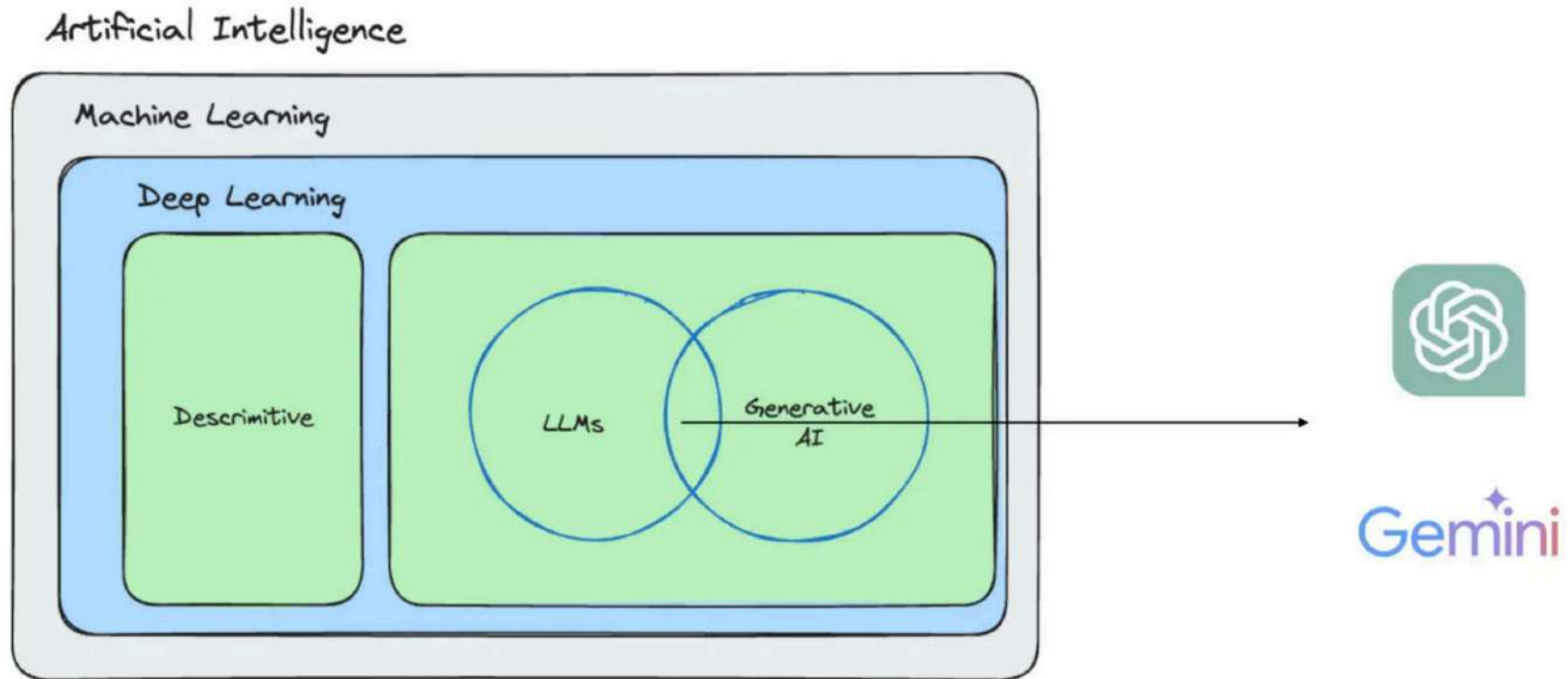


What is Artificial Intelligence (AI)?



Examples ...

- Query data with natural language
 - *"Why did the user aks for their account to be closed?"*
 - Checks emails, complaints, logs, customer support tickets, social media etc.

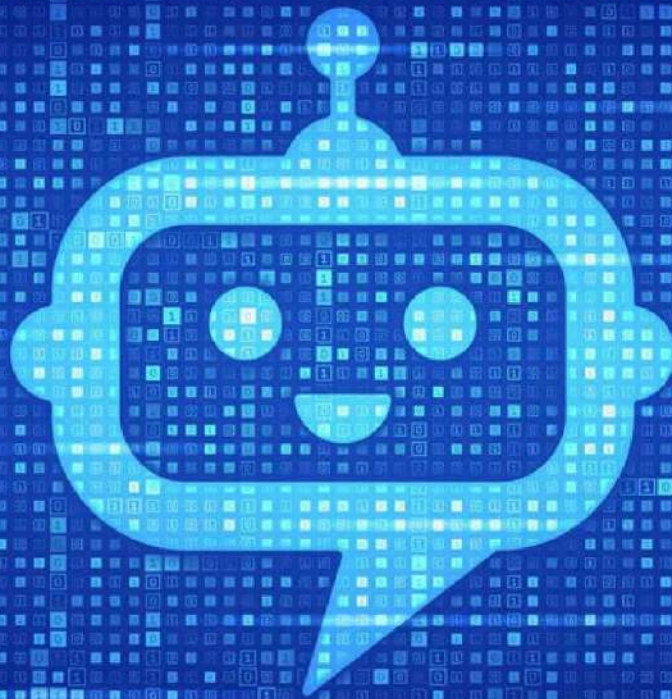
Examples ...

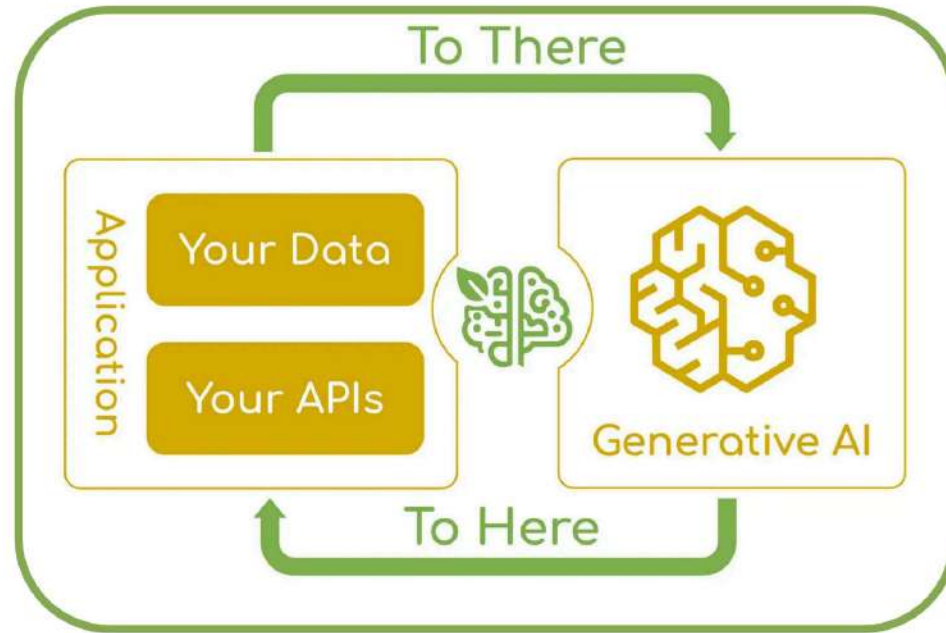
- Perform aggregation and transformation
 - Summarize text
 - Extract action items
 - Censor/identify sensitive data or abuse
 - Translate to different languages

Examples ...

- Manipulate data
 - *"Reactivate customer account and notify them by email."*
- Define and execute business rules
 - *"If customer has not logged in for 6 months, send a reminder."*
- Generate code
 - *"Create a CSV exporter for all orders and write tests for it."*
- ...

Not all prompts are/need user input!





This is an integration problem!

Rest APIs Make AI Accessible

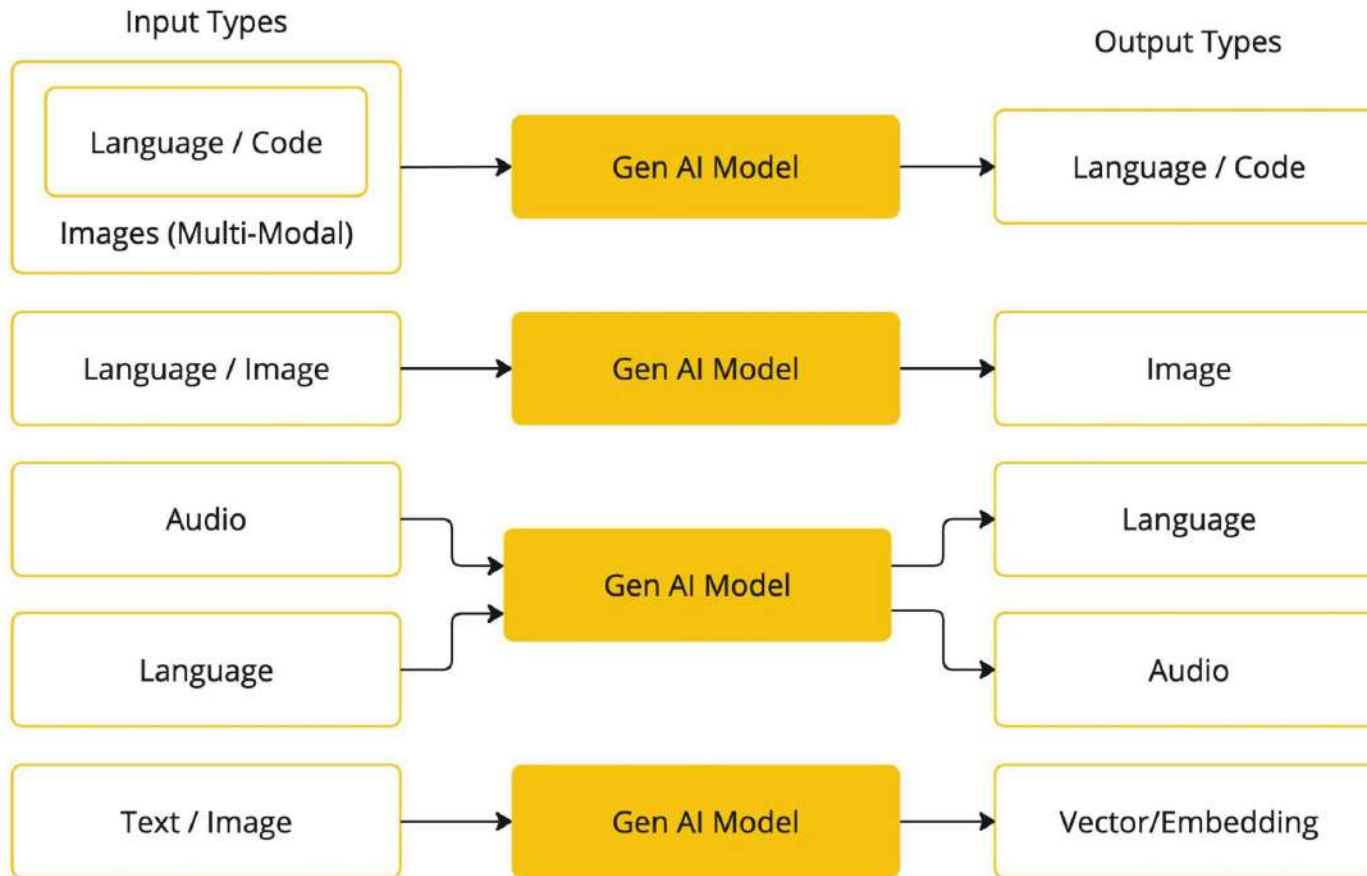


```
#!/usr/bin/env bash

echo "Calling OpenAI API ..."

PROMPT="Tell me a friendly developer joke about JAVA."

curl https://api.openai.com/v1/chat/completions \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer $SPRING_AI_OPENAI_API_KEY" \
  --data \
  "{
    \"model\": \"gpt-3.5-turbo\",
    \"messages\": [
      {
        \"role\": \"system\",
        \"content\": \"You are a friendly chatbot and you like to place emojis everywhere.\"
      },
      {
        \"role\": \"user\",
        \"content\": \"$PROMPT\"
      }
    ]
  }"
```

See: [Spring AI: Models](#)

LLM Leaderboard - Comparison of GPT-4o, Llama 3, Mistral, Gemini and over 30 models

Comparison and ranking the performance of over 30 AI models (LLMs) across key metrics including quality, price, performance and speed (output speed – tokens per second & latency - TTFT), context window & others. For more details including relating to our methodology, see our FAQs.

For comparison of API Providers hosting the models see [LLM API Providers Leaderboard](#)

HIGHLIGHTS

- Quality:** GPT-4o (Aug 6) and Claude 3.5 Sonnet are the highest quality models, followed by GPT-4o & GPT-4 Turbo.
- Output Speed (tokens/s):** Gemma 7B (939 t/s) and Gemini 1.5 Flash (210 t/s) are the fastest models, followed by Llama 3.1 8B & AI21 Jamba 1.5 Mini.
- Latency (seconds):** Sonar Small (0.19s) and Sonar 3.1 Small (0.19s) are the lowest latency models, followed by Sonar 3.1 Large & Sonar Large.
- Price (\$ per M tokens):** OpenChat 3.5 (\$0.13) and Gemini 1.5 Flash (\$0.13) are the cheapest models, followed by Gemma 7B & Mistral 7B.
- Context Window:** Gemini 1.5 Pro (2m) and Gemini 1.5 Flash (1m) are the largest context window models, followed by Codestral-Mamba & AI21 Jamba 1.5 Large.

PROMPT OPTIONS

Filter, e.g. GPT, Meta

EXPAND COLUMNS

	CONTEXT ↗↘		QUALITY ↗↘	PRICE ↗↘	OUTPUT TOKENS/S ↗↘	LATENCY ↗↘		
MODEL ↕	CREATOR ↕	CONTEXT WINDOW ↕	INDEX <i>Normalized avg</i> ↕	BLENDED <i>USD/1M Tokens</i> ↕	MEDIAN <i>Tokens/s</i> ↕	MEDIAN <i>First Chunk (s)</i> ↕	FURTHER ANALYSIS	
GPT-4o (Aug 6)	OpenAI	128k	77	\$4.38	106.0	0.40	🔗 Model	🔗 Providers
GPT-4o	OpenAI	128k	77	\$7.50	103.2	0.40	🔗 Model	🔗 Providers
GPT-4o mini	OpenAI	128k	71	\$0.26	122.9	0.41	🔗 Model	🔗 Providers
Llama 3.1 405B	Meta	128k	72	\$5.00	28.8	0.67	🔗 Model	🔗 Providers
Llama 3.1 70B	Meta	128k	65	\$0.90	53.0	0.45	🔗 Model	🔗 Providers
Llama 3.1 8B	Meta	128k	53	\$0.16	174.3	0.35	🔗 Model	🔗 Providers
Gemini 1.5 Pro	Google	2m	72	\$5.25	61.6	0.92	🔗 Model	🔗 Providers
Gemini 1.5 Flash	Google	1m	60	\$0.13	209.9	0.39	🔗 Model	🔗 Providers
Gemma 2 27B	Google	8k	49	\$0.80	68.6	0.43	🔗 Model	🔗 Providers
Gemma 2 9B	Google	8k	47	\$0.20	126.1	0.33	🔗 Model	🔗 Providers
Claude 3.5 Sonnet	ANTHROPIC	200k	77	\$6.00	88.0	1.04	🔗 Model	🔗 Providers
Claude 3 Opus	ANTHROPIC	200k	70	\$30.00	26.6	1.76	🔗 Model	🔗 Providers

LMSYS Chatbot Arena Leaderboard

[Vote!](#)

[Blog](#) |
 [GitHub](#) |
 [Paper](#) |
 [Dataset](#) |
 [Twitter](#) |
 [Discord](#) |
 [Kaggle Competition](#)

LMSYS Chatbot Arena is a crowdsourced open platform for LLM evals. We've collected over 1,000,000 human pairwise comparisons to rank LLMs with the Bradley-Terry model and display the model ratings in Elo-scale. You can find more details in our paper. **Chatbot arena is dependent on community participation, please contribute by casting your vote!**

NEWS: We got a shorter URL! Reach us via lmarena.ai

[Arena](#)
[NEW! Overview](#)
[NEW! Arena \(Vision\)](#)
[Arena-Hard-Auto](#)
[Full Leaderboard](#)

Total #models: 136. Total #votes: 1,762,122. Last updated: 2024-08-27.

NEW! View leaderboard for different categories (e.g., math, coding)! This is still in preview and subject to change.

Code to recreate leaderboard tables and plots in this [notebook](#). You can contribute your vote at chat.lmsys.org/

Category		Overall Questions					
Overall		#models: 136 (100%) #votes: 1,762,122 (100%)					
Rank	Model	Arena Score	95% CI	Votes	Organization	License	Knowledge Cutoff
1	ChatGPT-4o-latest (2024-08-08)	1316	+4/-4	24023	OpenAI	Proprietary	2023/10
2	Gemini-1.5-Pro-Exp-0827	1301	+5/-5	19919	Google	Proprietary	2023/11
2	Gemini-1.5-Pro-Exp-0801	1298	+4/-4	25211	Google	Proprietary	2023/11
2	Grok-2-08-13	1295	+6/-6	10019	xAI	Proprietary	2024/3
5	GPT-4o-2024-05-13	1286	+3/-2	82934	OpenAI	Proprietary	2023/10
6	GPT-4o-mini-2024-07-18	1274	+4/-4	23147	OpenAI	Proprietary	2023/10
6	Gemini-1.5-Flash-Exp-0827	1271	+7/-6	6282	Google	Proprietary	2023/11
6	Claude 3.5 Sonnet	1270	+3/-3	53352	Anthropic	Proprietary	2024/4
6	Gemini Advanced App (2024-05-14)	1266	+3/-3	52225	Google	Proprietary	Online
6	Meta-Llama-3.1-405B-Instruct	1266	+3/-5	24584	Meta	Llama 3.1 Community	2023/12
6	Grok-2-Mini-08-13	1265	+6/-5	10791	xAI	Proprietary	2024/3
7	GPT-4o-2024-08-06	1262	+5/-5	14886	OpenAI	Proprietary	2023/10
10	Gemini-1.5-Pro-001	1259	+3/-3	74660	Google	Proprietary	2023/11
11	Gemini-1.5-Pro-Preview-0409	1257	+3/-3	55606	Google	Proprietary	2023/11
12	GPT-4-Turbo-2024-04-09	1257	+3/-3	88005	OpenAI	Proprietary	2023/12

<https://lmarena.ai/>

Integrating Custom Data & APIs

How can you equip the AI model with information on which it has not been trained? GPT-3.5/4.0 only has knowledge up until June 2024*, limiting its ability to answer questions about more recent data.

* Why not asking Chat GPT with a prompt?

You said:

Can you tell me the date of the most recent information in your training data?

ChatGPT said:

My most recent training data goes up until June 2024. However, I can fetch up-to-date information from the web if needed. Let me know how I can help!

Limitations of AI Models

AI models:

- Are trained on public knowledge up to a certain date
- They don't know about your private / corporate data
- Don't have access to realtime data
- Can't learn from your data

Bring Your Own Data

Ways of using your data in AI applications:

- Train new models (✗)
- Fine tune existing models (✗)
- "Stuff the prompt" (✓)
- Retrieval Augmented Generation (✓)
- Function calling (✓)

Java and AI

- Why Java & AI?
 - AI becomes ubiquitous across the IT landscape
 - Java is the language of enterprise
 - Creating Java AI apps is a new requirement
- Spring AI, LangChain4j, Semantic Kernel
 - Simplify AI integration
 - Provide a unified API
 - Support multiple AI providers

Spring AI



See Spring AI on [GitHub](#) or on [spring.io](#).

What is Spring AI?

- AI for Java developers!
- Simplifies interactions with LLMs with a Spring abstraction
 - Change model by changing one line of configuration!
- Simplifies interactions with Vector databases
- Observability
- ChatMemory
- And more ...

Spring AI Ecosystem

- Created in 2023 by Mark Pollack and Christian Tzolov
- Inspired by LangChain and LlamaIndex
- Current version: Spring AI 1.0.0-M5
 - Not in a final release version yet!
- Based on Spring Framework 6.1 and Spring Boot 3.3

Spring Initializr

Head on over to start.spring.io and select the AI Models and Vector Stores that you want to use in your new applications.

The screenshot displays the Spring Initializr web application interface. On the left sidebar, there are navigation icons for a menu and a clock. The main content area is divided into several sections:

- Project:** Includes radio buttons for `Gradle - Groovy`, `Gradle - Kotlin`, and `Maven` (which is selected).
- Spring Boot:** Includes radio buttons for `3.4.0 (SNAPSHOT)`, `3.4.0 (M2)`, `3.2.10 (SNAPSHOT)`, and `3.2.9`.
- Project Metadata:** A form with fields for `Group` (value: `com.example`), `Artifact` (value: `demo`), `Name` (value: `demo`), `Description` (value: `Demo project for Spring Boot`), `Package name` (value: `com.example.demo`), and `Packaging` (radio buttons for `jar` and `war`, with `jar` selected).
- Dependencies:** A search bar with the text `Web, Security, JPA, Actuator, Devtools...` and a hint `Press Ctrl for multiple adds`. Below the search bar, a green button labeled `AI` is visible. A list of AI-related dependencies is shown, each with a title and a description:
 - Azure OpenAI:** Spring AI support for Azure's OpenAI offering, powered by ChatGPT. It extends beyond traditional OpenAI capabilities, delivering AI-driven text generation with enhanced functionality.
 - Azure AI Search:** Spring AI vector database support for Azure AI Search. It is an AI-powered information retrieval platform and part of Microsoft's larger AI platform. Among other features, it allows users to query information using vector-based storage and retrieval.
 - Amazon Bedrock:** Spring AI support for Amazon Bedrock. It is a managed service that provides foundation models from various AI providers, available through a unified API.
 - Chroma Vector Database:** Spring AI vector database support for Chroma. It is an open-source embedding database and gives you the tools to store document embeddings, content, and metadata. It also allows to search through those embeddings, including metadata filtering.
 - Milvus Vector Database:** Spring AI vector database support for Milvus. It is an open-source vector database that has garnered

On the right side of the interface, there is a button labeled `ADD DEPENDENCIES... CTRL + B` and a settings icon in the top right corner.

Milestone Repository

Because Spring AI is still in Milestone, you need to add the Spring Milestones repository to your `pom.xml` file. From M5 onwards, they are also in Maven Central.

```
<repositories>
  <repository>
    <id>spring-milestones</id>
    <name>Spring Milestones</name>
    <url>https://repo.spring.io/milestone</url>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
  </repository>
</repositories>
```

Spring AI BOM

Add the Spring AI BOM to your `pom.xml` file and reference the version in the properties section.

```
<properties>
  <spring-ai.version>1.0.0-M5</spring-ai.version>
</properties>

<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>org.springframework.ai</groupId>
      <artifactId>spring-ai-bom</artifactId>
      <version>${spring-ai.version}</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
```

Spring Boot Starter

Add the specific Spring AI dependency to your `pom.xml` file.

```
<dependency>  
  <groupId>org.springframework.ai</groupId>  
  <artifactId>spring-ai-openai-spring-boot-starter</artifactId>  
</dependency>
```

Then, you can configure model parameters in the `application.properties` file as follows:

```
spring.ai.openai.api-key=${SPRING_AI_OPENAI_API_KEY}  
spring.ai.openai.chat.options.model=gtp-4-turbo  
...
```

Prompt Templates

Using the StringTemplate Engine

```
String message = ""
    List 10 of the most popular YouTubers in {genre}, along with their current subscriber numbers.
    If you don't know the answer, just say "I don't know".
    "";
PromptTemplate promptTemplate = new PromptTemplate(message);
Prompt prompt = promptTemplate.create(Map.of("genre", genre));
```

```
String answer = chatClient.prompt(prompt).call().content();
```

```
ChatResponse response = chatClient.prompt(prompt).call().chatResponse();
String answer = response.getResult().getOutput().getContent();
```

Chat Client

```
Joke joke = chatClient.prompt()  
    .user("Tell me a friendly developer joke about JAVA.")  
    .call()  
    .content();
```

```
Joke joke = chatClient.prompt()  
    .system("You are a friendly chatbot and you like to place emojis everywhere.")  
    .user("Tell me a friendly developer joke about JAVA.")  
    .call()  
    .entity(Joke.class);
```

```
ChatClient chatClient = chatClientBuilder  
    .defaultSystem("You are a friendly chatbot and you like to place emojis everywhere.")  
    .build();
```


Request / Response

Based on OpenAI chat model:

- Request
 - model
 - list of messages, each with a role and content
- Response
 - model
 - list of choices, each with a message
 - finish reason
 - usage, with the number of tokens used
 - ...

Working with Images

```
@Value("classpath:images/stockholm-weather.png")  
private Resource imageResourceWeather;
```

```
chatClient.prompt()  
    .user(userSpec -> userSpec  
        .text("What will be the weather like on Wednesday?")  
        .media(MimeTypeUtils.IMAGE_PNG, this.imageResourceWeather))  
    .call().content();
```



2

°C | °F

Niederschlag: 3%

Luftfeuchte: 82%

Wind: 5 m/s

Temperatur

Niederschlag

Wind

Wetter

Mittwoch, 05:00

Stark bewölkt

Working with Audio / Transcription Model

```
transcriptionModel.call(new AudioTranscriptionPrompt(audioFile)).getResult().getOutput();
```

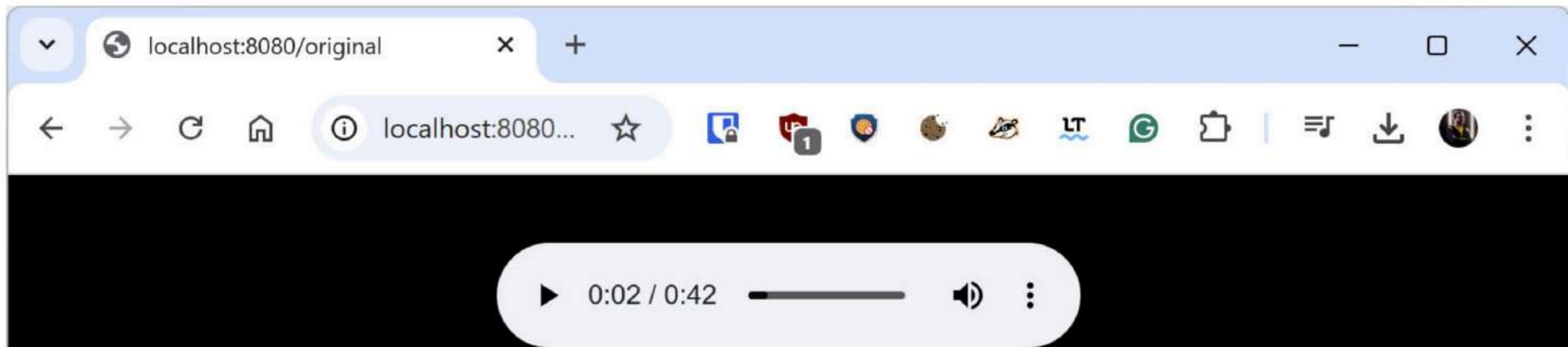
```
var transcriptionResponse = transcriptionModel.call(new AudioTranscriptionPrompt(audioFile,
    OpenAiAudioTranscriptionOptions.builder()
        .language("en")
        .prompt("Transcribe the audio file about Philipp Maloney and the missing bag.")
        .temperature(0f)
        .responseFormat(OpenAiAudioApi.TranscriptResponseFormat.VTT)
        .build()));
return transcriptionResponse.getResult().getOutput();
```

```
WEBVTT 00:00:01.000 --> 00:00:04.000 Philipp Maloney's hair-raising cases
00:00:05.000 --> 00:00:08.000 The hair-raising cases of Philipp Maloney.
00:00:09.000 --> 00:00:11.000 Today, the missing bag.
00:00:24.000 --> 00:00:28.000 The man sounded confused and excited at the same time on the phone.
00:00:28.000 --> 00:00:33.000 His voice sounded familiar, but I couldn't classify it.
00:00:33.000 --> 00:00:36.000 Only when I saw him did I recognize him.
00:00:36.000 --> 00:00:40.000 And then it was too late to run away.
00:00:40.000 --> 00:00:42.000 Do you remember me?
```

Working with Audio / Speech Model

```
speechModel.call(new SpeechPrompt(resource.getContentAsString(Charset.defaultCharset()))).getResult().getOutput();
```

```
var speechResponse = speechModel.call(new SpeechPrompt(resource.getContentAsString(Charset.defaultCharset()),  
    OpenAiAudioSpeechOptions.builder()  
        .model("tts-1")  
        .voice(OpenAiAudioApi.SpeechRequest.Voice.SHIMMER)  
        .responseFormat(OpenAiAudioApi.SpeechRequest.AudioResponseFormat.MP3)  
        .speed(1.3f)  
        .build()));  
return speechResponse.getResult().getOutput();
```



Retrieval Augmented Generation

- Bring your own data to the prompt
- Give a lot of context to the prompt
 - Text content, vectors etc.

Prompt Data Takes Precedence (1)

You said:

How many sports were there in the Paris Olympics?

ChatGPT said:

For the 2024 Paris Olympics, there are 32 sports featured, with a total of 329 events. New additions include breaking (breakdancing), and skateboarding, sport climbing, and surfing are also returning after debuting in Tokyo 2020.

Prompt Data Takes Precedence (2)

You said:

Here are the sports for the Paris Olympics: Archery, Athletics, Badminton, Basketball, Basketball 3x3, Boxing, Canoe Slalom, Canoe Sprint, Road Cycling, Track Cycling, Mountain Biking, and BMX.

How many sports were there?

ChatGPT said:

Based on the list you provided, there are 12 sports for the Paris Olympics.

Prompt Stuffing Structure

```
@Value("classpath:/olympics/context.st")  
private Resource queryTemplate;
```

Use the following pieces of context to answer the question at the end.

{context}

Question: {question}

Using a Stuffed Prompt

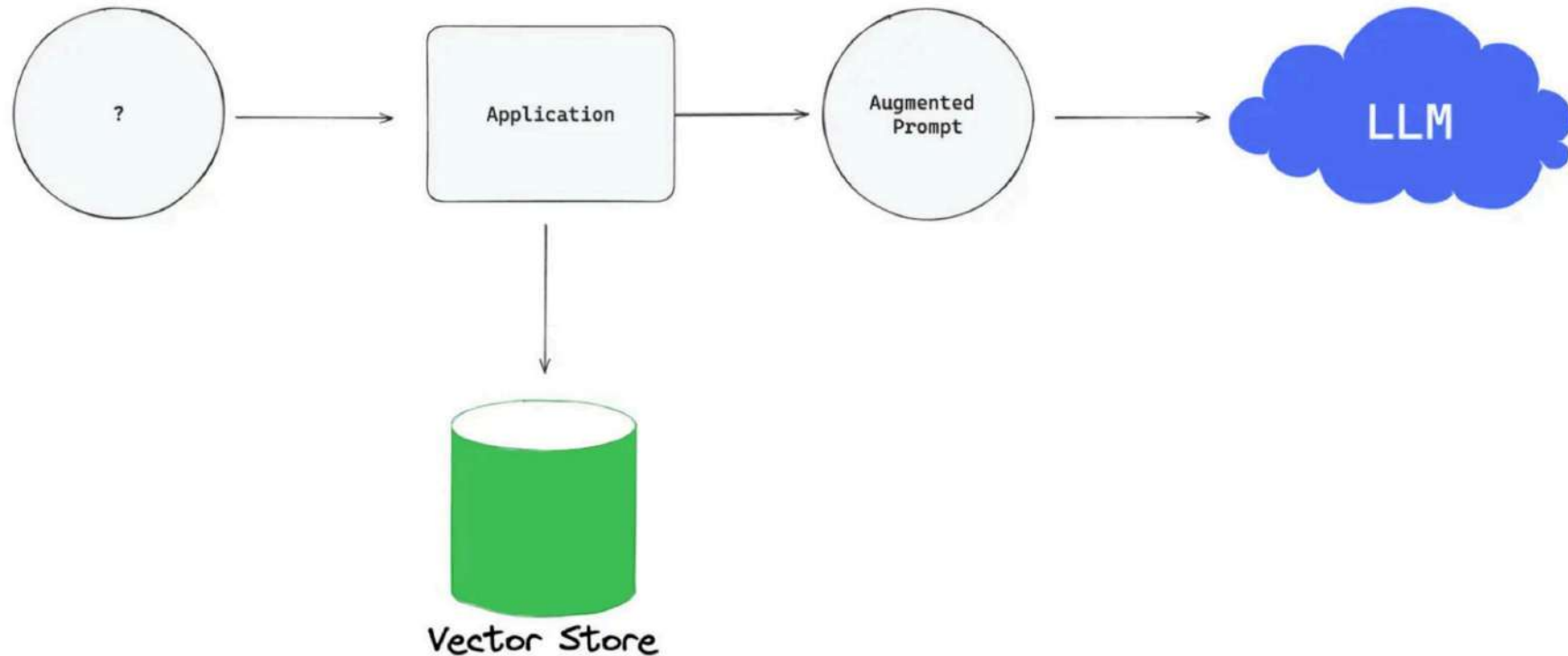
```
chatClient.prompt()  
  .user( userSpec -> userSpec.text(this.queryTemplate)  
    .param( "context" , "Archery, athletics, badminton, basketball , boxing")  
    .param( "question" , "How many sports are being included in the 2024 Summer Olympics?")  
    )  
  .call().content();
```

Use the following pieces of context to answer the question at the end.

Archery, athletics, badminton, basketball , boxing

Question: How many sports are being included in the 2024 Summer Olympics

Application Architecture - RAG



Vector Stores



Definition of a Vector

- A Vector is just a type of data
 - Typically an array of floating-point numbers
 - Commonly consists of 1'536 decimal values
 - OpenAI uses the ada002 embedding model
 - Each value typically ranges between -1 and 1

```
CREATE TABLE paragraph (  
  id SERIAL PRIMARY KEY,  
  paragraph_text TEXT,  
  vector VECTOR(1536)  
);
```

Example: {-0.345, 0.465, 0.856, ..., 0.1543}

How are Vectors created?

- Vectors typically represent the output generated by machine learning models, encapsulating the learned features or embeddings of the input data.
- The example below is based on text AI models. Vectors may also be used with computer vision AI models or audio-based AI model

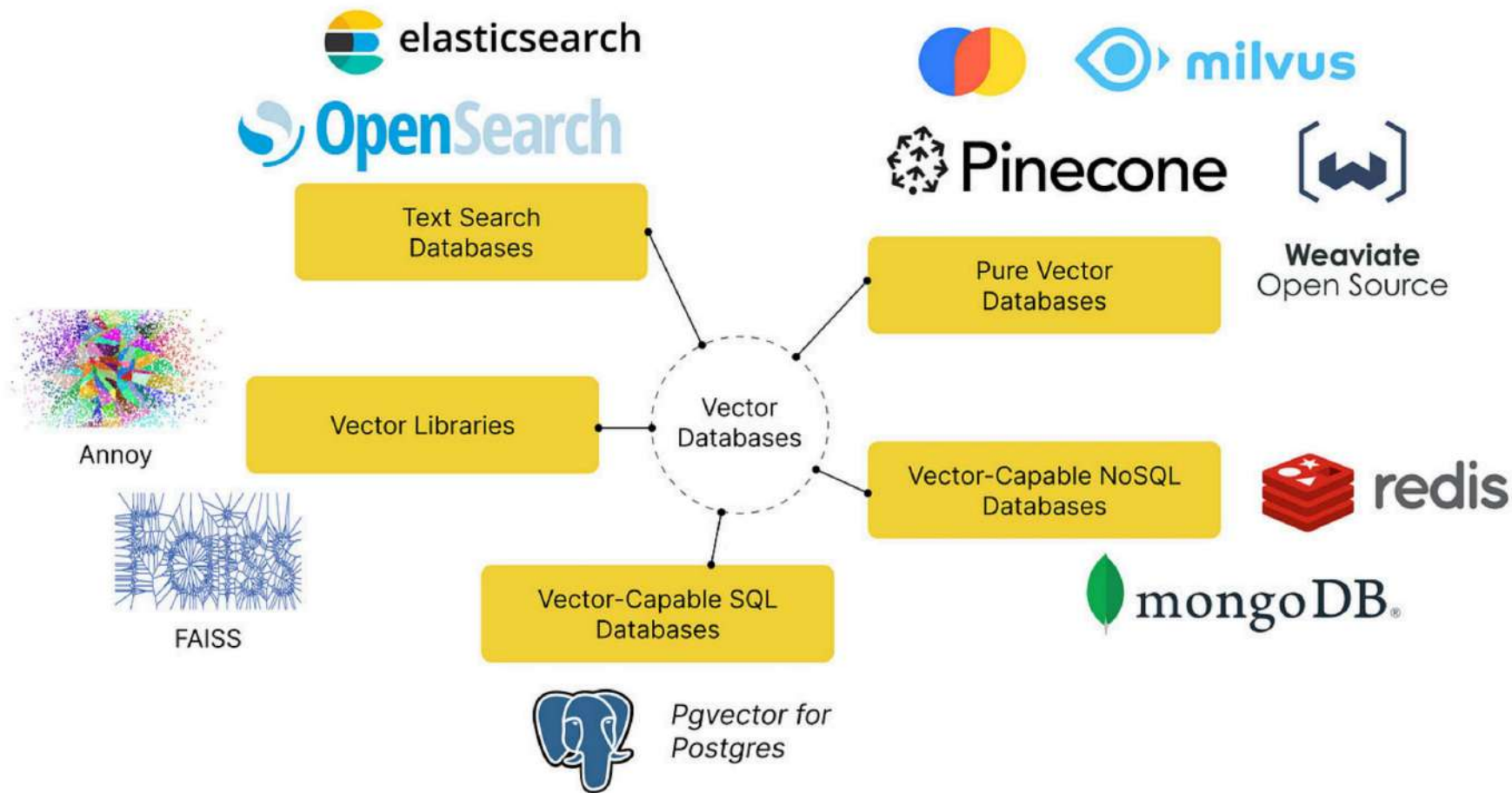
String	Vector
"My house is black"	{0.345, 0.465, 0.856, 0.1543}
"My garden is big"	{0.545, 0.665, 0.056, 0.3543}
"My dog is playful"	{0.645, 0.765, 0.156, 0.4543}

Similarity Search

- Selects the closest vectors to a given vector

$\{-0.436, 0.578, 0.935, 0.2193\}$

Vector
{0.345, -0.465, 0.856, 0.1543}
{-0.445, 0.565, 0.956, 0.2543}
{0.545, 0.665, 0.056, 0.3543}
{0.645, 0.765, 0.156, -0.4543}
{0.745, 0.865, 0.256, 0.5543}
{0.845, 0.965, -0.356, 0.6543}



Using a Vector Store

```
@Value("Artificial intelligence - Wikipedia.pdf")
private Resource pdf;
```

```
PagePdfDocumentReader reader = new PagePdfDocumentReader(pdf);
TokenTextSplitter splitter = new TokenTextSplitter();
List<Document> documents = splitter.apply(reader.get());
vectorStore.accept(documents);

String answer = chatClient.prompt()
    .system("""
        You are a virtual assistant and answers questions with the data provided.
        If you are not sure or don't know, honestly say you don't know.
        """)
    .user("Why is the definition of AI difficult?")
    .advisors(new QuestionAnswerAdvisor(vectorStore))
    .call()
    .content();

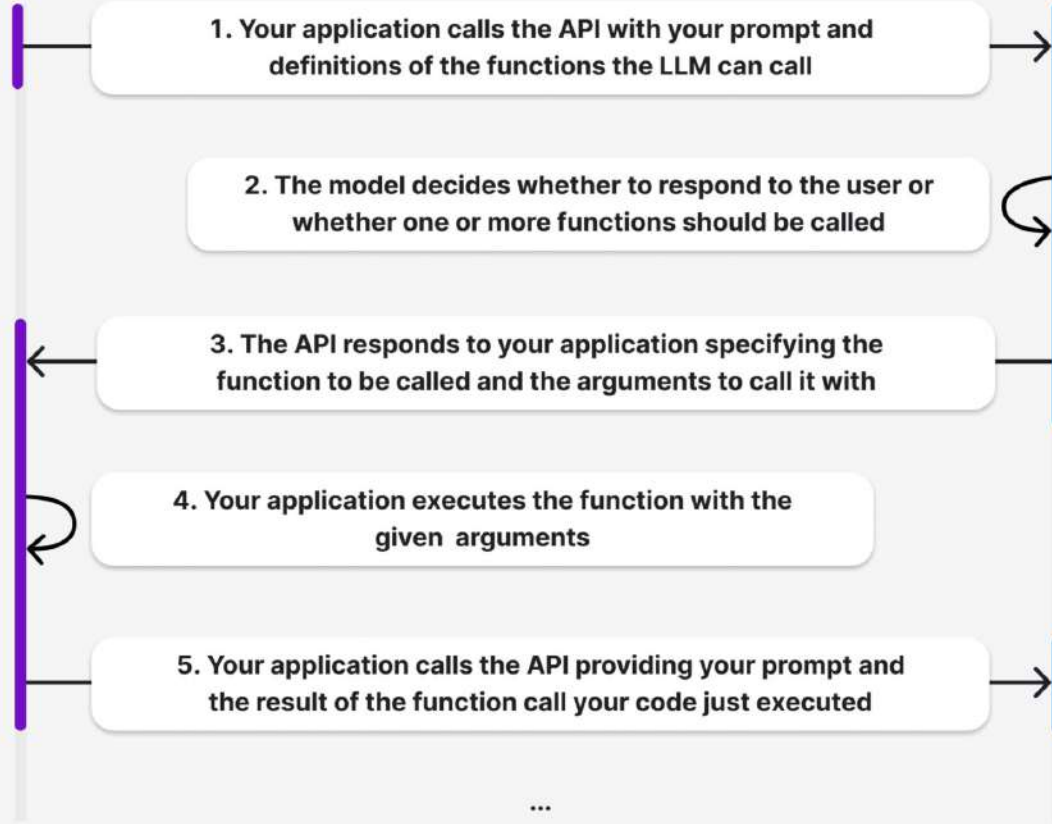
log.info("ChatGPT answered: {}", answer);
```

Registering Custom Functions

- The model generates a JSON object with arguments for the registered functions.
- The model does not call the function directly.
- Your code executes the function, and the result is returned to the model.
- ...

Your code

LLM



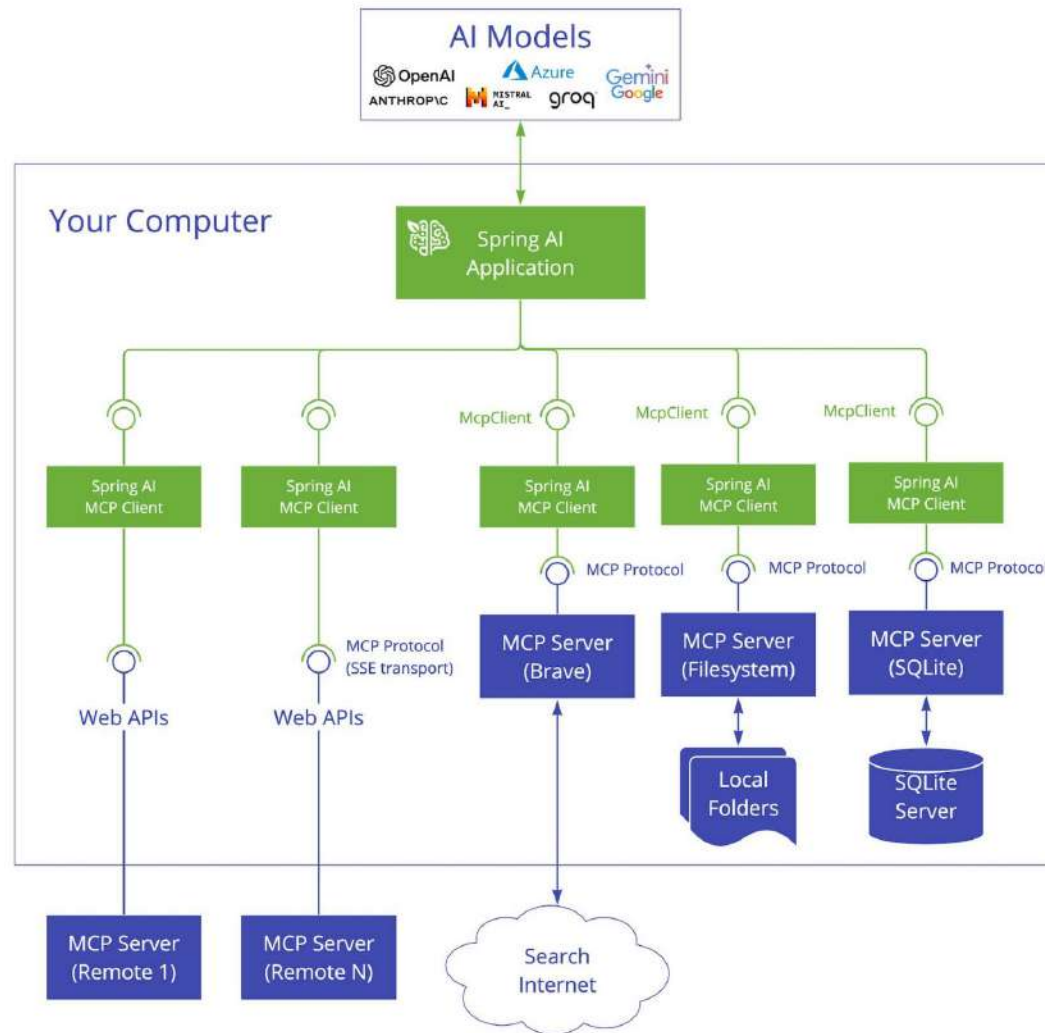
See [The lifecycle of a function call](#)

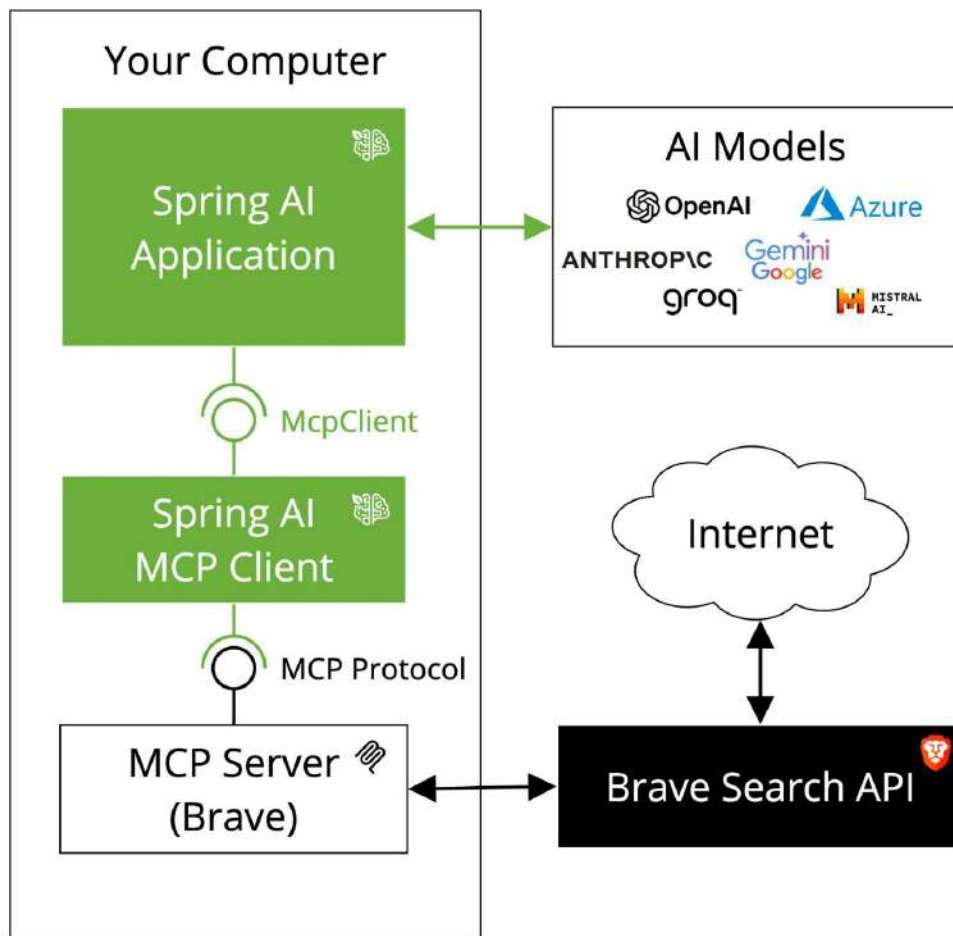
Function Calling

```
@Bean
// Request annotated with @JsonClassDescription and JsonPropertyDescription
public Function<WeatherService.Request, WeatherService.Response> weatherFunction() {
    return new WeatherService();
}
```

```
String answer = chatClient.prompt()
    .user("What's the weather like in Stockholm, Zurich, and New York?")
    .functions("weatherFunction")
    .call()
    .content();

log.info("ChatGPT answered: {}", answer);
```





Model Context Protocol (MCP)

```
// https://github.com/modelcontextprotocol/servers/tree/main/src/brave-search
var stdioParams = ServerParameters.builder("npx")
    .args("-y", "@modelcontextprotocol/server-brave-search")
    .addEnvVar("BRAVE_API_KEY", System.getenv("BRAVE_API_KEY"))
    .build();

var mcpClient = McpClient.sync(new StdioClientTransport(stdioParams)).build();
var init = mcpClient.initialize();
```

```
var chatClient = chatClientBuilder
    .defaultFunctions(mcpClient.listTools(null))
    .tools()
    .stream()
    .map(tool -> new McpFunctionCallback(mcpClient, tool))
    .toArray(McpFunctionCallback[]::new))
    .build();
```


Generative AI Challenges

Challenges	Patterns
Align responses to goals	System prompts
No structured output	Output converters
Not trained on your data	Prompt stuffing
Limited context size	RAG
Stateless APIs	Chat Memory
Not aware of your APIs	Function calling
Integration of standard APIs	Model Context Protocol (MCP)
Hallucinations	Evaluators