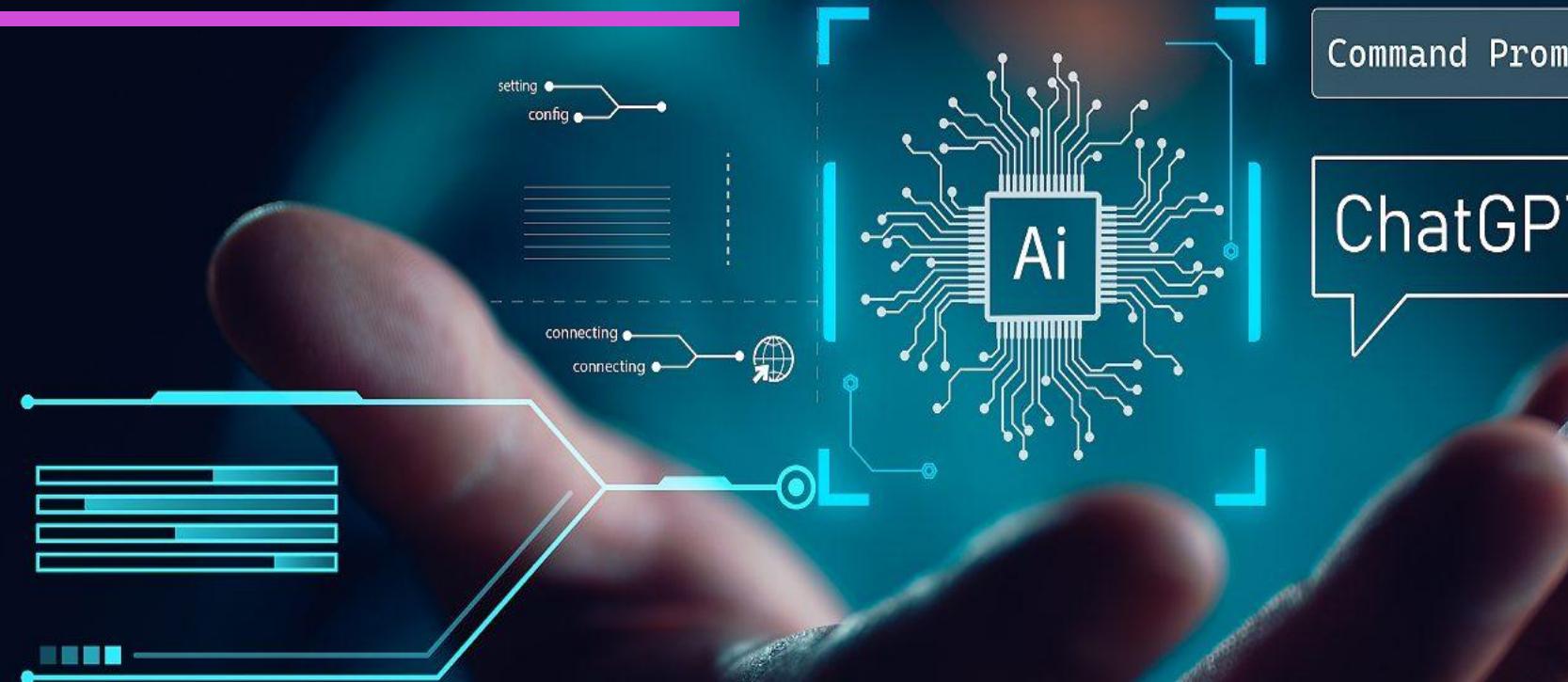


Generative AI

Topic 1: Generative AI – Concepts and Advantages



Generative AI



What is Generative AI

Image



Generated with OpenAI Image Generator G1

Create an image of a puppy
eating a popsicle

Text/Chat

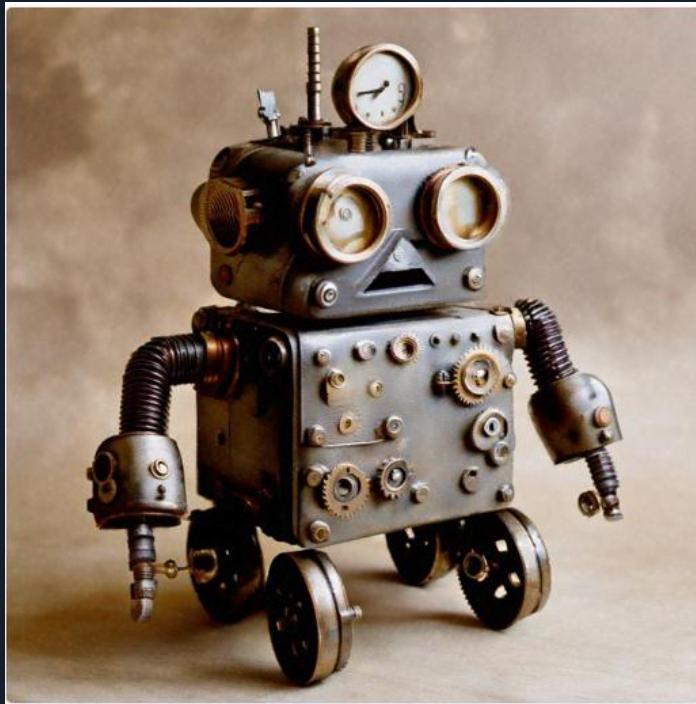
> Question: What is generative AI?

Generative AI refers to a type of artificial intelligence that is capable of creating new and original content. It uses algorithms and machine learning to generate text, images, music, and other forms of media based on patterns and rules learned from existing data.

Generated with OpenAI Image Generator G1

What is Generative AI

Image



Generated with Amazon Titan Image Generator G1

A stylized picture of a cute old steampunk robot

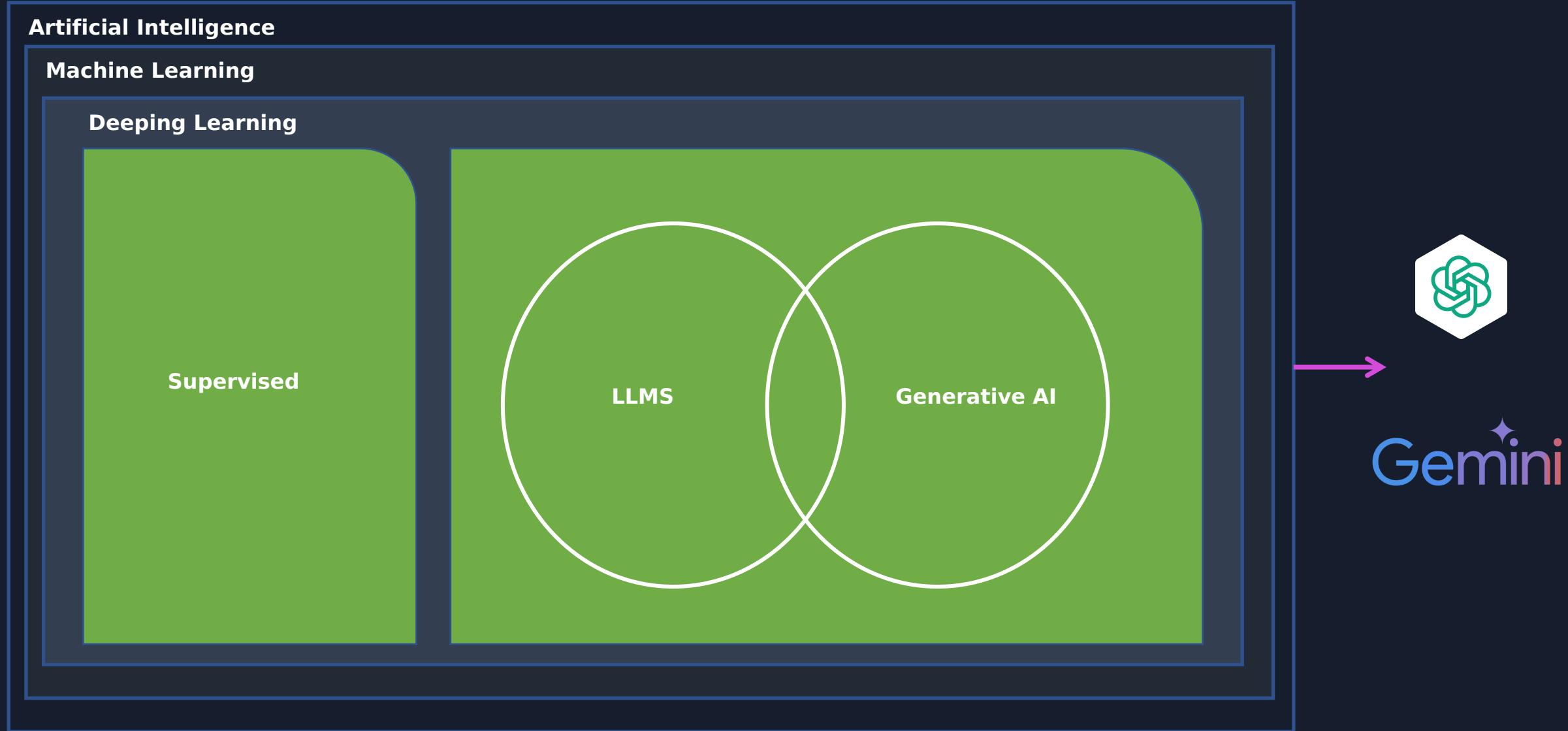
Text/Chat

> **Question: What is generative AI?**

Generative AI refers to a type of artificial intelligence that is capable of creating new and original content. It uses algorithms and machine learning to generate text, images, music, and other forms of media based on patterns and rules learned from existing data.

Generated with Amazon Titan Text G1 - Express

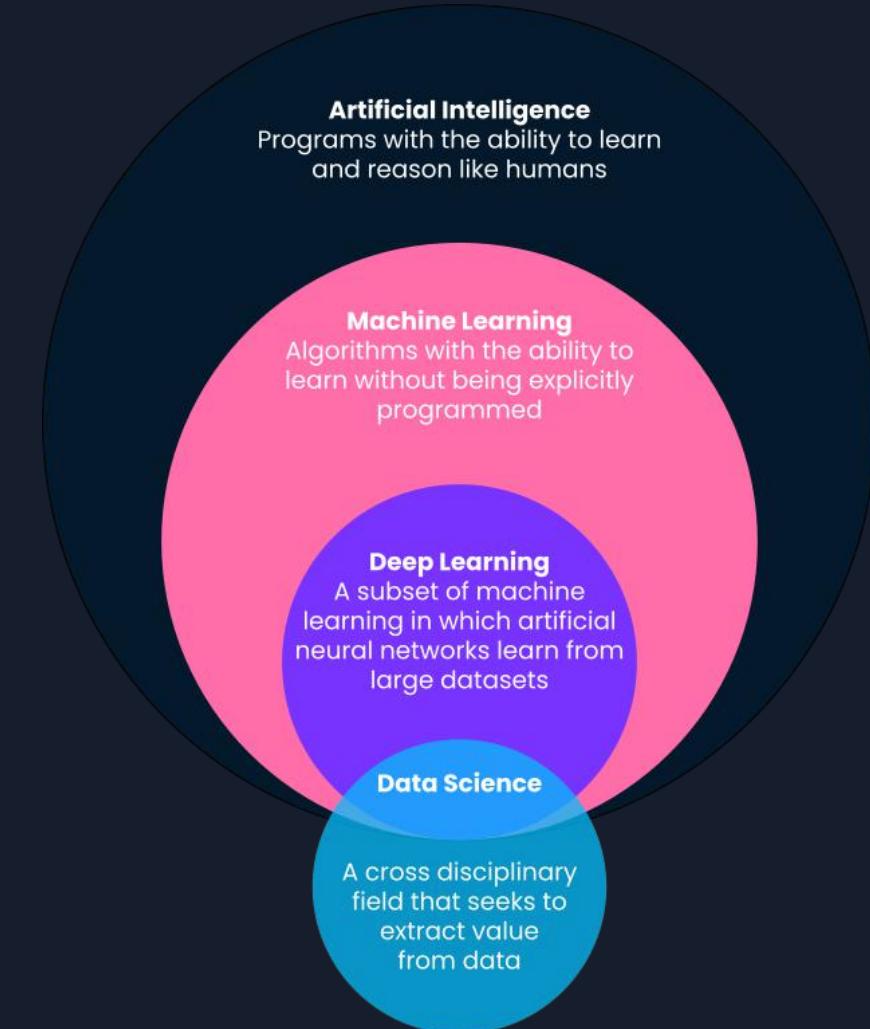
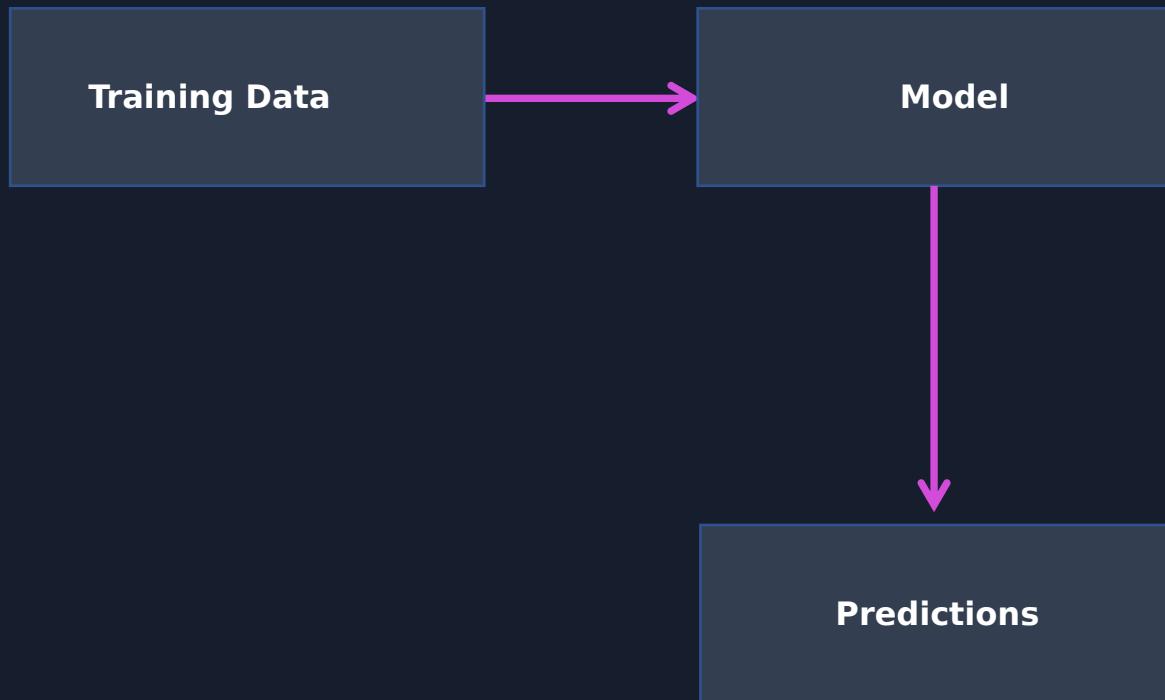
What is Generative AI



Machine Learning



What is machine Learning?

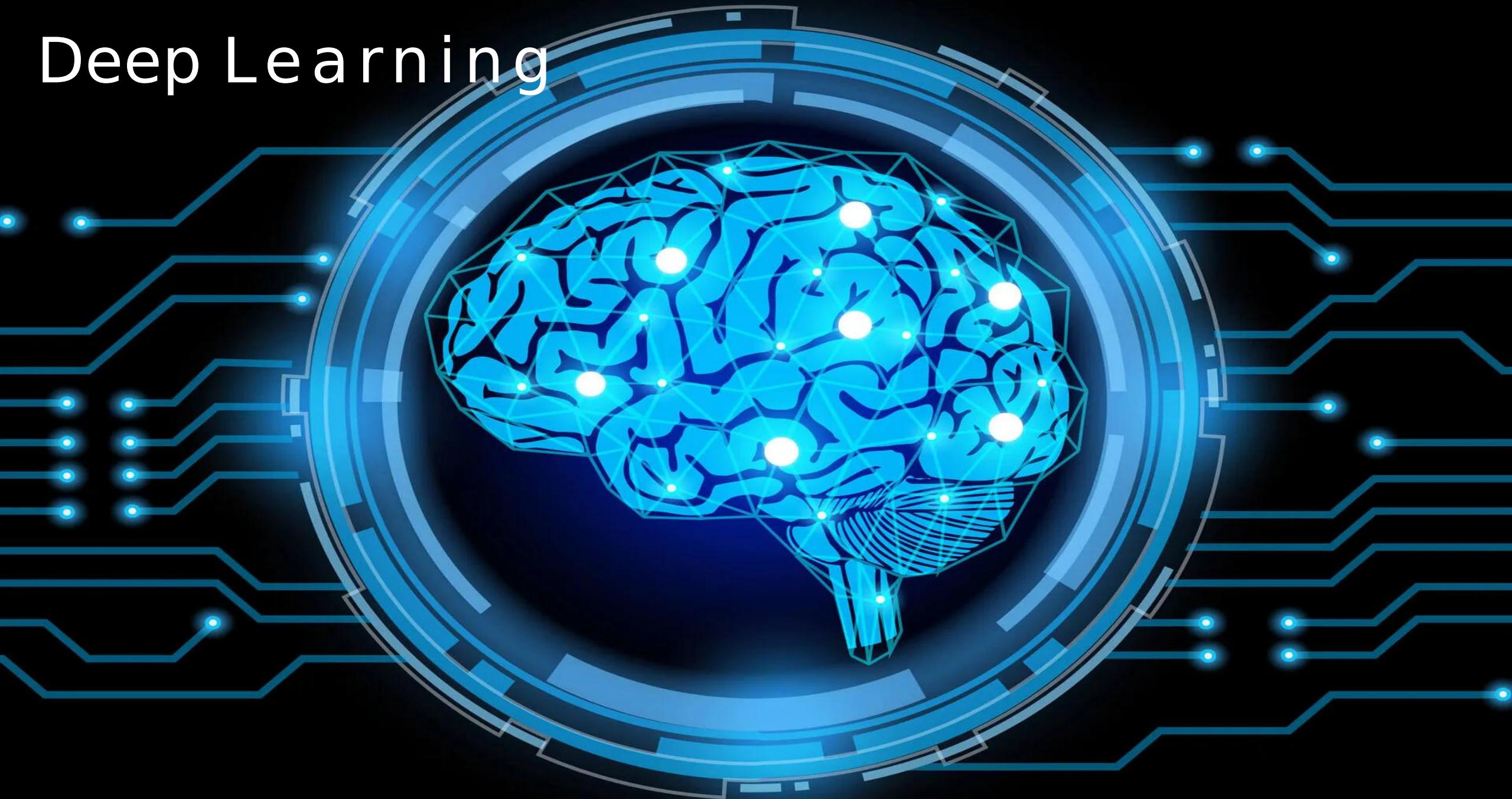


Machine Learning

Use Cases

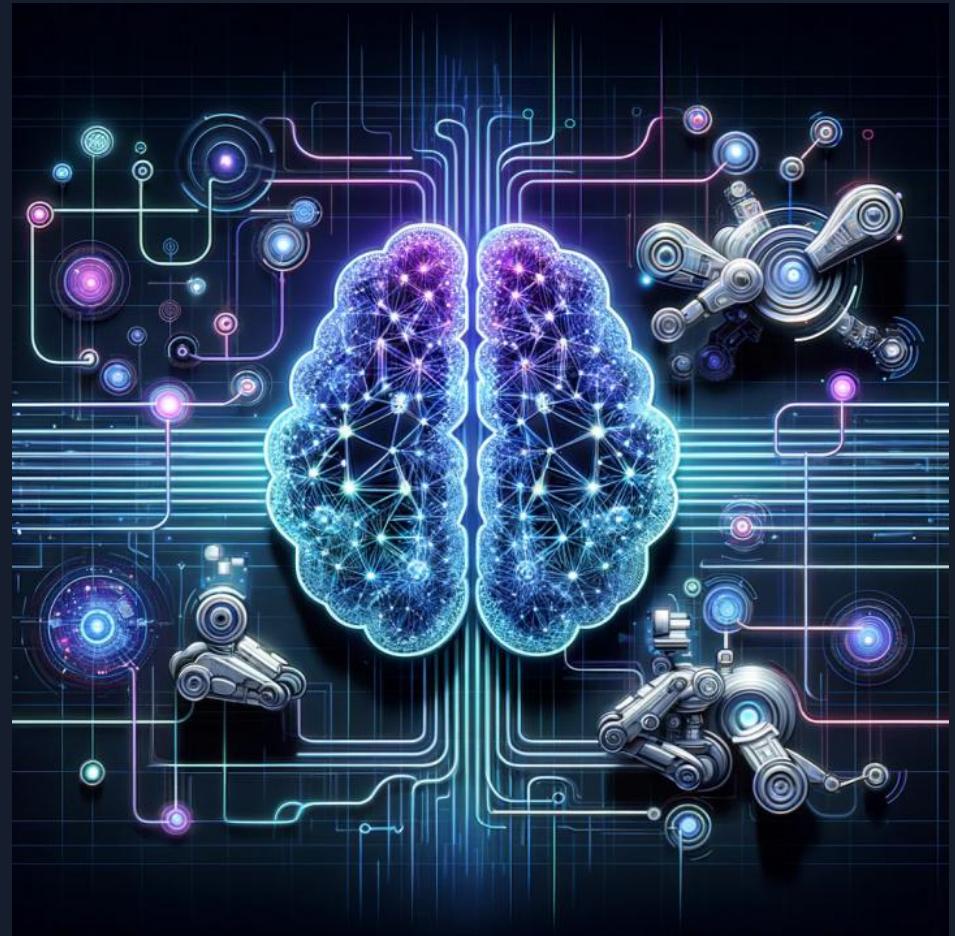
- **Facial Recognition**
- **Recognize Tumors on x-ray scans**
- **Abnormality on ultrasounds**
- **Self-driving mode (recognize stop sign / pedestrian / etc)**
- **Fraud detection**
- **Product Recommendations (YouTube)**
- **Spam Filtering**

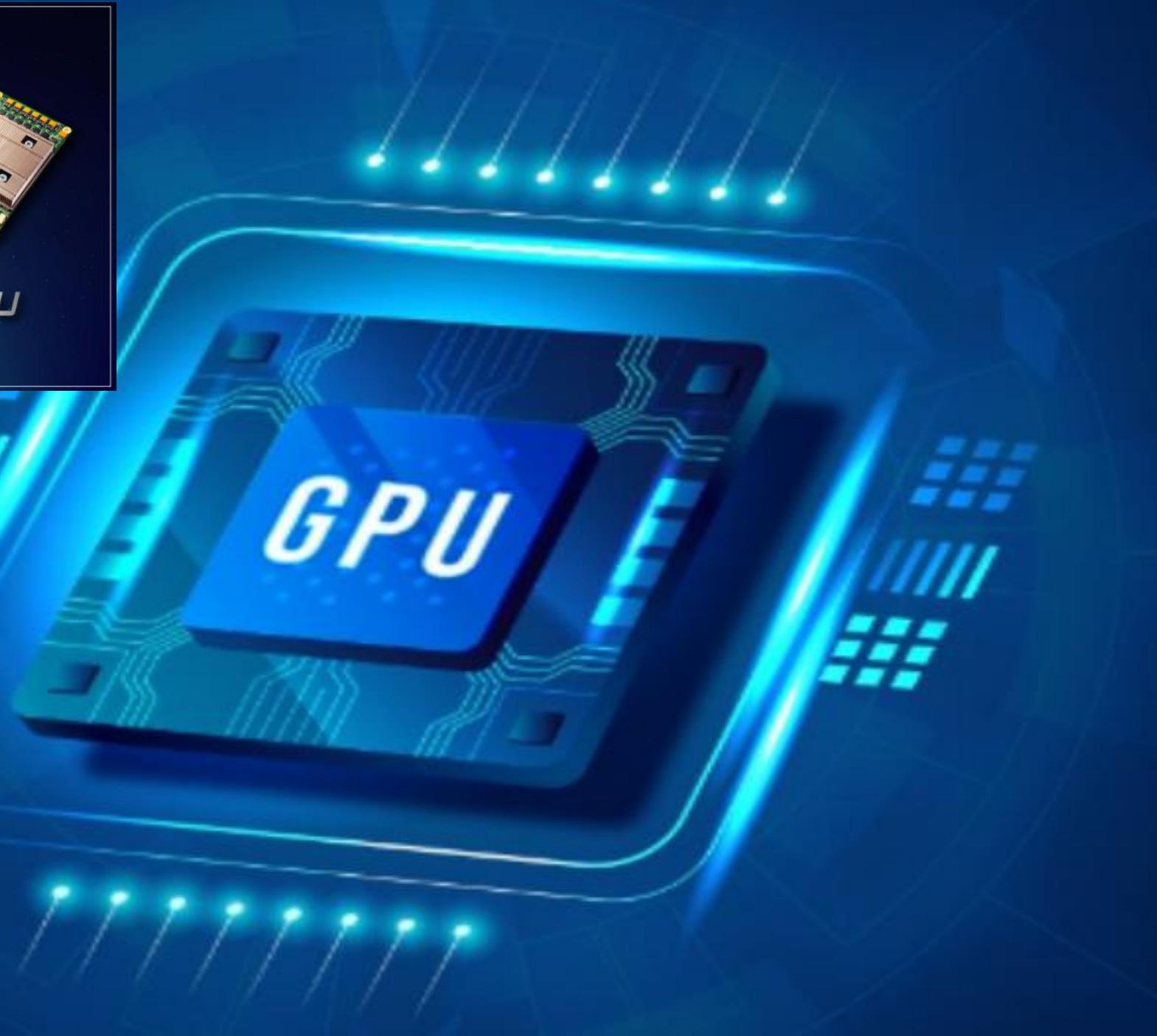
Deep Learning



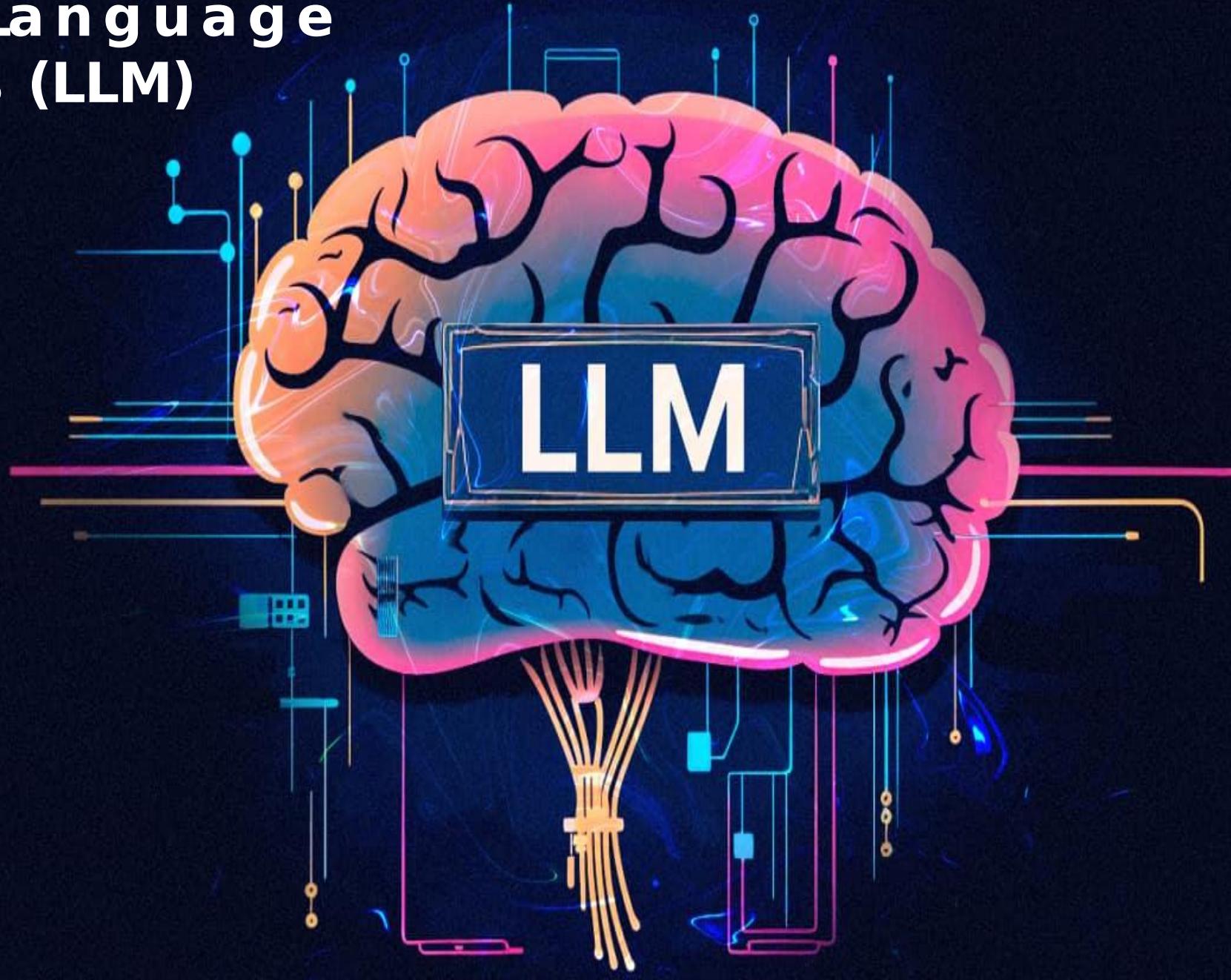
Artificial Neural Network

- 1. Scientific Advances - Deep Learning**
- 2. Availability of Big Data (You need data to configure these neural networks)**
- 3. Lots of Compute Power**





Large Language Models (LLM)



Large Language Models

So what are LLMs

LLMs are a type of artificial intelligence that can generate text, understand language, answer questions and more.

They are large because they have a vast number of parameters, which are the parts of the model that are learned from data during training.

ChatGPT

- 1. 175 Billion Parameters**
- 2. Training Data - Hundreds of Billions of words**



Describe your qualities to me explaining to a developer ?



GPT-4o Mini

Sure! Here are some key qualities I can describe, specifically in the context of a developer:

Access to Information: I have access to a wide range of information and knowledge...

Troubleshooting: I am able to help identify and resolve software issues by providing diagnostics...

Multi-Language Support: I can provide support for a variety of programming languages...

Documentation and Reference: I have the ability to generate and explain documentation for APIs...

Continuous Learning: I am always up to date with my latest knowledge...

Project Management Assistance: In addition to helping with the actual coding...

Personalization of Answers: I can tailor my answers to the developer's level of knowledge...

Tool and Libraries Suggestions: I can recommend tools, libraries...

These qualities make me a valuable resource for developers looking for quick information, support, and solutions to their coding challenges.

Attention is all you need

Bigger is Better

Very Large Neural Networks

Vast Amounts of Training Data

Huge Compute Power to Train Data

General Purpose AI

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
nom@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

1 Introduction

Recurrent neural networks, long short-term memory [12] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [29, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [31, 21, 13].

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

Attention is all you need

Transformer Architecture

Specialized architecture for token prediction

Key Innovation

Attention mechanisms

Not Just a big neural network

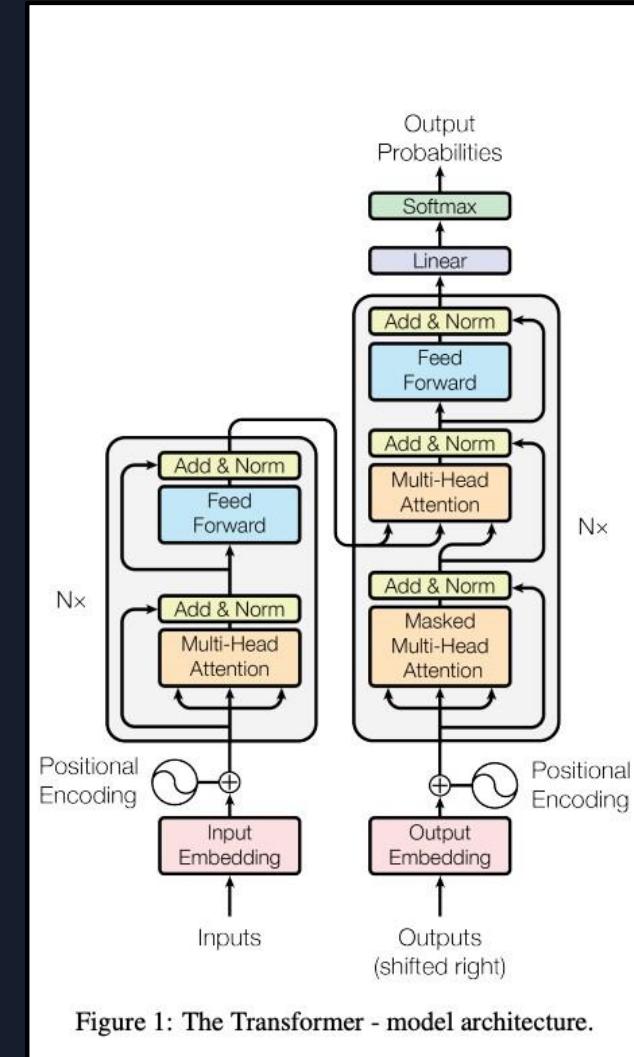
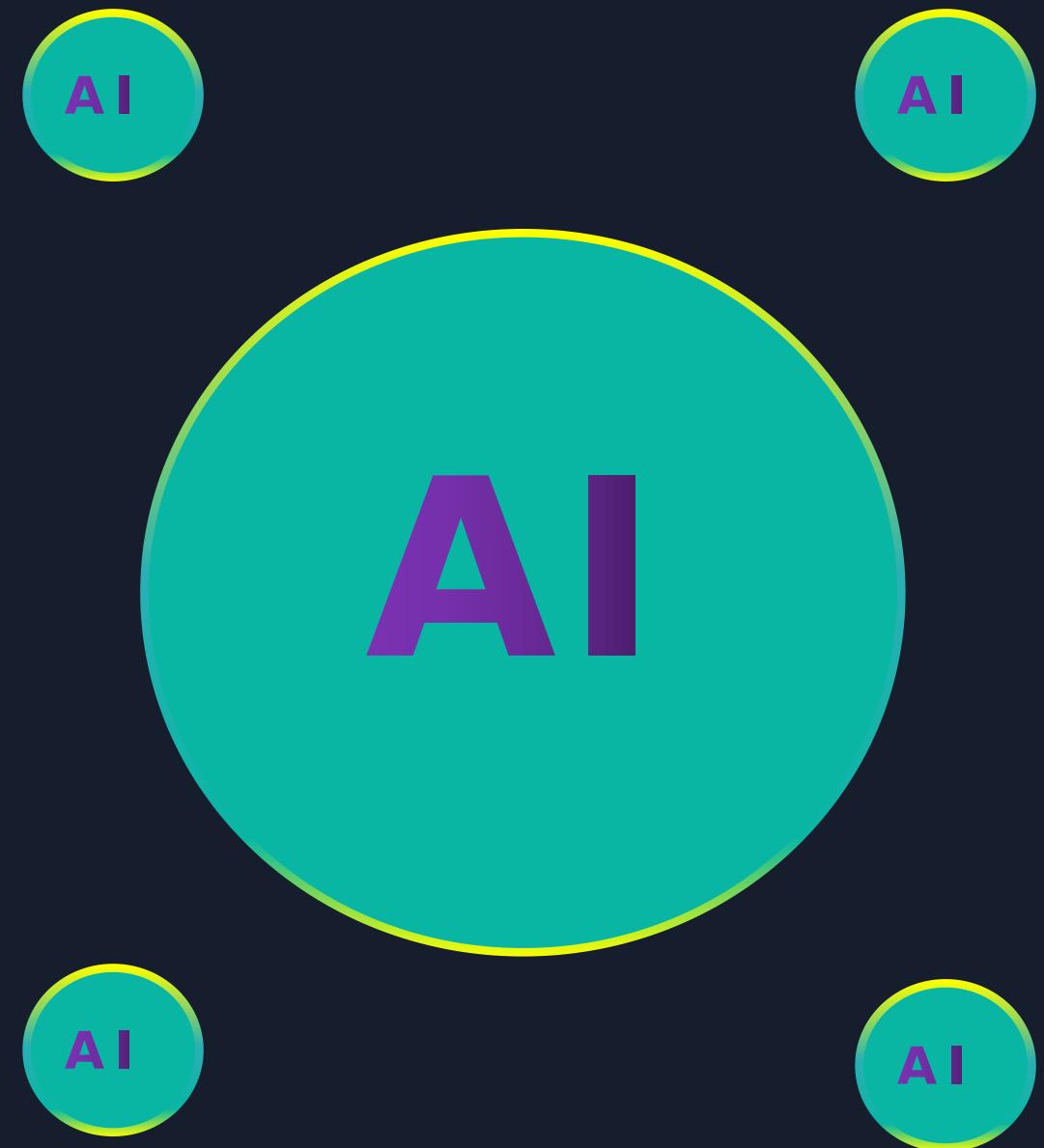


Figure 1: The Transformer - model architecture.

WHAT ARE AI MODELS?



As Artificial Intelligence (AI) models become more important and widespread in almost every sector, it is increasingly important for businesses to understand how these artificial intelligence models work and the potential implications of using them.



AI models and their various applications. We will explore popular use cases of AI models and discuss how they can be used effectively in business and other settings.



Tasks Libraries Datasets Languages Licenses Other

Filter Tasks by name

Multimodal

 Image-Text-to-Text Visual Question Answering
 Document Question Answering Video-Text-to-Text
 Any-to-Any

Computer Vision

 Depth Estimation Image Classification
 Object Detection Image Segmentation
 Text-to-Image Image-to-Text Image-to-Image
 Image-to-Video Unconditional Image Generation
 Video Classification Text-to-Video
 Zero-Shot Image Classification Mask Generation
 Zero-Shot Object Detection Text-to-3D
 Image-to-3D Image Feature Extraction
 Keypoint Detection

Natural Language Processing

 Text Classification Token Classification
 Table Question Answering Question Answering
 Zero-Shot Classification Translation
 Summarization Feature Extraction
 Text Generation Text2Text Generation

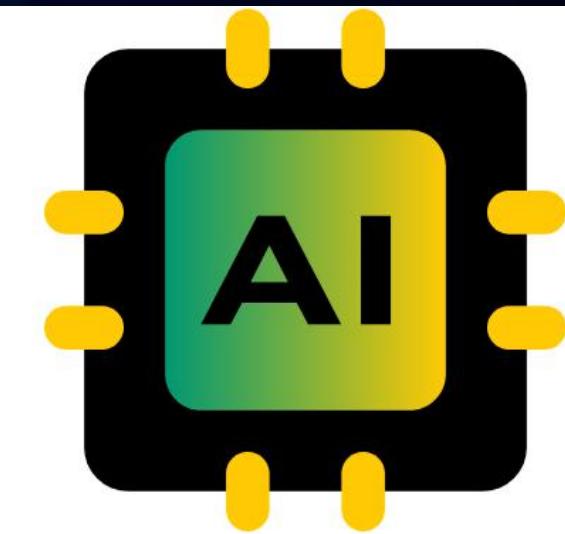
Models 974,286

Filter by name

Full-text search

Sort: Trending

 stepfun-ai/GOT-OCR2_0
Image-Text-to-Text • Updated 3 days ago • ↓121k • ❤ 417 black-forest-labs/FLUX.1-dev
Text-to-Image • Updated Aug 16 • ↓1M • ⚡ • ❤ 4 mistralai/Mistral-Small-Instruct-2409
Updated 1 day ago • ↓2.51k • ❤ 200 jinaai/reader-lm-1.5b
Text Generation • Updated about 6 hours ago • ↓ jinaai/jina-embeddings-v3
Feature Extraction • Updated 1 day ago • ↓8.87k • ❤ 152 mistralai/Pixtral-12B-2409
Updated 3 days ago • ↓9 • ❤ 261 upstage/solar-pro-preview-instruct
Text Generation • Updated about 17 hours ago • ↓3.06k • ❤ 365 meta-llama/Meta-Llama-3.1-8B-Instruct
Text Generation • Updated Aug 20 • ↓4.11M • ⚡ • ❤ 2.53k Qwen/Qwen2.5-72B-Instruct
Text Generation • Updated 1 day ago • ↓2.74k • ⚡ • ❤ 110 ICTNLP/Llama-3.1-8B-Omni
Updated 7 days ago • ↓1.28k • ❤ 306 fishaudio/fish-speech-1.4
Text-to-Speech • Updated 1 day ago • ↓4.54k • ❤ 327 black-forest-labs/FLUX.1-schnell
Text-to-Image • Updated Aug 16 • ↓1.35M • ⚡ • ❤ 2.32k microsoft/GRIN-MoE
Text Generation • Updated about 22 hours ago • ↓174 • ❤ 92 iiiorrg/piiranha-v1-detect-personal-information
Token Classification • Updated 7 days ago • ↓4.7k • ❤ 108 kyutai/moshiko-pytorch-bf16
Updated 2 days ago • ❤ 82 Qwen/Qwen2.5-7B-Instruct
Text Generation • Updated 1 day ago • ↓4.48k • ❤ 77 mattshumer/Reflection-Llama-3.1-70B
Text Generation • Updated 14 days ago • ↓34.6k • ❤ 1.62k Qwen/Qwen2-VL-72B-Instruct
Image-Text-to-Text • Updated 2 days ago • ↓1.61k • ❤ 75<https://huggingface.co/models>



Spring

WHAT IS SPRING AI



spring[®]



Spring AI

AI for Spring Developers

A new Spring Project

Mark Pollack

Current Version 1.0.0-M2

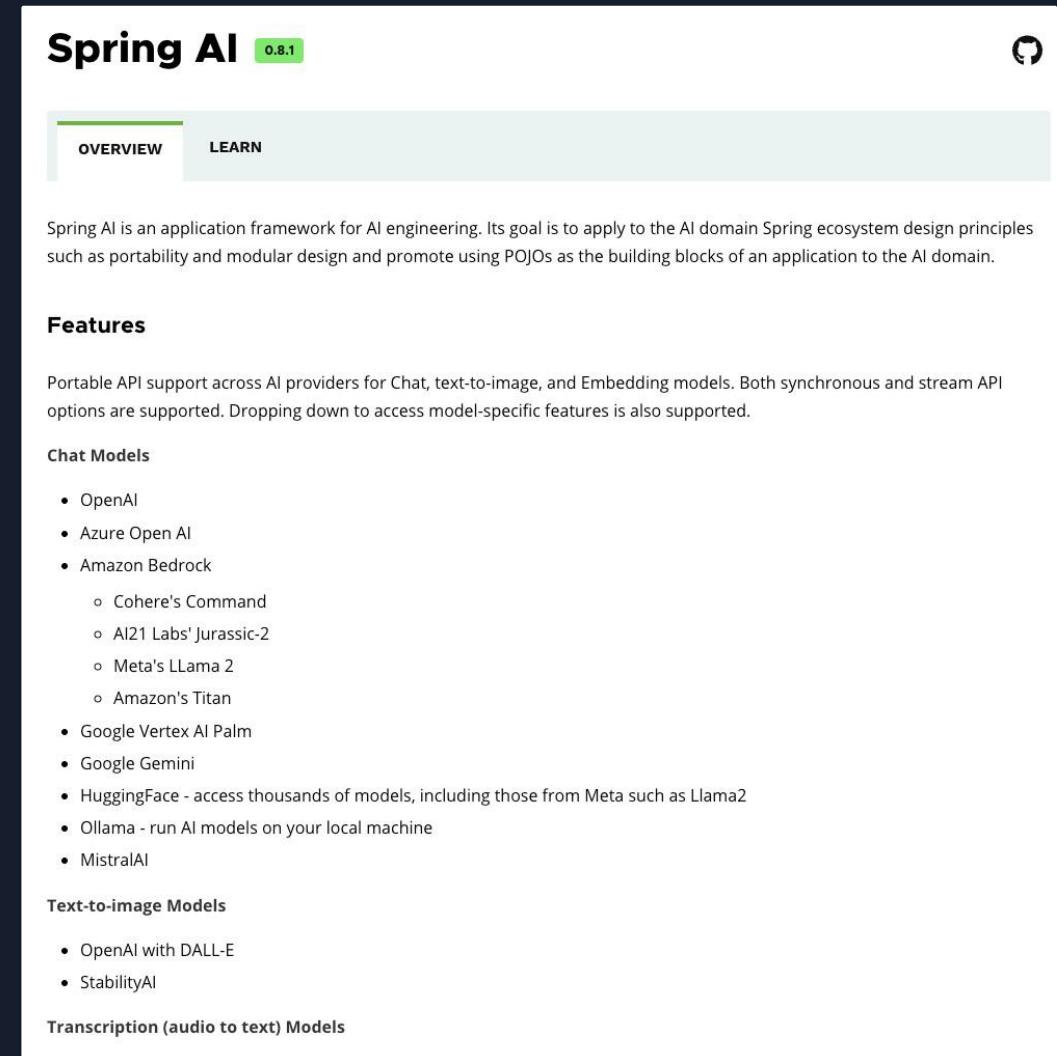
<https://spring.io/projects/spring-ai>

Inspired by Python projects

LangChain

Llmalndex

<https://docs.spring.io/spring-ai/reference/index.html>



The screenshot shows the official project page for Spring AI on the Spring website. The header features the "Spring AI" logo with a version "0.8.1" badge. Below the header are two tabs: "OVERVIEW" (which is active) and "LEARN". The main content area starts with a brief description: "Spring AI is an application framework for AI engineering. Its goal is to apply to the AI domain Spring ecosystem design principles such as portability and modular design and promote using POJOs as the building blocks of an application to the AI domain." This is followed by a "Features" section, which includes a list of supported AI providers under the "Chat Models" heading. The "Text-to-image Models" and "Transcription (audio to text) Models" sections are also listed.

Spring AI 0.8.1

OVERVIEW **LEARN**

Spring AI is an application framework for AI engineering. Its goal is to apply to the AI domain Spring ecosystem design principles such as portability and modular design and promote using POJOs as the building blocks of an application to the AI domain.

Features

Portable API support across AI providers for Chat, text-to-image, and Embedding models. Both synchronous and stream API options are supported. Dropping down to access model-specific features is also supported.

Chat Models

- OpenAI
- Azure Open AI
- Amazon Bedrock
 - Cohere's Command
 - AI21 Labs' Jurassic-2
 - Meta's Llama 2
 - Amazon's Titan
- Google Vertex AI Palm
- Google Gemini
- HuggingFace - access thousands of models, including those from Meta such as Llama2
- Ollama - run AI models on your local machine
- MistralAI

Text-to-image Models

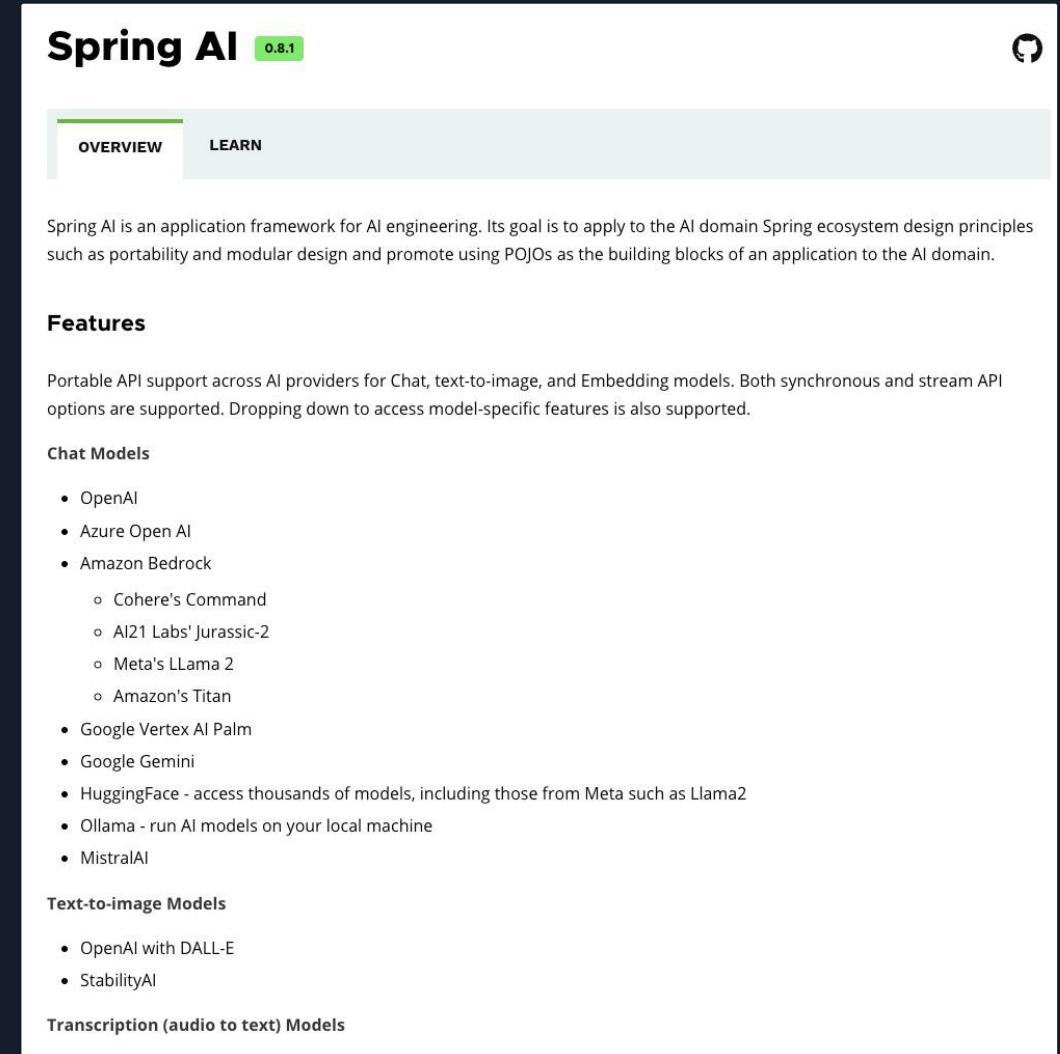
- OpenAI with DALL-E
- StabilityAI

Transcription (audio to text) Models

Spring AI

AI for Spring Developers

Aligns with Spring project design values
Component Abstractions & Default Implementations
Portable Chat Completion and EmbeddingClient
Multimodality Support
Portable Vector Store API & Query Language
Function Calling
Key Components



The screenshot shows the Spring AI documentation website. At the top, there's a header with the title "Spring AI" and a version "0.8.1". Below the header, there are two tabs: "OVERVIEW" (which is selected) and "LEARN". The main content area starts with a brief introduction: "Spring AI is an application framework for AI engineering. Its goal is to apply to the AI domain Spring ecosystem design principles such as portability and modular design and promote using POJOs as the building blocks of an application to the AI domain." Under the "Features" section, it says: "Portable API support across AI providers for Chat, text-to-image, and Embedding models. Both synchronous and stream API options are supported. Dropping down to access model-specific features is also supported." The "Chat Models" section lists the following providers:

- OpenAI
- Azure Open AI
- Amazon Bedrock
 - Cohere's Command
 - AI21 Labs' Jurassic-2
 - Meta's Llama 2
 - Amazon's Titan
- Google Vertex AI Palm
- Google Gemini
- HuggingFace - access thousands of models, including those from Meta such as Llama2
- Ollama - run AI models on your local machine
- MistralAI

The "Text-to-image Models" section lists:

- OpenAI with DALL-E
- StabilityAI

The "Transcription (audio to text) Models" section is mentioned but has no list items.

<https://docs.spring.io/spring-ai/reference/index.html>

Spring AI API

The Spring AI API covers a wide range of functionalities

Chat Model

Give it some text,
get some text back

Text to Image

Give it some text,
get an image back

Transcription

Give it some audio,
get some text back

Embedding

Portable API across
Vector Store
Providers

Functions

AI model invoke
your POJO
`java.util.Function.`
Object

Chat Model

Give it some text, get some text back

Open AI

Azure Open AI

Amazon Bedrock

Google Vertex AI Palm

Google Gemeni

**HuggingFace - Access to thousands of models,
including those from Meta such as Llama 2**

Ollama - Run AI Models on your local machine

MistralAI



Text-to-Image Models

Give it some text, get an image back

- **OpenAI with DALL-E**
- **StabilityAI**



Transcription

Give it some audio, get some text back

- Open AI



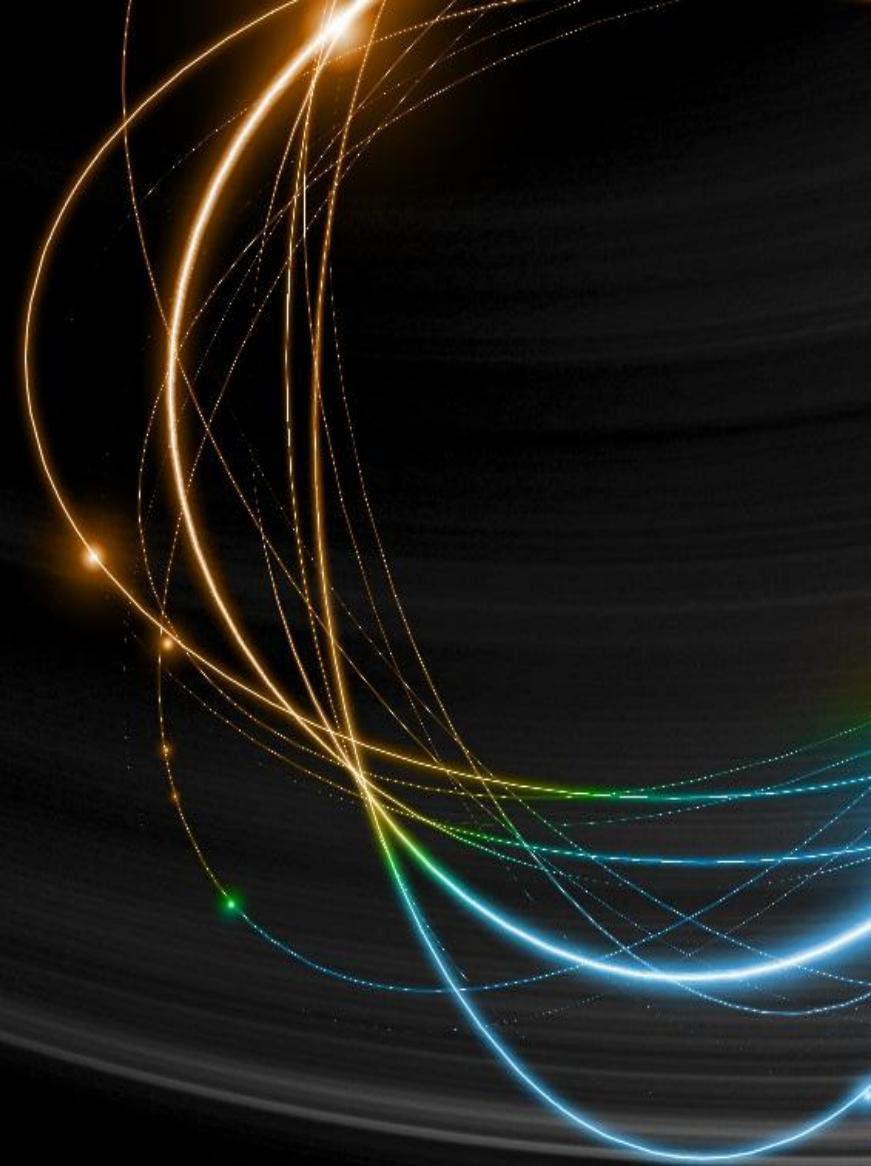
Embedding Models

Portable API across Vector Store providers

- **Open AI**
- **Azure Open AI**
- **Ollama**
- **ONNX**
- **PostgresML**
- **Bedrock Cohere**
- **Bedrock Titan**
- **Google VertexAI**
- **MistralAI**



WHAT ARE AI PROMPTS?



Prompts serve as the foundation for the language-based inputs that guide an AI model to produce specific outputs. For those familiar with ChatGPT, a prompt might seem like merely the text entered into a dialog box that is sent to the API. However, it encompasses much more than that. In many AI Models, the text for the prompt is not just a simple string.

Prompt Engineering

Effective Communication

- Input directing an AI model to produce specific outputs
- Model outputs are greatly inference by prompt style and wording
- Prompt Engineering
- Prompt techniques and effective prompts are share in the community
- OpenAI Guidelines
- Mini Course: ChatGPT Prompt Engineering for developers
- Prompts and Spring
- Prompt management relies on Text Template Engines
- Analogous to the view in Spring MVC



```
public class Prompt implements ModelRequest<List<Message >>{ private final List<Message> messages;
    private ChatOptions modelOptions;

    public Prompt(String contents) { this((Message)(new
        UserMessage(contents)));
    }

    public Prompt(Message message)
    { this(Collections.singletonList(message));
    }

    public Prompt(List<Message> messages) { this.messages =
        messages;
    }

    public Prompt(String contents, ChatOptions modelOptions) { this((Message)(new UserMessage(contents)),
        modelOptions);
    }

    public Prompt(Message message, ChatOptions modelOptions) { this(Collections.singletonList(message),
        modelOptions);
    }

    public Prompt(List<Message> messages, ChatOptions modelOptions) { this.messages = messages;
        this.modelOptions = modelOptions;
    }
}
```



```
public class Prompt implements ModelRequest<List<Message>>{ private final List<Message> messages;
    private ChatOptions modelOptions;

    public Prompt(String contents) { this((Message)(new UserMessage(contents)));
    }

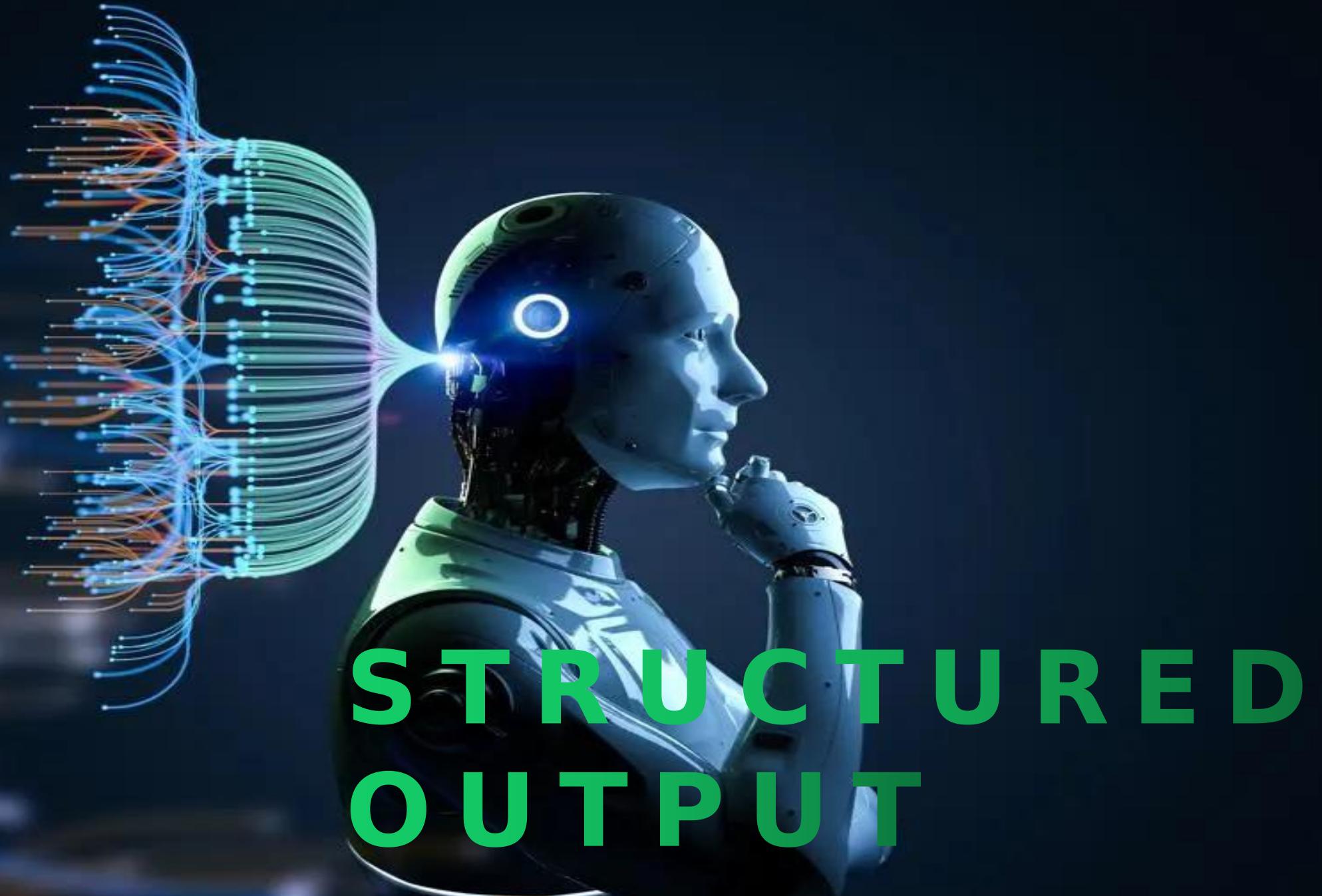
    public Prompt(Message message) { this(Collections.singletonList(message));
    }

    public Prompt(List<Message> messages) { this.messages =
        Choose Implementation of Message (6 found)
        © AbstractMessage (org.springframework.ai.chat.messages) Maven: org.springframework.ai:spring-ai-core:0.8.1 (spring-ai-core-0.8.1.jar) 📁
        © AssistantMessage (org.springframework.ai.chat.messages) Maven: org.springframework.ai:spring-ai-core:0.8.1 (spring-ai-core-0.8.1.jar) 📁
        © ChatMessage (org.springframework.ai.chat.messages) Maven: org.springframework.ai:spring-ai-core:0.8.1 (spring-ai-core-0.8.1.jar) 📁
        © FunctionMessage (org.springframework.ai.chat.messages) Maven: org.springframework.ai:spring-ai-core:0.8.1 (spring-ai-core-0.8.1.jar) 📁
        © SystemMessage (org.springframework.ai.chat.messages) Maven: org.springframework.ai:spring-ai-core:0.8.1 (spring-ai-core-0.8.1.jar) 📁
        © UserMessage (org.springframework.ai.chat.messages) Maven: org.springframework.ai:spring-ai-core:0.8.1 (spring-ai-core-0.8.1.jar) 📁
    public Prompt(Message message, ChatOptions modelOptions, { this.Collections.singletonList(message),
        modelOptions);
    }

    public Prompt(List<Message> messages, ChatOptions modelOptions) { this.messages = messages;
        this.modelOptions = modelOptions;
    }
}
```

Roles

- **System Role:** Guides the AI's behavior and response style, setting parameters or rules for how the AI interprets and replies to the input. It's akin to providing instructions to the AI before initiating a conversation.
- **User Role:** Represents the user's input - their questions, commands, or statements to the AI. This role is fundamental as it forms the basis of the AI's response.
- **Assistant Role:** The AI's response to the user's input. More than just an answer or reaction, it's crucial for maintaining the flow of the conversation. By tracking the AI's previous responses (its 'Assistant Role' messages), the system ensures coherent and contextually relevant interactions.
- **Function Role:** This role deals with specific tasks or operations during the conversation. While the System Role sets the AI's overall behavior, the Function Role focuses on carrying out certain actions or commands the user asks for. It's like a special feature in the AI, used when needed to perform specific functions such as calculations, fetching data, or other tasks beyond just talking. This role allows the AI to offer practical help in addition to conversational responses.



**STRUCTURED
OUTPUT**

Output Parsing

Challenges with handling the response

The Challenge

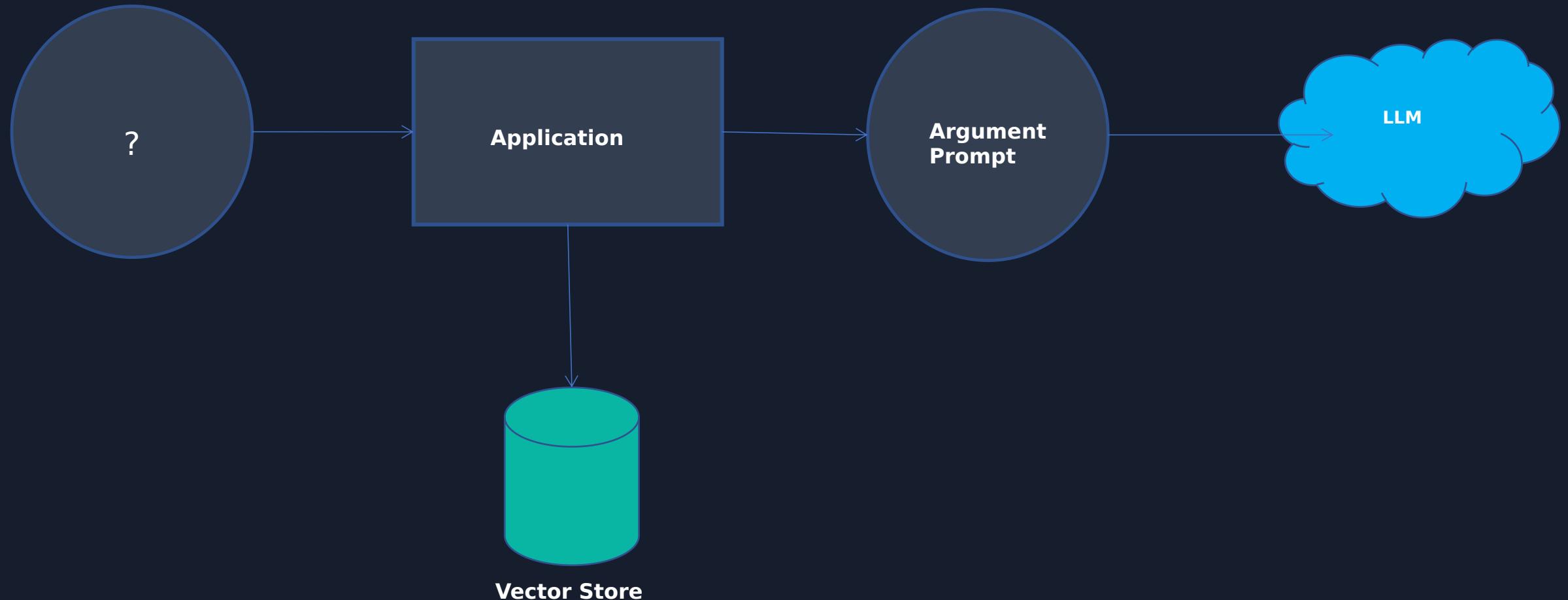
- Output of Generative LLM is a `java.util.String`
- Even if you ask for JSON, you get a JSON String
- ChatGPT wants to chat, not reply in JSON
- OpenAI has introduced a new feature to help with this
- Spring AI's `OutputParser` uses refined prompts to desired results



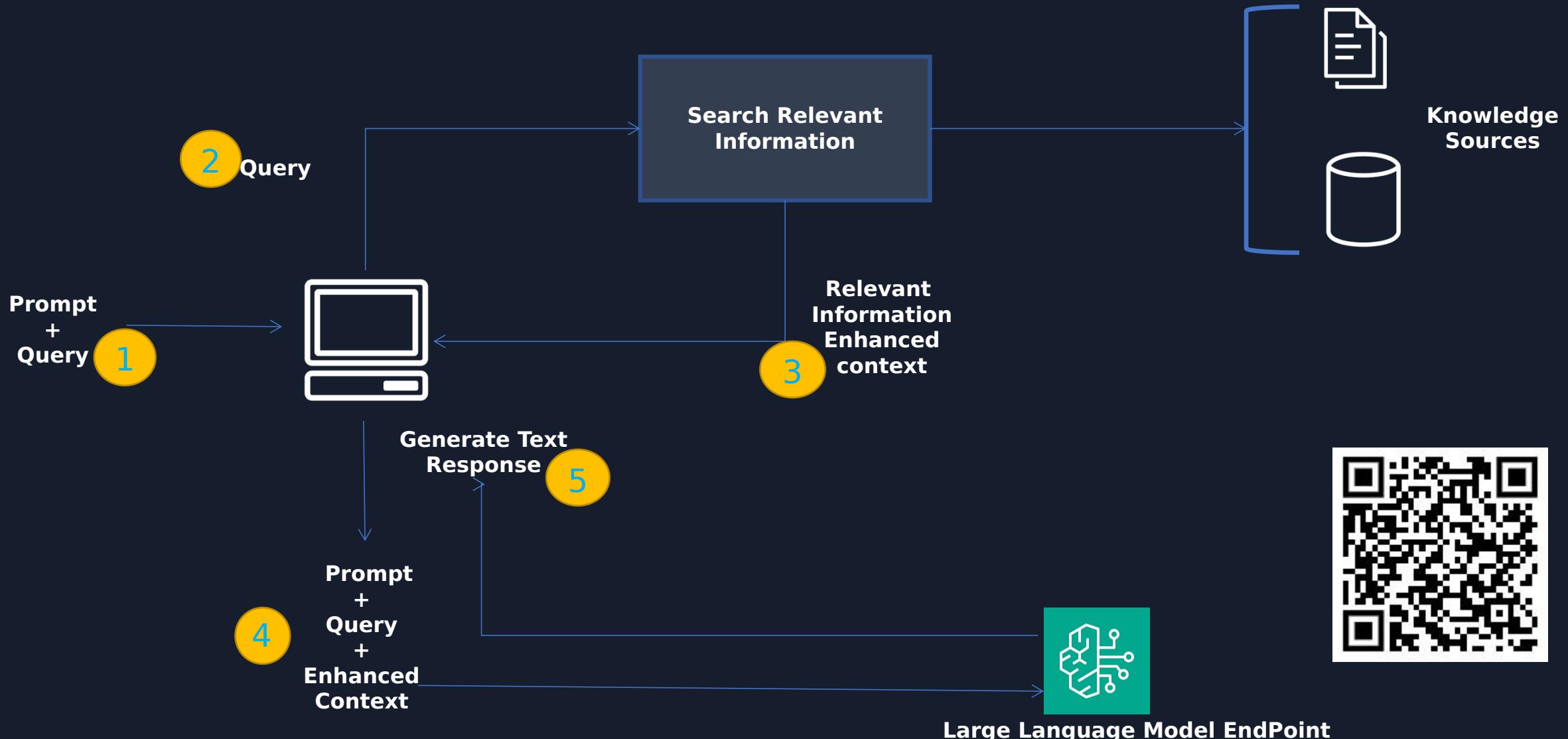
RAG

RETRIEVAL AUGMENTED GENERATION

AI Application Architecture - RAG



Conceptual Flow of using RAG with LLMs



Vector Stores

Not just for text search

- Azure Vector Search
- ChromaVectorStore
- MilvusVectorStore
- Neo4JVectorStore
- PgVectorStore
- QdrantVectorStore
- RedisVectorStore
- WeaviateVecvtorStore
- SimpleVectorStore

gpt-4o ◊

Compare ↗

SYSTEM

Enter system instructions

USER

Hello my name is Weder

ASSISTANT

Hello, Weder! How can I assist you today?

USER

What is my name?

ASSISTANT

Your name is Weder. How can I help you today, Weder?

Latency 681ms · Finish reason: stop · 58 tokens

Enter user message...

User



Add

Run Ctrl ↵

The web is stateless



Stateless



Stateful



Topic 2: Getting to Know OpenAI and AWS Bedrock





```
#!/bin/bash

echo "Calling Open AI"
MY_OPENAI_KEY="YOUR_API_KEY_HERE"
PROMPT="When was the first version of Java released?"

curl https://api.openai.com/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $MY_OPENAI_KEY" \
-d '{"model": "gpt-3.5-turbo-0125", "messages": [{"role": "user", "content": "${PROMPT}"}]}'
```

Response

Calling Open AI

```
{  
  "id": "chatcmpl-A5e48Ns4Tu9NscMH8oAIdC1NF8dH6",  
  "object": "chat.completion",  
  "created": 1725909016,  
  "model": "gpt-3.5-turbo-0125",  
  "choices": [  
    {  
      "index": 0,  
      "message": {  
        "role": "assistant",  
        "content": "The first version of Java, known as Java 1.0, was released by Sun Microsystems on January 23, 1996.",  
        "refusal": null  
      },  
      "logprobs": null,  
      "finish_reason": "stop"  
    }  
  ],  
  "usage": {  
    "prompt_tokens": 16,  
    "completion_tokens": 28,  
    "total_tokens": 44  
  },  
  "system_fingerprint": null  
}
```

```
public static void main(String[] args) throws IOException, InterruptedException {
    var apiKey = "YOUR_API_KEY_HERE";
    var body = """
        {
            "model": "gpt-4",
            "messages": [
                {
                    "role": "assistant",
                    "content": "What is Spring Boot?"
                }
            ]
        }""";
    HttpRequest request = HttpRequest.newBuilder()
        .uri(URI.create("https://api.openai.com/v1/chat/completions"))
        .header("Content-Type", "application/json")
        .header("Authorization", "Bearer " + apiKey)
        .POST(HttpRequest.BodyPublishers.ofString(body))
        .build();

    var client = HttpClient.newHttpClient();
    var response = client.send(request, HttpResponse.BodyHandlers.ofString());
    System.out.println(response.body());
}
```



OpenAI

<https://platform.openai.com> · Traduzir esta página

OpenAI Platform

Explore resources, tutorials, API docs, and dynamic examples to get the most out of OpenAI's developer platform.



ChatGPT →

Interact with our flagship language models in a conversational interface

API →

Integrate OpenAI models into your application or business



DASHBOARD

Assistants

Fine-tuning

Batches

Storage

Usage

API keys

API keys

+ Create new secret key

Project API keys have replaced user API keys.

We recommend using project based API keys for more granular control over your resources. [Learn more](#)[View user API keys](#)

As an owner of this project, you can view and manage all API keys in this project.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically disable any API key that has leaked publicly.

View usage per API key on the [Usage page](#).

NAME	SECRET KEY	CREATED	LAST USED	CREATED BY	PERMISSIONS		
NewSpringAI	sk-...2v08	28 de jun. de 2024	28 de jul. de 2024	Weder Sousa	All		

Models

Models

Flagship models

GPT-4o

Our high-intelligence flagship model for complex, multi-step tasks

Text and image input, text output

128k context length

Optimized for intelligence, higher price per token

GPT-4o mini

Our affordable and intelligent small model for fast, lightweight tasks

Text and image input, text output

128k context length

Optimized for speed, lower price per token

New

[Model pricing details >](#)

Models overview

The OpenAI API is powered by a diverse set of models with different capabilities and price points. You can also make customizations to our models for your specific use case with [fine-tuning](#).

MODEL	DESCRIPTION
GPT-4o	Our high-intelligence flagship model for complex, multi-step tasks
GPT-4o mini	Our affordable and intelligent small model for fast, lightweight tasks
GPT-4 Turbo and GPT-4	The previous set of high-intelligence models
GPT-3.5 Turbo	A fast, inexpensive model for simple tasks
DALL-E	A model that can generate and edit images given a natural language prompt
TTS	A set of models that can convert text into natural sounding spoken audio
Whisper	A model that can convert audio into text

Tokens

GPT-4o

GPT-4o is our most advanced multimodal model that's faster and cheaper than GPT-4 Turbo with stronger vision capabilities. The model has 128K context and an October 2023 knowledge cutoff.

[Learn about GPT-4o ↗](#)

Model	Pricing	Pricing with Batch API*
gpt-4o	\$5.00 / 1M input tokens	\$2.50 / 1M input tokens
	\$15.00 / 1M output tokens	\$7.50 / 1M output tokens
gpt-4o-2024-08-06	\$2.50 / 1M input tokens	\$1.25 / 1M input tokens
	\$10.00 / 1M output tokens	\$5.00 / 1M output tokens
gpt-4o-2024-05-13	\$5.00 / 1M input tokens	\$2.50 / 1M input tokens
	\$15.00 / 1M output tokens	\$7.50 / 1M output tokens

[Vision pricing calculator](#)

Set model

gpt-4o-2024-08-06

Set width

150

px

by

150

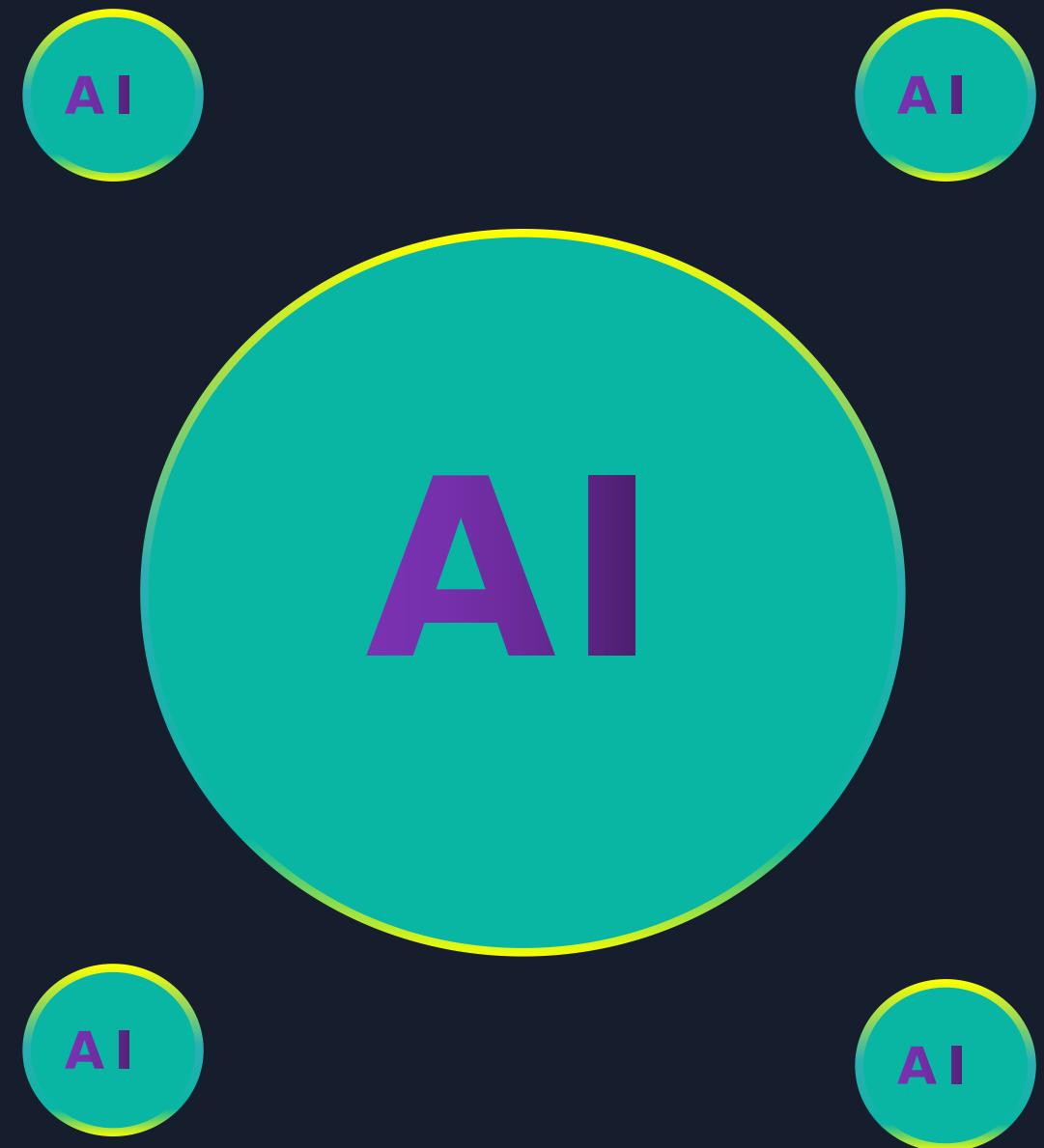
px

= \$0.000638

Set height

Low resolution

WHAT ARE AI TOKENS?



Tokenizer

Learn about language model tokenization

OpenAI's large language models (sometimes referred to as GPTs) process text using **tokens**, which are common sequences of characters found in a set of text. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens.

You can use the tool below to understand how a piece of text might be tokenized by a language model, and the total count of tokens in that piece of text.

It's important to note that the exact tokenization process varies between models. Newer models like GPT-3.5 and GPT-4 use a different tokenizer than previous models, and will produce different tokens for the same input text.

GPT-4o & GPT-4o mini (coming soon)

GPT-3.5 & GPT-4

GPT-3 (Legacy)

My name is Weder Sousa, I have worked in the IT field since 2000.
I currently have a degree in Computer Science.
I have two specializations in Information Security (UniLeya) and another
in Intermediate Media (UFG).
I graduated from Cepa (Center for Professional Studies of Anápolis) in
Systems Development and participated in FreeCodeCamp to develop
knowledge.

Clear

Show example

Tokens	Characters
83	362

My name is Weder Sousa, I have worked in the IT field since 2000.
I currently have a degree in Computer Science.
I have two specializations in Information Security (UniLeya) and another
in Intermediate Media (UFG).
I graduated from Cepa (Center for Professional Studies of Anápolis) in
Systems Development and participated in FreeCodeCamp to develop
knowledge.

Text Token IDs



Amazon Bedrock



Amazon Bedrock

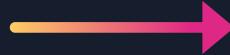


https://aws.amazon.com/pt/bedrock/?did=ap_card&trk=ap_card

MACHINE
LEARNING

SIMPLE

INPUTS



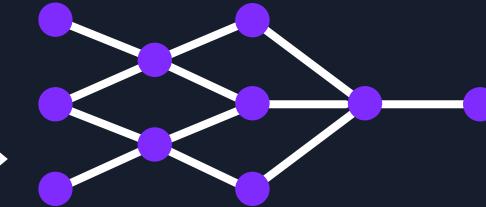
SIMPLE

OUTPUTS

DEEP
LEARNING

COMPLEX

INPUTS



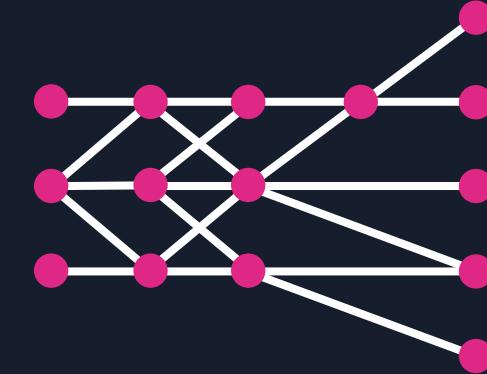
SIMPLE

OUTPUTS

FOUNDATION
MODELS

COMPLEX

INPUTS



COMPLEX

OUTPUTS

Foundation Models



Jurassic-2 series
By AI21 Labs



Titan
By Amazon



Claude
By Anthropic



Command
By Cohere



Llama 2
By Meta



Stable Diffusion
By Stability AI



Mistral
By Mistral AI

+

+



Amazon Bedrock

The easiest way to build and scale generative AI applications with foundation models

AI21labs

amazon

ANTHROPIC

cohere

Meta

MISTRAL
AI

stability.ai



Contextual answers,
summarization,
paraphrasing

Text summarization,
generation, Q&A, search,
image generation

Summarization, complex
reasoning, writing, coding

Text generation,
search, classification

Q&A and reading
comprehension

Text summarization,
Q&A, text classification,
text completion, code
generation

High-quality images
and art

Jurassic-2 Ultra

Amazon Titan Text Premier

Claude 3 Opus

Command

Llama 3 8B

Mistral Large

Stable Diffusion XL 1.0

Jurassic-2 Mid

Amazon Titan Text Lite

Claude 3 Sonnet

Command Light

Llama 3 70B

Mistral 7B

Stable Diffusion XL 0.8

Amazon Titan Text Express

Claude 3 Haiku

Embed English

Llama 2 13B

Mixtral 8x7B

Amazon Titan Text Embeddings

Claude 2.1

Embed Multilingual

Llama 2 70B

Amazon Titan Text Embeddings
V2

Claude 2

Command R+

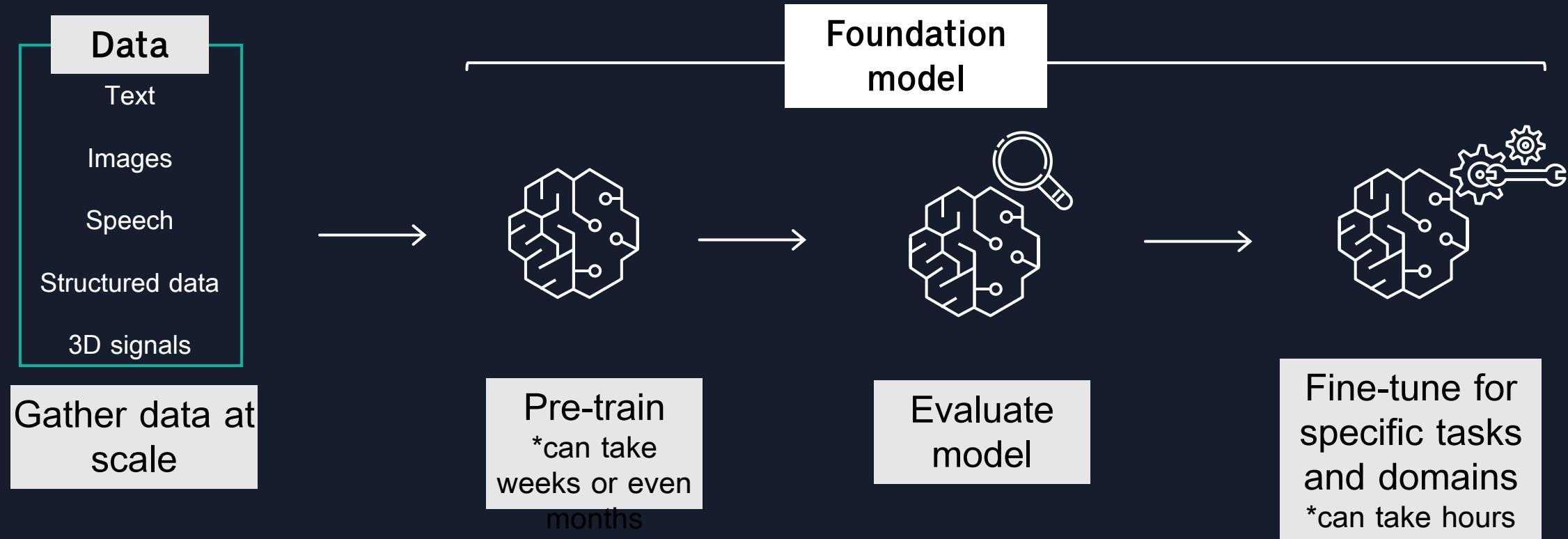
Amazon Titan Multimodal
Embeddings

Claude Instant

Command R

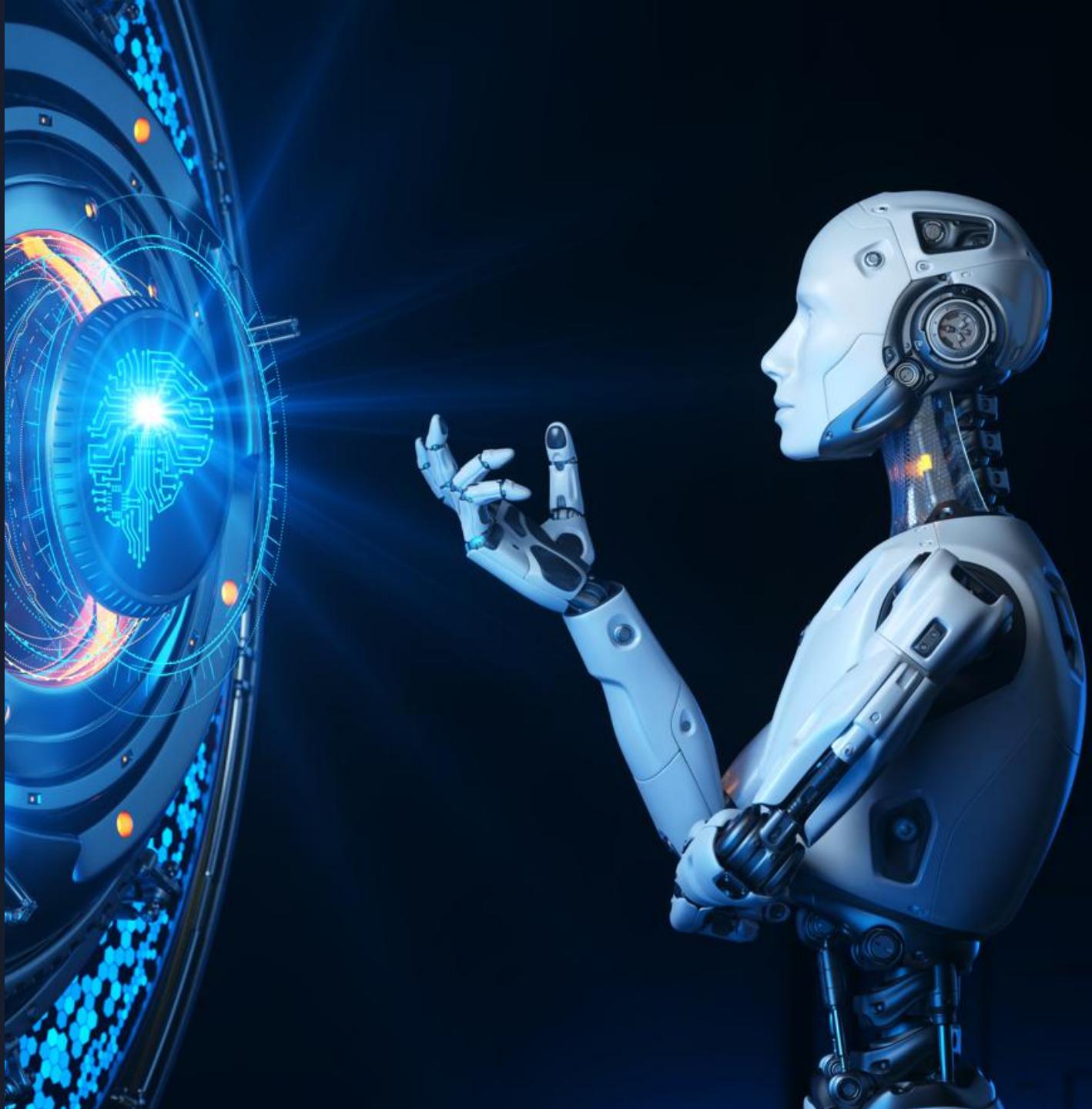
Amazon Titan Image Generator

How does a foundation model work?

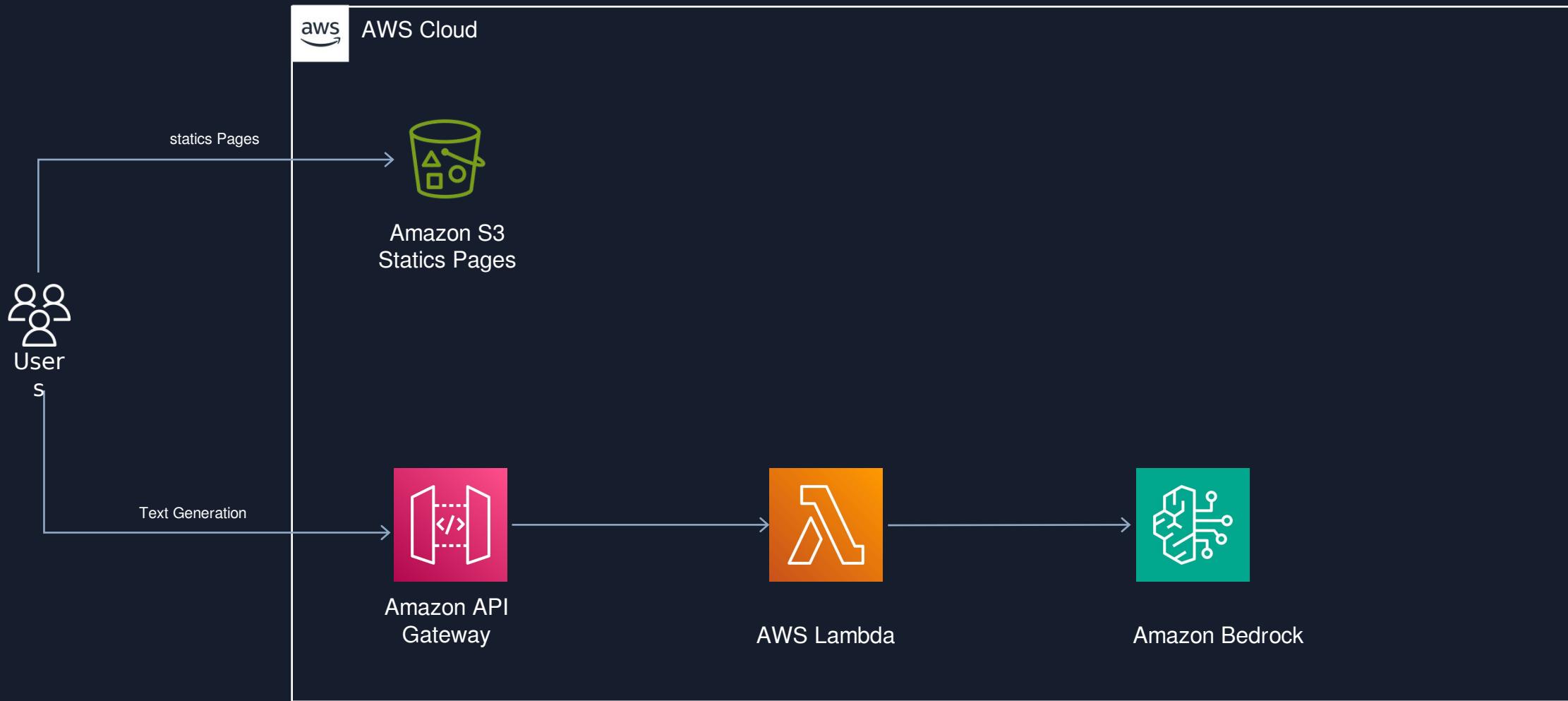


Architecture AWS Bedrock

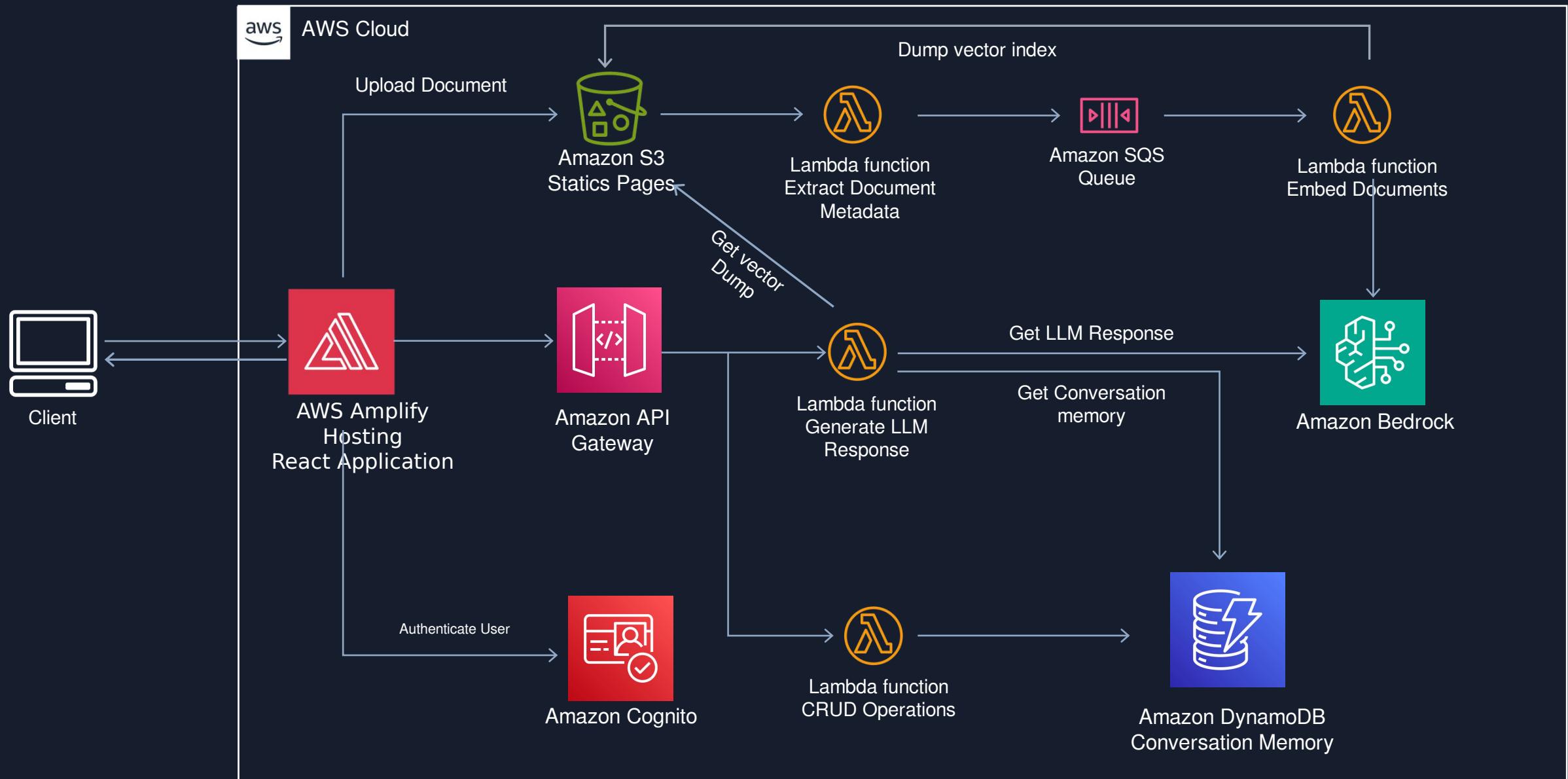
**Topic 6: Presentation of the integration
architecture AWS Bedrock**



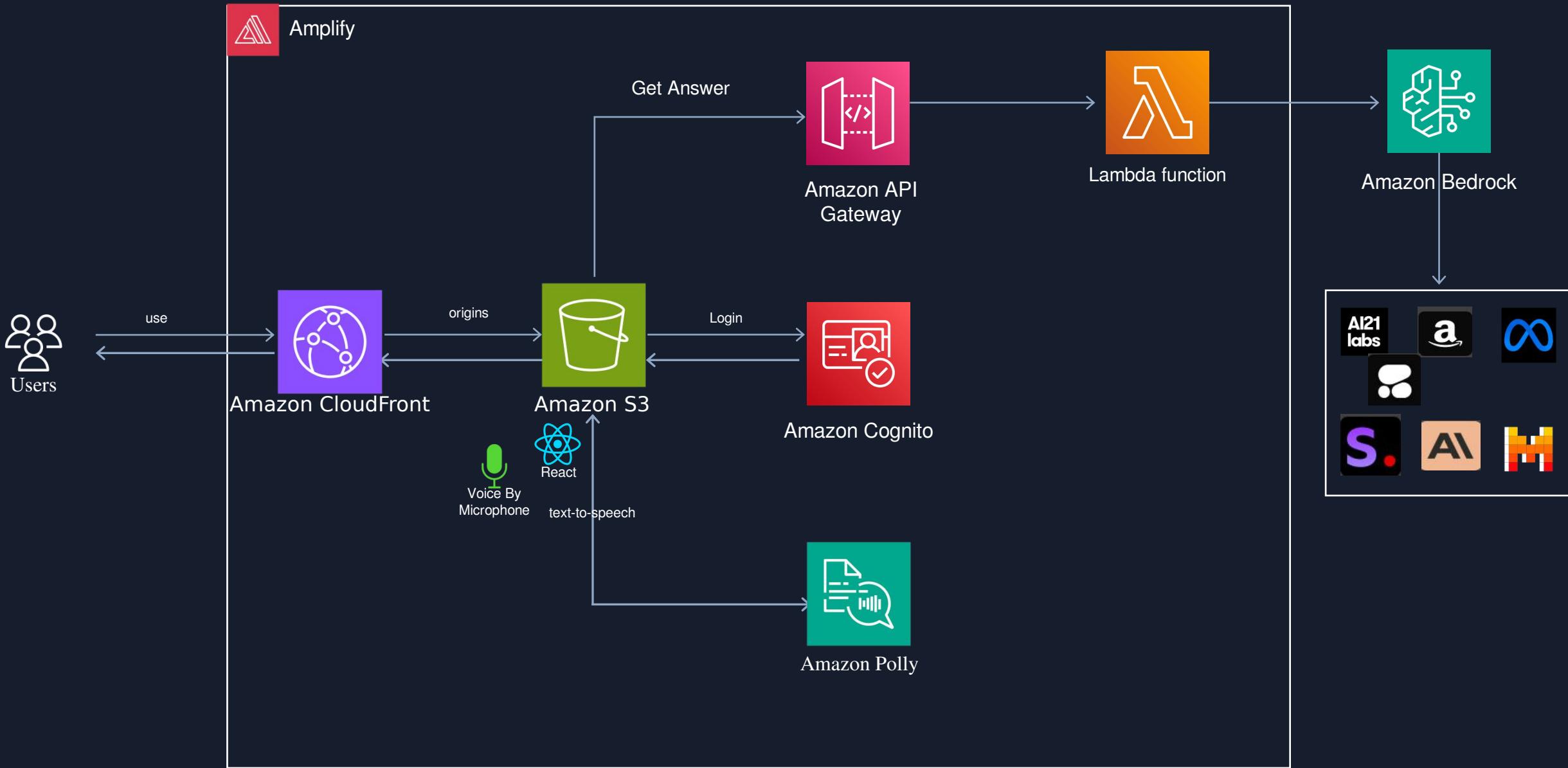
Amazon Bedrock Serveless Basic Architecture



Advantages of the Serverless Approach



Advantages of the Serverless Approach





Amazon Q Developer

The most capable generative AI-powered assistant for software development

The screenshot shows the Amazon Q Developer interface. On the left, there's a sidebar titled "AMAZON Q: CHAT" with various icons. Below it, a message from "Q - Dev" says: "formatting. By following these steps, you can update the `image_understanding_st.py` script to display the uploaded image in the first column and add a button to generate a description for the image using the Claude-3-Sonnet model from Anthropic. Additionally, you will add tests for the new `describe_image` function and update the `requirements.txt` file if necessary. Note: To handle current demand, Q will use a smaller, higher-speed model for code generation." A message box asks, "Would you like to generate a suggestion for this? You'll review a file diff before inserting into your project." Below this is a "Generating code ..." button. A "Code suggestions 2 file(s)" section lists changes: "Changes 2 file(s)", "image_examples 1 file(s)", "image_understanding_st.py", and "tests 1 file(s)", "test_image_understanding....". At the bottom of the sidebar are "Insert code" and "Provide feedback to regenerate" buttons.

The main area is a code editor titled "gen-ai-workshop". It has tabs for "Welcome", "image_understanding_st.py", and "image_understanding_st.py ↔ image_understanding_st.py". The code is a Python script:

```
60 def pil_to_base64(image, format="png"):
61     with io.BytesIO() as buffer:
62         image.save(buffer, format)
63         return base64.b64encode(buffer.getvalue()).decode()
64
65+ def describe_image(image):
66+     image_base64 = pil_to_base64(image)
67+     description = call_claude_sonnet(image_base64)
68+     return description
69
70 # Streamlit file uploader for only for images
71 user_image = st.file_uploader("Upload an image", type=["png", "jpg", "jpeg"])
72
73 # Creat two columns, one to show the uploaded image, another for the new image
74 col1, col2 = st.columns(2)
75
76 # Show the uploaded image
77 if user_image is not None:
78     user_image = Image.open(user_image)
79     # TODO Finish App with Q
80
81+     if col2.button("Describe Image"):
82+         description = describe_image(user_image)
83+         col2.write(description)
84
85 else:
86     col2.write("No image uploaded")
87
```

The code editor includes syntax highlighting and line numbers. The right side of the interface shows a terminal window with the command "ubuntu@dev:~/gen-ai-workshop\$". The bottom status bar indicates "Ln 57, Col 1" and "Spaces: 4 Python Layout: U.S."



Amazon Q

ChatClient.java © Prompt.java © BedrockSDKTitanServiceImp.java © GenerateIm... : Amazon Q Chat

17 public class BedrockSDKTitanServiceImp implements BedrockSDKTitanServ... 2 1 ↻

18 public String invokeModel(String prompt) { 1 usage ▲ WederSousa

22 var client = BedrockRuntimeClient.builder()

23 .credentialsProvider(DefaultCredentialsProvider.create())

24 .region(Region.US_EAST_1)

25 .build();

26 // Set the model ID, e.g., Titan Image G1.

27 var modelId = "amazon.titan-image-generator-v1";

28 // The InvokeModel API uses the model's native payload.

29 // Learn more about the available inference parameters and response fields

30 // <https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-ti>

31 var nativeRequestTemplate = """"

32 {

33 "taskType": "TEXT_IMAGE",

34 "textToImageParams": { "text": "{{prompt}}" },

35 "imageGenerationConfig": { "seed": {{seed}} } }

36 }""";

37 // Define the prompt for the image generation.

38 // Get a random 31-bit seed for the image generation (max. 2,147,483,647).

39 var seed = new BigInteger(numBits: 31, new SecureRandom());

40 // Embed the prompt and seed in the model's native request payload.

41 var nativeRequest = nativeRequestTemplate

42 .replace(target: "{{prompt}}", prompt)

43 .replace(target: "{{seed}}", seed.toString());

44 try {

45 // Encode and send the request to the Bedrock Runtime.

46 var response = client.invokeModel(request -> request

47 }

48 }

49 }

50 }

51 }

Chat X +

5. Embedding Prompt and Seed in the Request

- It replaces the placeholders in the `nativeRequestTemplate` with the actual prompt (`prompt`) and the generated seed (`seed.toString()`).
- The resulting string (`nativeRequest`) contains the complete request payload for the Titan Image Generator model.

6. Invoking the Model

- It prepares to invoke the model by calling the `invokeModel` method on the `client` object.
- The `invokeModel` method takes a lambda expression that sets the request body to the `nativeRequest` payload, encoded as UTF-8 bytes using `SdkBytes.fromUtf8String`.

At this point, the code is ready to send the request to the Bedrock Runtime service and receive the generated image based on the provided prompt and seed value.

In summary, this code sets up the necessary client, prepares a request payload with a prompt and a random seed, and gets ready to invoke the Titan Image Generator model on the Amazon Bedrock service to generate an image based on the provided text.

Ask a question or enter "/" for quick commands

4000/4000

Use of Amazon Q is subject to the [AWS Responsible AI Policy](#).
https://aws.amazon.com/pt/q/developer/?did=ap_card&trk=ap_card



Generative AI



aws COMMUNITY

Generative AI

Your space for generative AI tools, guides, and technology.

The Secrets of Agents

Are agents and agentic workflows taking over as the leading use case for generative AI? Let's take a look at agents...



Mike Chambers
Amazon Employee

AI running its own code! : Agentic Code Interpreter

Public Preview: Code Interpreter for Agents for Amazon Bedrock - With sample code.

Jul 17 • generative-ai



Mike Chambers
Amazon Employee

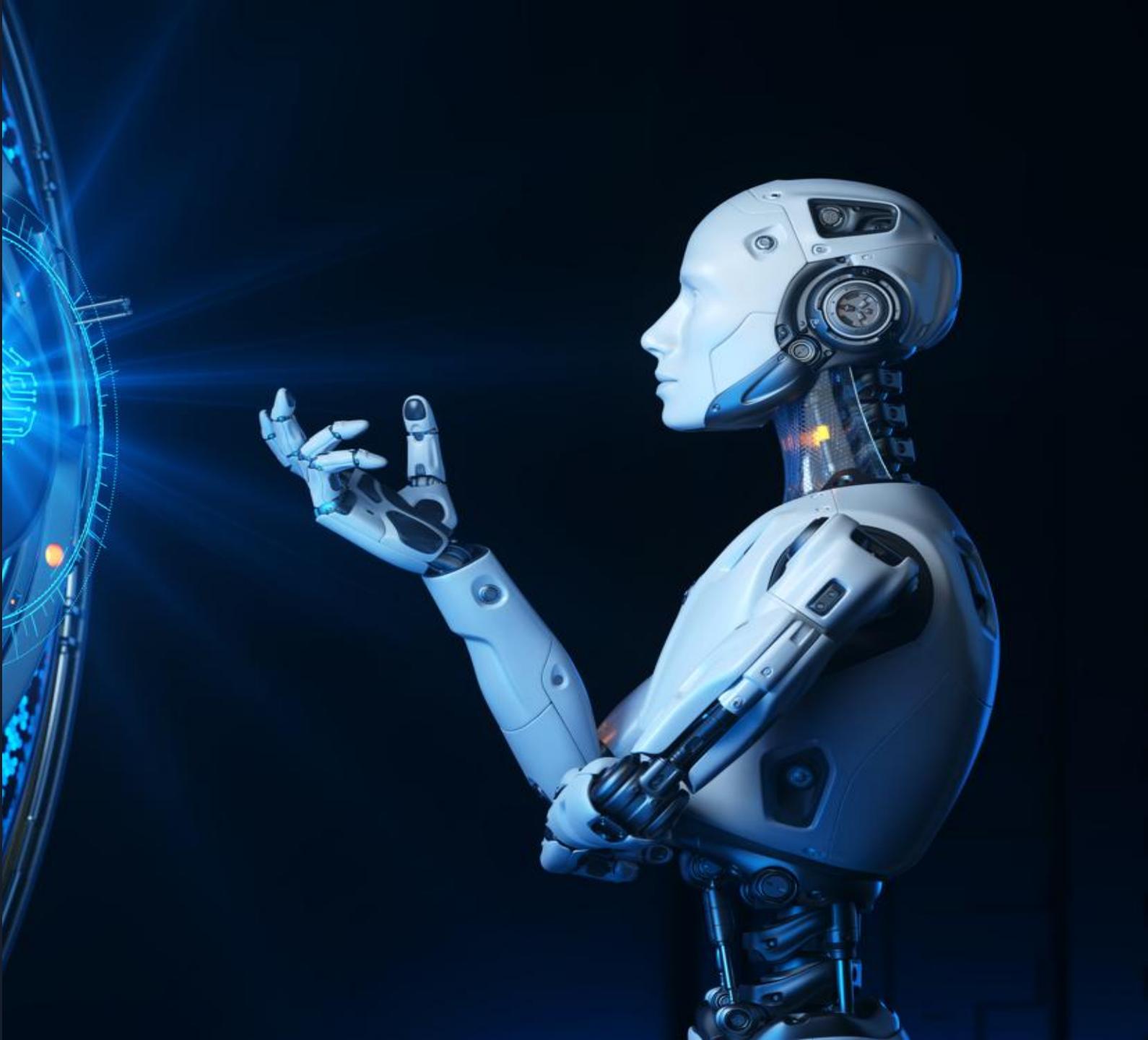
Solving LLM Amnesia - Cross Session Memory

Public Preview: Long Term Memory for Agents for Amazon Bedrock

Jul 17 • generative-ai



Topic 3: Integration: Spring AI with OpenAI (ChatGPT)





DEMO

Getting started



Project

Gradle - Groovy Gradle - Kotlin

Maven

Language

Java Kotlin Groovy

Spring Boot

3.4.0 (SNAPSHOT) 3.4.0 (M2) 3.3.4 (SNAPSHOT) 3.3.3
 3.2.10 (SNAPSHOT) 3.2.9

Project Metadata

Group dev.wsousa

Artifact spring-ai

Name spring-ai

Description Demo project for Spring Boot

Package name dev.wsousa.spring-ai

Packaging Jar War

Java 22 21 17

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

OpenAI AI

Spring AI support for ChatGPT, the AI language model and DALL-E, the Image generation model from OpenAI.

```

spring.application.name=hello-spring-ai
spring.ai.openai.api-key=YOUR_KEY_HERE
spring.ai.openai.chat.options.model=gpt-4

```

Configuration Properties

The prefix `spring.ai.openai.chat` is the property prefix that lets you configure the chat client implementation for OpenAI.

Property	Description	Default
<code>spring.ai.openai.chat.enabled</code>	Enable OpenAI chat client.	true
<code>spring.ai.openai.chat.base-url</code>	Optional overrides the <code>spring.ai.openai.base-url</code> to provide chat specific url	-
<code>spring.ai.openai.chat.api-key</code>	Optional overrides the <code>spring.ai.openai.api-key</code> to provide chat specific api-key	-
<code>spring.ai.openai.chat.options.model</code>	This is the OpenAI Chat model to use	<code>gpt-3.5-turbo</code> (the <code>gpt-3.5-turbo</code> , <code>gpt-4</code> , and <code>gpt-4-32k</code> point to the latest model versions)
<code>spring.ai.openai.chat.options.temperature</code>	The sampling temperature to use that controls the apparent creativity of generated completions. Higher values will make output more random while lower values will make results more focused and deterministic. It is not recommended to modify temperature and <code>top_p</code> for the same completions request as the interaction of these two settings is difficult to predict.	0.8
<code>spring.ai.openai.chat.options.frequencyPenalty</code>	Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat at the same line verbatim.	0.0f
<code>spring.ai.openai.chat.options.logitBias</code>	Modify the likelihood of specified tokens appearing in the completion.	-
<code>spring.ai.openai.chat.options.maxTokens</code>	The maximum number of tokens to generate in the chat completion. The total length of input tokens and generated tokens is limited by the model's context length.	-
<code>spring.ai.openai.chat.options.n</code>	How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep n as 1 to minimize costs.	1
<code>spring.ai.openai.chat.options.presencePenalty</code>	Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.	-
<code>spring.ai.openai.chat.options.responseFormat</code>	An object specifying the format that the model must output. Setting to <code>{ "type": "json_object" }</code> enables JSON mode, which guarantees the message the model generates is valid JSON.	-

application.properties

```
1 spring.application.name=spring-ai-open-ai
2 spring.ai.openai.api-key=${OPEN_AI_KEY}
3 spring.ai.openai.chat.options.model=gpt-4o
4
5 weather.api-key=95ae10c827154be6b00193851242807
6 weather.api-url=http://api.weatherapi.com/v1
```

GET [http://localhost:8080/chat/direct/?message="What is Spring Boot?"](http://localhost:8080/chat/direct/?message='What is Spring Boot?')

Params • Authorization Headers (6) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description	... Bulk Edit
message	"What is Spring Boot?"		
Key	Value	Description	

Body Cookies Headers (5) Test Results Status: 200 OK Time: 13.41 s Size: 1.11 KB Save as example ...

Pretty Raw Preview Visualize Text Copy Search

```
1 {
2     "id": "chatmpl-A5fe8Got1CUp0WuZ8H4QvE8Ry1rMn",
3     "object": "chat.completion",
4     "created": 1725915092,
5     "model": "gpt-4-0613",
6     "choices": [
7         {
8             "index": 0,
9             "message": {
10                 "role": "assistant",
11                 "content": "Spring Boot is an open-source Java-based framework used to create stand-alone, production-grade Spring based applications with minimal effort. It is designed to simplify the bootstrapping and development of a new Spring application. The framework takes an opinionated approach to configuration, freeing developers from the need to define a boilerplate configuration. Spring Boot offers a 'run and see' methodology and aims to shorten the code length and provide an easier way to develop Spring applications.",
12                 "refusal": null
13             },
14             "logprobs": null,
15             "finish_reason": "stop"
16         }
17     ],
18     "usage": {
19         "prompt_tokens": 12,
20         "completion_tokens": 90,
21         "total_tokens": 102
22     },
23     "system_fingerprint": null
24 }
```

```
@RestController no usages ▲ Weder Sousa
@RequestMapping("/chat")
public class ChatController {

    private final ChatClient chatClient; 4 usages
```

```
    public ChatController(ChatClient.Builder builder) { no usages ▲ Weder Sousa
        this.chatClient = builder
            .build();
    }
```

```
@GetMapping("/") no usages ▲ Weder Sousa
public String joke(@RequestParam(value = "message", defaultValue = "Tell me a dad joke about dogs") String message) {
    return chatClient.prompt() ChatClientRequest
        .user(message)
        .call() CallResponseSpec
        .content(); // short for getResult().getOutput().getContent();
}
```

```
@GetMapping("/jokes-by-topic") no usages ▲ Weder Sousa
public String jokesByTopic(@RequestParam String topic) {
    return chatClient.prompt() ChatClientRequest
        .user(u -> u.text("Tell me a joke about {topic}").param( k: "topic",topic))
        .call() CallResponseSpec
        .content();
}
```

```
@GetMapping("jokes-with-response") no usages ▲ Weder Sousa
public ChatResponse jokeWithResponse(@RequestParam(value = "message", defaultValue = "Tell me a dad joke about computers") String message) {
    return chatClient.prompt() ChatClientRequest
        .user(message)
        .call() CallResponseSpec
        .chatResponse();
}
```

```
weder@weder:~$ http get :8080/chat/?message="Tell me a dad joke about dogs" -b --pretty none
Sure, here's one for you:

Why did the dog sit in the shade?

Because he didn't want to be a hot dog! 🐶☀️
```

1

2

```
@RestController no usages ▲ Weder Sousa
@RequestMapping("/chat")
public class ChatController {

    private final ChatClient chatClient; 4 usages

    public ChatController(ChatClient.Builder builder) { no usages ▲ Weder Sousa
        this.chatClient = builder
            .build();
    }

    @GetMapping("/") no usages ▲ Weder Sousa
    public String joke(@RequestParam(value = "message", defaultValue = "Tell me a dad joke about dogs") String message) {
        return chatClient.prompt() ChatClientRequest
            .user(message)
            .call() CallResponseSpec
            .content(); // short for getResult().getOutput().getContent();
    }

    @GetMapping("/jokes-by-topic") no usages ▲ Weder Sousa
    public String jokesByTopic(@RequestParam String topic) {
        return chatClient.prompt() ChatClientRequest
            .user(u -> u.text("Tell me a joke about {topic}").param( k: "topic",topic))
            .call() CallResponseSpec
            .content();
    }

    @GetMapping("jokes-with-response") no usages ▲ Weder Sousa
    public ChatResponse jokeWithResponse(@RequestParam(value = "message", defaultValue = "Tell me a dad joke about computers") String message) {
        return chatClient.prompt() ChatClientRequest
            .user(message)
            .call() CallResponseSpec
            .chatResponse();
    }
}
```

```
import org.springframework.ai.image.ImageModel;
import org.springframework.ai.image.ImagePrompt;
import org.springframework.ai.image.ImageResponse;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController no usages ▲ Weder Sousa
@RequestMapping("/generate")
public class GenerateImageController {

    private final ImageModel imageModel; 2 usages

    public GenerateImageController(ImageModel imageModel) { this.imageModel = imageModel; }
```

```
@GetMapping("/image") no usages ▲ Weder Sousa
public ImageResponse generateImageResponse(@RequestParam String description) {
    return imageModel.call(
        new ImagePrompt(description));
}
```

```
weder@weder:~$ http get :8080/generate/image?description="crie uma imagem de um cachorrinho chupando picolé" -b --pretty none
{"metadata":{}, "result": {"metadata": {"revisedPrompt": "Create an image of a small puppy enjoying a delicious popsicle on a warm sunny day. The puppy could be of any breed, with soft, fluffy fur that shines in the sunlight. Its small tongue is going after the frozen treat, which is melting a little from the heat. The background features a backyard scene filled with bright green grass and colorful flowers."}, "output": {"url": "https://oaidalleapiproscus.blob.core.windows.net/private/org-aA500E00spbN2sHDJ606gTLZ/user-XkPx8AYpBTILok3KaTQDWIK4/img-2Wg4Fw0Cj0vFRDlnEZmYb8X0.png?st=2024-09-18T20%3A16%3A07Z&se=2024-09-18T22%3A16%3A07Z&sp=r&sv=2024-08-04&sr=b&rscd=inline&rsct=image/png&skoid=d505667d-d6c1-4a0a-bac7-5c84a87759f8&sktid=a48cca56-e6da-484e-a814-9c849652bcb3&skt=2024-09-17T23%3A54%3A17Z&ske=2024-09-18T23%3A54%3A17Z&sks=b&skv=2024-08-04&sig=Llf2vAd0Zj4n6%2B8e8JgRd1NWmEn9PionN2RhXdk2Nc%3D", "b64Json": null}}, "results": [{"metadata": {"revisedPrompt": "Create an image of a small puppy enjoying a delicious popsicle on a warm sunny day. The puppy could be of any breed, with soft, fluffy fur that shines in the sunlight. Its small tongue is going after the frozen treat, which is melting a little from the heat. The background features a backyard scene filled with bright green grass and colorful flowers."}, "output": {"url": "https://oaidalleapiproscus.blob.core.windows.net/private/org-aA500E00spbN2sHDJ606gTLZ/user-XkPx8AYpBTILok3KaTQDWIK4/img-2Wg4Fw0Cj0vFRDlnEZmYb8X0.png?st=2024-09-18T20%3A16%3A07Z&se=2024-09-18T22%3A16%3A07Z&sp=r&sv=2024-08-04&sr=b&rscd=inline&rsct=image/png&skoid=d505667d-d6c1-4a0a-bac7-5c84a87759f8&sktid=a48cca56-e6da-484e-a814-9c849652bcb3&skt=2024-09-17T23%3A54%3A17Z&ske=2024-09-18T23%3A54%3A17Z&sks=b&skv=2024-08-04&sig=Llf2vAd0Zj4n6%2B8e8JgRd1NWmEn9PionN2RhXdk2Nc%3D", "b64Json": null}}]}
```

2



3





2

The image shows a workspace that includes a laptop (an ASUS model), a smartphone, a pen, a notebook or planner, and a coffee mug filled with coffee. Additionally, there are several baby toys and accessories scattered around, including a pacifier, a plush toy, colorful rings, and a small wooden toy with wheels.
The setting suggests a combination of work and childcare activities in the same space.

3

A imagem mostra um espaço de trabalho que inclui um laptop (um modelo ASUS), um smartphone, uma caneta, um caderno ou agenda e uma caneca de café cheia de café.
Além disso, há vários brinquedos e acessórios de bebê espalhados, incluindo uma chupeta, um brinquedo de pelúcia, anéis coloridos e um pequeno brinquedo de madeira com rodas. O cenário sugere uma combinação de trabalho e atividades de cuidados infantis no mesmo espaço.

```
import org.springframework.ai.chat.client.ChatClient;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.core.io.Resource;
import org.springframework.util.MimeTypeUtils;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
```

```
import java.io.IOException;
```

```
@RestController no usages ▲ Weder Sousa
@RequestMapping("/detection") ←
public class ImageDetectionController {
```

```
    private final ChatClient chatClient; 2 usages
    @Value("classpath:/images/sincerely-media-2ULZpdNzn2w-unsplash.jpg") 1 usage
    Resource sampleImage;

    public ImageDetectionController(ChatClient.Builder builder) { no usages ▲ Weder Sousa
        this.chatClient = builder.build();
    }
```

```
@GetMapping("/image-to-text") no usages ▲ Weder Sousa
public String image() throws IOException {
    return chatClient.prompt()
        .user(u -> u
            .text("Can you please explain what you see in the following image?")
            .media(MimeTypeUtils.IMAGE_JPEG, sampleImage)
        )
        .call() CallResponseSpec
        .content();
```

1

```
@RestController 1 usage  Weder Sousa
@RequestMapping("/olympics")
public class OlympicsController {

    private static final Logger log = LoggerFactory.getLogger(OlympicsController.class);
    private final ChatClient chatClient; 2 usages
    @Value("classpath:/docs/olympic-sports.txt") 1 usage
    private Resource docsToStuffResource;
    @Value("classpath:/prompts/olympic-sports.st") 1 usage
    private Resource olympicSportsResource;

    public OlympicsController(ChatClient.Builder builder) { this.chatClient = builder.build(); }

    @GetMapping("/2024") no usages  Weder Sousa
    public String get2024OlympicSports(
        @RequestParam(value = "message", defaultValue = "What sports are being included in the 2024 Summer Olympics?") String message,
        @RequestParam(value = "stuffit", defaultValue = "false") boolean stuffit
    ) throws IOException {
        String sports = docsToStuffResource.getContentAsString(Charset.defaultCharset());
        log.info("Sports: {}", sports);
        return chatClient.prompt() ChatClientRequest
            .user(u -> {
                u.text(olympicSportsResource);
                u.param(k: "question", message);
                u.param(k: "context", stuffit ? sports : "");
            })
            .call() CallResponseSpec
            .content();
    }
}
```

1 Use the following pieces of context to answer the question at the end.
If you don't know the answer just say "I'm sorry but I don't know the answer to that".
{context}
Question: {question}

1

2 Archery, athletics, badminton, basketball , basketball 3x3,
boxing,canoe slalom, canoe sprint, road cycling, cycling track,
mountain biking,BMX freestyle, BMX racing, equestrian, fencing,
football, golf,artistic gymnastics, rhythmic gymnastics,
trampoline, handball, hockey,judo, modern pentathlon, rowing, rugby,
sailing, shooting, table tennis,taekwondo, tennis, triathlon,
volleyball, beach volleyball,diving, marathon swimming, artistic swimming,
swimming,water polo, weightlifting,wrestling,breaking,
sport climbing, skateboarding, and surfing.

2

3 I'm sorry but I don't know the answer to that.

3

```
@RestController no usages ▲ Weder Sousa
@RequestMapping("/youtube")
public class YouTubePromptController {

    private final ChatClient chatClient; 3 usages
    @Value("classpath:/prompts/youtube.st") 1 usage
    private Resource ytPromptResource;

    public YouTubePromptController(ChatClient.Builder builder) { this.chatClient = builder.build(); }

    @GetMapping("/popular") no usages ▲ Weder Sousa
    public String findPopularYouTubersStepOne(@RequestParam(value = "genre", defaultValue = "tech") String genre) {
        String message = """
            List 10 of the most popular YouTubers in {genre} along with their current subscriber counts. If you don't know
            the answer, just say "I don't know".
            """;
        return chatClient.prompt() ChatClientRequest
            .user(u -> u.text(message).param(k: "genre", genre))
            .call() CallResponseSpec
            .content();
    }

    @GetMapping("/popular-resource") no usages ▲ Weder Sousa
    public String findPopularYouTubers(@RequestParam(value = "genre", defaultValue = "tech") String genre) {
        return chatClient.prompt() ChatClientRequest
            .user(u -> u.text(ytPromptResource).param(k: "genre", genre))
            .call() CallResponseSpec
            .content();
    }
}
```

1

I don't have real-time access to current data, so I can't provide the exact subscriber counts for YouTubers. However, I can certainly list some of the most popular tech YouTubers as of my last update:

1. Marques Brownlee (MKBHD)
2. Linus Tech Tips
3. Unbox Therapy
4. Dave Lee (Dave2D)
5. Austin Evans
6. Jonathan Morrison (TLD)
7. iJustine
8. The Verge
9. Android Authority
10. Mrwhosetheboss

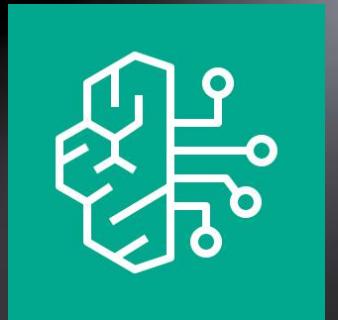
For the most up-to-date subscriber counts, I recommend visiting their YouTube channels directly.

2



Topic 4: Integration: Spring AI with AWS (Bedrock)





Amazon Bedrock

DEMO

Getting started



Sign in as IAM user

Account ID (12 digits) or account alias

IAM user name

Password

Remember this account

Sign in

[Sign in using root user email](#)

[Forgot password?](#)

Manage less and discover the value of Amazon RDS

5-month payback period, 3-year ROI of 258%, and 63% more efficient DBA teams

[Learn more >](#)

English ▾

Identity and Access Management (IAM)

1

Search IAM

Dashboard

Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

2

Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Create access key

3

AKIAI
[REDACTED]

Description

-

Last used

12 minutes ago

Last used region

us-east-1

Status

Active

Created

43 days ago

Last used service

sts

Actions ▾

Amazon Bedrock

The easiest way to build and scale generative AI applications with foundation models (FMs)

Try Bedrock

Get started

1

▼ Assessment

Model Evaluation

User guide 

Bedrock Service Terms 

▼ Bedrock configurations

Model access

Bedrock Studio 

Settings

2

What is Model access?

To use Bedrock, account users with the correct IAM Permissions  must enable access to available Bedrock foundation models (FMs). View all Bedrock Model Terms  for Bedrock FMs.

[Modify model access](#)

1

Visit [Amazon Bedrock Quotas](#)  for a quick guide to the default quotas and limits that apply to Amazon Bedrock.

Base models (34)

Not seeing a model you're interested in? Check out all supported models by region [here](#) .

Find model

Models

▼ AI21 Labs (3)

Jamba-Instruct

Jurassic-2 Ultra

Jurassic-2 Mid

▼ Amazon (8)

Titan Embeddings G1 - Text

Titan Text G1 - Lite

Titan Text G1 - Express

Titan Image Generator G1 v2

Titan Image Generator G1

Titan Multimodal Embeddings G1

2

Access status

3/3 access granted

 Access granted

 Access granted

 Access granted

8/8 access granted

 Access granted

 Access granted

 Access granted

 Access granted

 Access granted

 Access granted

3

4

Modality

Text

Text

Text

Embedding

Text

Text

Image

Image

Embedding



```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.3.2</version>
    <relativePath/> <!-- lookup parent from repository -->
  </parent>
  <groupId>dev.wsousa</groupId>
  <artifactId>spring-ai-aws-bedrock-ai</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>spring-ai-aws-bedrock-ai</name>
  <description>Demo project for Spring Boot</description>
  <properties>
    <java.version>21</java.version>
    <spring-ai.version>1.0.0-M2</spring-ai.version>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
      <artifactId>spring-ai-bedrock-ai-spring-boot-starter</artifactId>
      <groupId>org.springframework.ai</groupId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>software.amazon.awssdk</groupId>
      <artifactId>aws-sdk-java</artifactId>
      <version>2.27.18</version>
      <scope>provided</scope>
    </dependency>
  </dependencies>
```



1

2

application.properties

```
1  spring.application.name=spring-ai-aws-bedrock
2
3  spring.ai.bedrock.aws.region=us-east-1
4
5  spring.ai.bedrock.aws.access-key=${ACCESS_KEY}
6  spring.ai.bedrock.aws.secret-key=${SECRET_KEY}
7  #spring.ai.bedrock.anthropic3.chat.enabled=true
8
9  #Enable or disable support for Titan embedding
10 spring.ai.bedrock.titan.embedding.enabled = true
11 #The model id to use. See the TitanEmbeddingModel for the supported models.
12 spring.ai.bedrock.titan.embedding.model = amazon.titan-embed-image-v1
13
14 spring.ai.bedrock.titan.chat.enabled=true
15 spring.ai.bedrock.titan.chat.options.temperature=0.8
16
17 spring.ai.bedrock.aws.timeout=10m
18
19 weather.api-key=3e72f2bcababfe07692ed64856195eb
20 weather.api-url=http://api.weatherapi.com/v1
```

1

Project

spring-ai-aws-bedrock [spring-ai-bedrock] ~/Documents

- > .idea
- > .mvn
- > scripts
- src
 - main
 - java
 - dev.wsousa
 - controller
 - ActorController
 - BedrockSDKController
 - ChatController
 - CityWeatherFunctionController
 - GenerateImageController
 - ImageDetectionController
 - OlympicsController
 - StatefulMemoryController
 - StreamController
 - TreasurePromptController
 - YouTubePromptController
 - controllerRag
 - functions
 - response
 - service
 - Application
 - > resources
 - > test
 - > target
 - gitignore
 - mvnw

© ActorController.java ×

```
15     public class ActorController {  
16  
17         private final ChatClient chatClient; 5 usages  
18  
19     @  
20         public ActorController(ChatClient.Builder builder) { no usages ▲ Weder Sousa  
21             this.chatClient = builder  
22                 .build();  
23     }  
24  
25         @GetMapping("/films-string") no usages ▲ Weder Sousa  
26         public String getActorFilmsString() {  
27             return chatClient.prompt() ChatClientRequestSpec  
28                 .user( text: "Generate a filmography for a Anthony Hopkins for the year 2010." )  
29                 .call() CallResponseSpec  
30                 .content();  
31     }  
32  
33         @GetMapping("/films") no usages ▲ Weder Sousa  
34         public ActorFilms getActorFilms() {  
35             return chatClient.prompt() ChatClientRequestSpec  
36                 .user( text: "Generate a filmography for a Anthony Hopkins." )  
37                 .call() CallResponseSpec  
38                 .entity(ActorFilms.class);  
39     }  
40  
41         @GetMapping("/films-list") no usages ▲ Weder Sousa *  
42         public List<ActorFilms> listActorFilms() {  
43             return chatClient.prompt() ChatClientRequestSpec  
44                 .user( text: "Generate a filmography for the actors Denzel Washington and Tom Hanks" )  
45                 .call() CallResponseSpec  
46                 .entity( new ParameterizedTypeReference<>() {} ); ▲ Weder Sousa  
47         }
```

1

2

Project ▾

spring-ai-aws-bedrock [spring-ai-bedrock] ~/Documents

- > .idea
- > .mvn
- > scripts
- src
 - main
 - java
 - dev.wsousa
 - controller
 - ActorController
 - BedrockSDKController
 - ChatController
 - CityWeatherFunctionController
 - GenerateImageController
 - ImageDetectionController
 - OlympicsController
 - StatefulMemoryController
 - StreamController
 - TreasurePromptController
 - YouTubePromptController
 - controllerRag
 - functions
 - response
 - service
 - Application
 - resources
 - test
 - target
 - .gitignore
 - mvnw
 - mvnw.cmd

© ActorController.java © BedrockSDKController.java © BedrockSDKService.java © BedrockSDKServiceImp.java © ChatController.java

```
7 import org.springframework.ai.embedding.EmbeddingRequest;
8 import org.springframework.ai.embedding.EmbeddingResponse;
9 import org.springframework.web.bind.annotation.GetMapping;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.bind.annotation.RequestParam;
12 import org.springframework.web.bind.annotation.RestController;
13
14 import java.util.List;
15
16 @RestController no usages ▲ Weder Sousa
17 @RequestMapping("/generate")
18 public class GenerateImageController {
19     private final EmbeddingModel embeddingModel; 2 usages
20     private final BedrockSDKTitanService bedrockSDKTitanService; 2 usages
21
22     public GenerateImageController(EmbeddingModel embeddingModel, BedrockSDKTitanService bedrockSDKTitanService) {
23         this.embeddingModel = embeddingModel;
24         this.bedrockSDKTitanService = bedrockSDKTitanService;
25     }
26
27     @GetMapping("/embeddingModel") no usages ▲ Weder Sousa
28     public EmbeddingResponse image(@RequestParam(value = "prompt") String prompt) {
29         return embeddingModel.call(
30             new EmbeddingRequest(List.of(prompt),
31                 BedrockTitanEmbeddingOptions.builder()
32                     .withInputType(BedrockTitanEmbeddingModel.InputType.TEXT)
33                     .build()));
34     }
35
36     @GetMapping("/image-titan") no usages ▲ Weder Sousa
37     public String imageTitan(@RequestParam(value = "prompt") String prompt) {
38         return bedrockSDKTitanService.invokeModel(prompt);
39     }
40 }
```

1

Project

spring-ai-aws-bedrock [spring-ai-bedrock] ~/Documents

.idea
.mvn
scripts
src
main
java
dev.wsousa
controller
controllerRag
functions
response
service
impl
BedrockSDKServiceImp
BedrockSDKTitanServiceImp
BedrockSDKService
BedrockSDKTitanService
Application
resources
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
README.md
External Libraries
Scratches and Consoles

BedrockSDKTitanServiceImp.java

```
import java.security.SecureRandom;
@Service no usages + Weder Sousa *
public class BedrockSDKTitanServiceImp implements BedrockSDKTitanService {
    public String invokeModel(String prompt) { 1 usage + Weder Sousa *
        // Create a Bedrock Runtime client in the AWS Region you want to use.
        // Replace the DefaultCredentialsProvider with your preferred credentials provider.
        var client = BedrockRuntimeClient.builder()
            .credentialsProvider(DefaultCredentialsProvider.create())
            .region(Region.US_EAST_1)
            .build();

        // Set the model ID, e.g., Titan Image G1.
        var modelId = "amazon.titan-image-generator-v1";
        var nativeRequestTemplate = """
            {
                "taskType": "TEXT_IMAGE",
                "textToImageParams": { "text": "{{prompt}}" },
                "imageGenerationConfig": { "seed": {{seed}} }
           }""";

        // Define the prompt for the image generation.
        // Get a random 31-bit seed for the image generation (max. 2,147,483,647).
        var seed = new BigInteger( numBits: 31, new SecureRandom() );

        // Embed the prompt and seed in the model's native request payload.
        var nativeRequest = nativeRequestTemplate.replace( target: "{{prompt}}", prompt).replace( target: "{{seed}}", seed.toString());

        try {
            // Encode and send the request to the Bedrock Runtime.
            var response = client.invokeModel(request -> request.body(SdkBytes.fromUtf8String(nativeRequest)).modelId(modelId));
            // Decode the response body.
            var responseBody = new JSONObject(response.body().asUtf8String());
            // Retrieve the generated image data from the model's response.
            var base64ImageData = new JSONPointer("/images/0").queryFrom(responseBody).toString();
            return base64ImageData;
        } catch (SdkClientException e) {
    }
```

Next Steps



Getting started



**Get started with
Amazon Bedrock**



**Discover features with
a step-by-step tutorial**



**Dive deep with a
hands-on workshop**

GenAI Roadshow - Virtual



Amazon Web Services Latin America



- [/aws.amazon.com](http://aws.amazon.com)
- [/amazonwebservices](#)
- [/AWSLatam](#)
- [/AWSBrasil](#)
- [/company/amazon-web-services](#)



Resources

#Links (BedRock)

<https://www.ranthebuilder.cloud/post/automating-api-calls-with-agents-for-amazon-bedrock-with-powertools>
<https://www.slightinsight.com/tech/developing-a-spring-boot-application-with-amazon-bedrock-api/>
<https://community.aws/content/2dhKdwY1kzhFTg9CTLbaJ9MmTN/build-generative-ai-applications-with-amazon-bedrock>
https://www.eficode.com/blog/building-ai-on-aws-bedrock-brings-brilliant-building-blocks?utm_campaign=AWS&utm_content=189323971&utm_medium=social&utm_source=twitter&hss_channel=tw-142208607
<https://cloudacademy.com/learning-paths/integrating-aws-services-with-langs-and-other-fms-14068/>
https://www.youtube.com/watch?v=CE_zrMvcuk&list=PLhr1KZpdzukfmv7jxvB0rL8SWoycA9TIM&index=5
<https://programadriano.medium.com/conhecendo-o-amazon-bedrock-c687c7e9777f>
<https://github.com/AWS-Cloud-Drops-Builders-Edition/show?tab=readme-ov-file>
<https://levelup.gitconnected.com/ai-powered-video-summarizer-with-amazon-bedrock-and-anthropic-claude-9f1832f397dc>
<https://catalog.workshops.aws/persona-based-access-genai-application/en-US/04-testing-application/01-updating-code>
<https://dev.to/aws-builders/have-fun-with-aws-partyrock-3755>
<https://dev.to/aws-builders/build-serverless-generative-ai-api-service-with-aws-lambda-and-amazon-bedrock-3abc>
<https://norahsakal.medium.com/how-to-use-aws-titans-ai-multimodal-embeddings-for-better-e-commerce-recommendations-b9f4adb60c02>

#Importante

<https://www.linkedin.com/pulse/aplica%C3%A7%C3%B5es-serverless-llm-com-amazon-bedrock-diogo-santos-yjo3f/>
<https://aws.amazon.com/pt/blogs/aws-brasil/crie-aplicativos-de-ia-generativa-usando-o-aws-step-functions-e-o-amazon-bedrock/>
<https://community.aws/generative-ai>
<https://aws.amazon.com/pt/what-is/retrieval-augmented-generation/>

Resources

#Links Spring AI OpenAI

<https://github.com/danvega/hello-gpt/blob/main/src/main/java/dev/danvega/Application.java>

https://www.youtube.com/watch?v=yyvjT0v3lpY&list=PLZV0a2jwt22uoDm3LNDFvN6i2cAVU_HTH

<https://tecnoblog.net/guias/4-sites-gratis-para-transcrever-video-do-youtube/>

#Spring AI Vector

<https://www.youtube.com/watch?v=azKntWC6d3w>

#Top

<https://github.com/kousen/openaidemo>

<https://www.youtube.com/watch?v=ZeH3bBKdqRU>

About the Speaker

Q & A

Weder Mariano de Sousa

Specialist Senior Java - GFT

Technician **System Development**

Graduated **Computer Science**

Post Graduate in **Midias UFG**

Post Graduate in **Information Security**



- <https://www.linkedin.com/in/wedermarianodesousa/>
- <https://github.com/weder96>
- <https://twitter.com/weder96>
- <https://dev.to/weder96>



GOJava



AWS User Group Goiânia



THANK
YOU