RAG Spring AI

Java can also handle LLMs

Przemek Skwiercz 11.2024

Przemek Skwiercz

- Programista i cyklista.
- Jeden z założycieli i organizatorów bydgoskiego JUG’a.
- Od ponad 20 lat zarabiam na kromkę chleba i kufelek piwa tworząc oprogramowanie więc jestem już chyba seniorem (wiekowo na pewno) :)
- W wolnym czasie miłośnik: wypraw rowerowych, kraftowego piwa, starego punk rocka i teatru (niekoniecznie w tej kolejności).

BYDGOSZCZ JUG

RAG Spring AI

Presentation

RAG - What, Why and How

How SPRING AI support creation of RAG App

RAG Spring AI - Demo app

RAG
What, Why & How

LLM

Main Problems

- “Static knowledge” of LLMs
- No real time updates (dynamic state)
- No access to highly specific data
- Security of customers data

LLM

Providing custom data to LLMs

- **Fine tune model** - rather for people with knowledge in AI areas, not for common users of LLMs
- **Function Calling** - permits the model to request the execution of client-side functions, thereby accessing necessary information or performing tasks dynamically as required
- **“Stuff the prompt”** - adds custom data to prompts -> **RAG**

RAG

Use cases

- Customer feedback analysis
- Speeding up onboarding new employees
- Boosting customer support
- Search engines
- Medical analysis

RAG

What is RAG?

RAG - Retrieval-Augmented Generation

Retrieval Phase

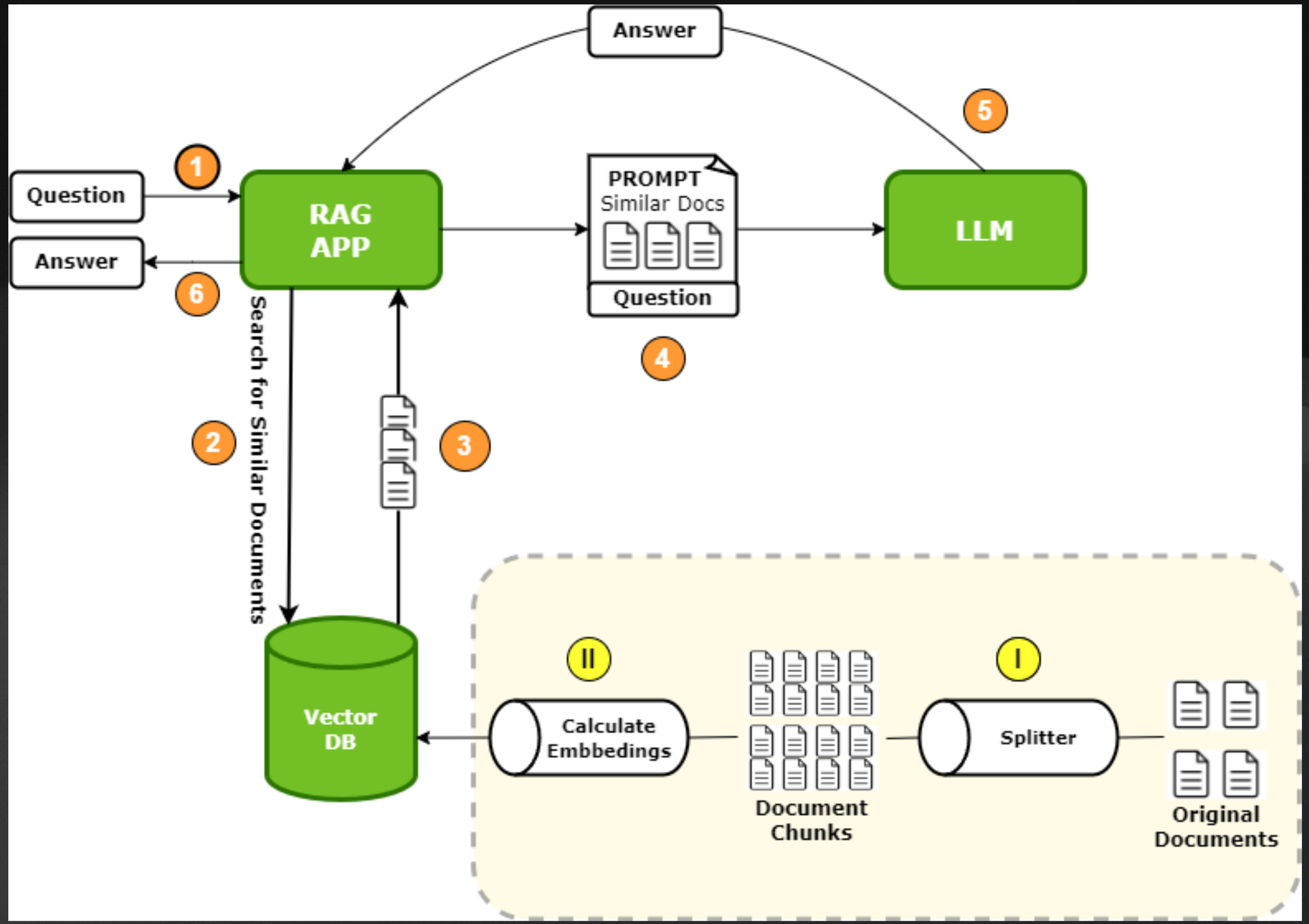
Search local data (stored in vector DB) to find relevant information to query

Generation Phase

LLM which works with custom data and generates answer

RAG

How it works?

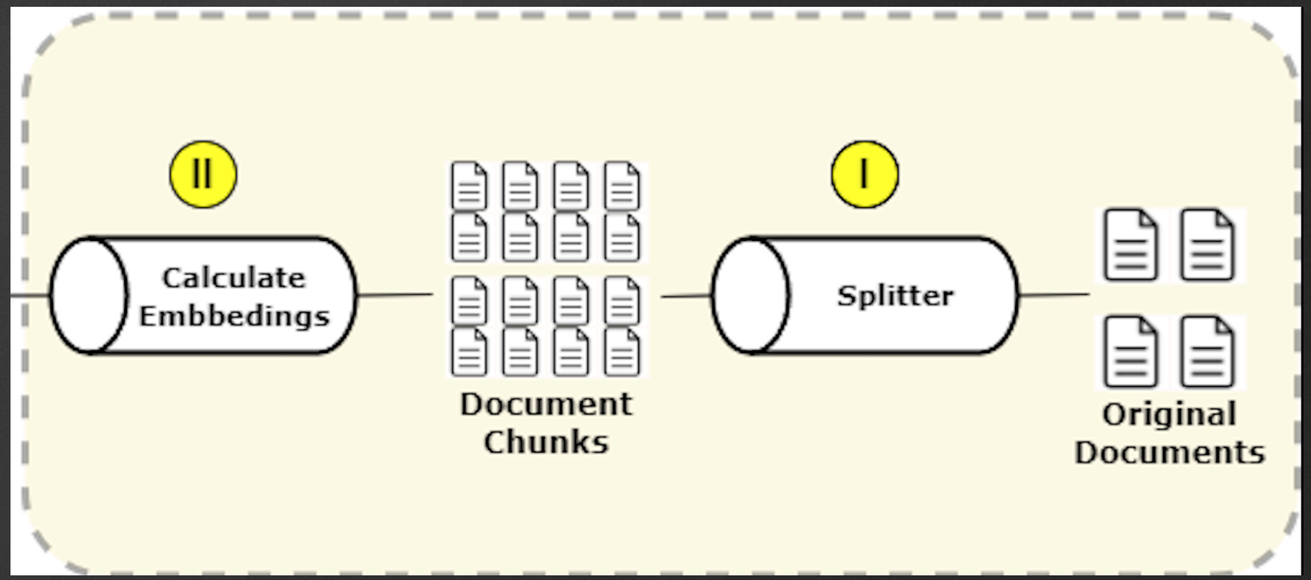


RAG

Embeddings

Embeddings - convert data into vector representation. It is performed in two steps:

1. **Tokenisation** - (algorithm **BPE** - **B**yte **P**air **E**ncoding)
2. **Transformation** - tokens to vectors (libraries provided by LLMs)



RAG
Embeddings

experiment with tokenisation:

GPT-4o & GPT-4o mini | GPT-3.5 & GPT-4 | GPT-3 (Legacy)

Many words map to one token, but some don't: indivisible.

Unicode characters like emojis may be split into many tokens containing the underlying bytes: 🖐

Sequences of characters commonly found next to each other may be grouped together: 1234567890

Clear | Show example

Tokens 53 **Characters** 252

Many words map to one token, but some don't: indivisible.

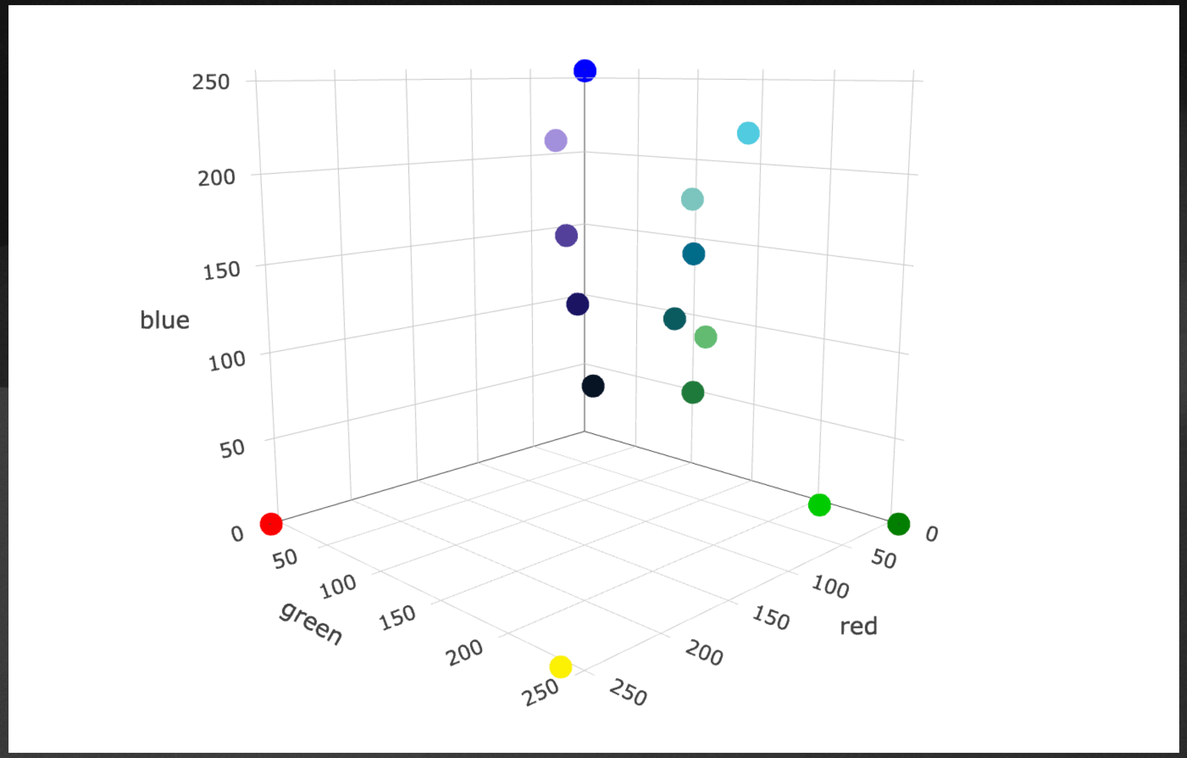
Unicode characters like emojis may be split into many tokens containing the underlying bytes: ����

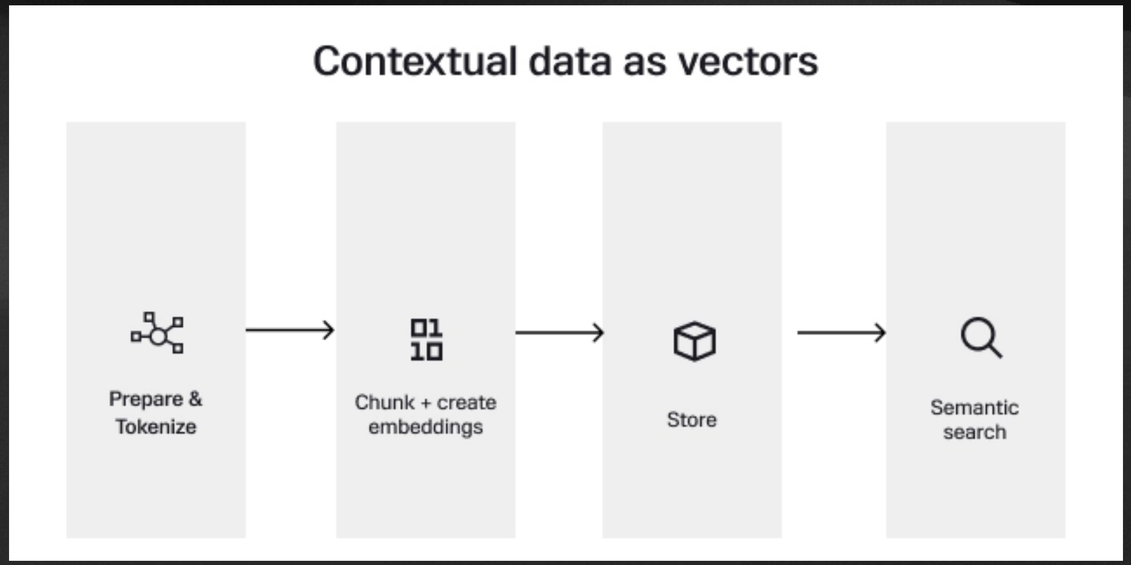
Sequences of characters commonly found next to each other may be grouped together: 1234567890

Text | Token IDs

RAG

Data as Vectors





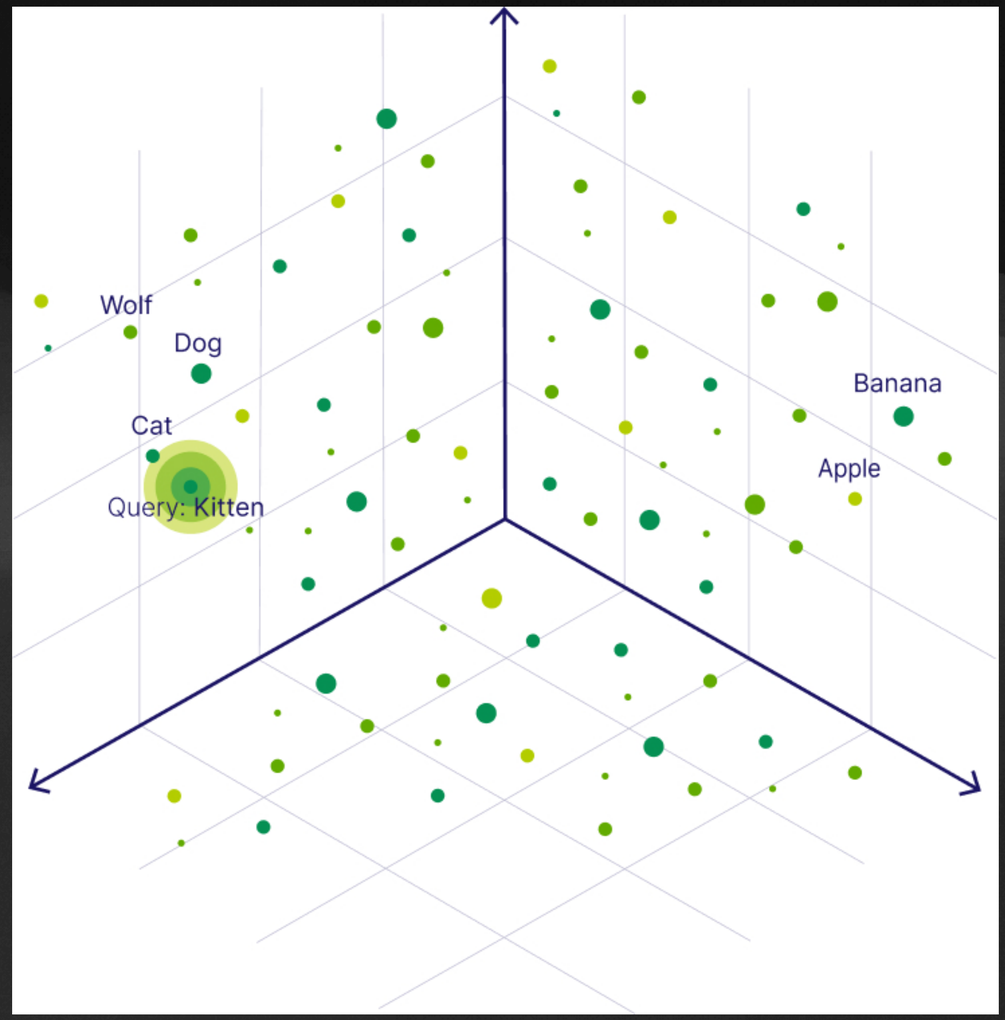
Source: https://weaviate.io/blog/what-is-a-vector-database

Source: https://www.singlestore.com/blog/a-complete-guide-to-vector-databases/

RAG

Similarity Search

Vector embeddings allow us to find and retrieve similar objects from the vector database by searching for objects that are close to each other in the vector space, which is called vector search, **similarity search**, or semantic search.



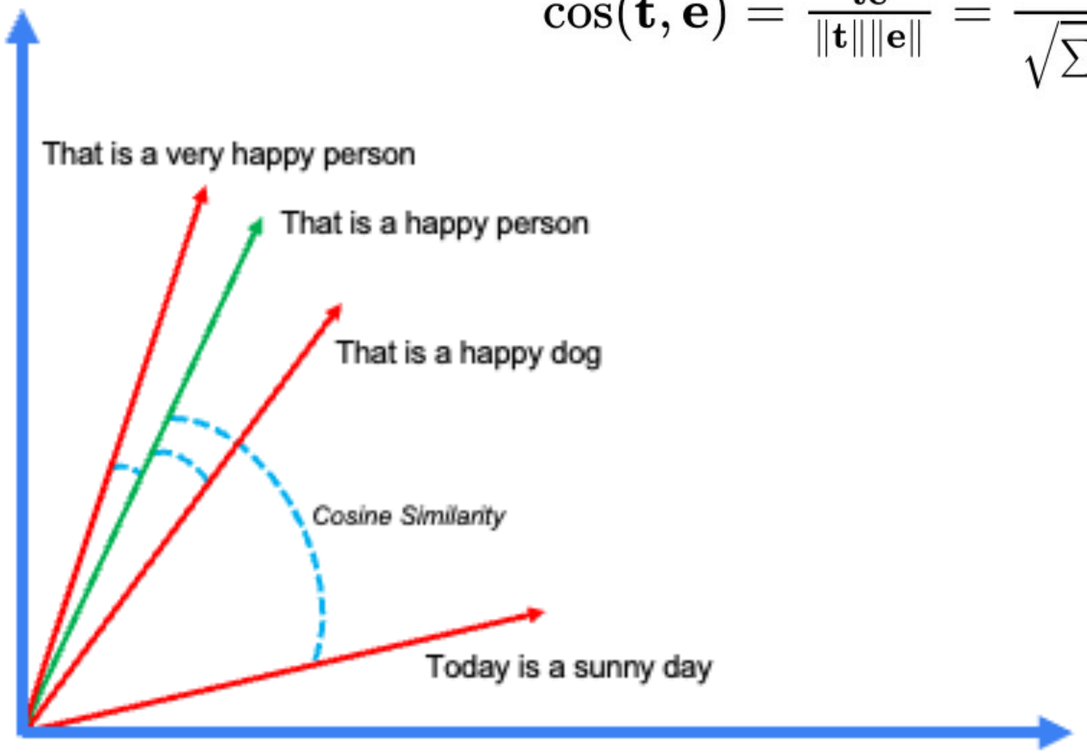
Paris - France + Poland = Warsaw

King - Man + Woman = Queen

Source: https://weaviate.io/blog/what-is-a-vector-database

RAG
Cosine Similarity

$$\cos(\mathbf{t}, \mathbf{e}) = \frac{\mathbf{te}}{\|\mathbf{t}\|\|\mathbf{e}\|} = \frac{\sum_{i=1}^{n} \mathbf{t}_i \mathbf{e}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{t}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{e}_i)^2}}$$



Source: https://mlops.community/vector-similarity-search-from-basics-to-production/

RAG

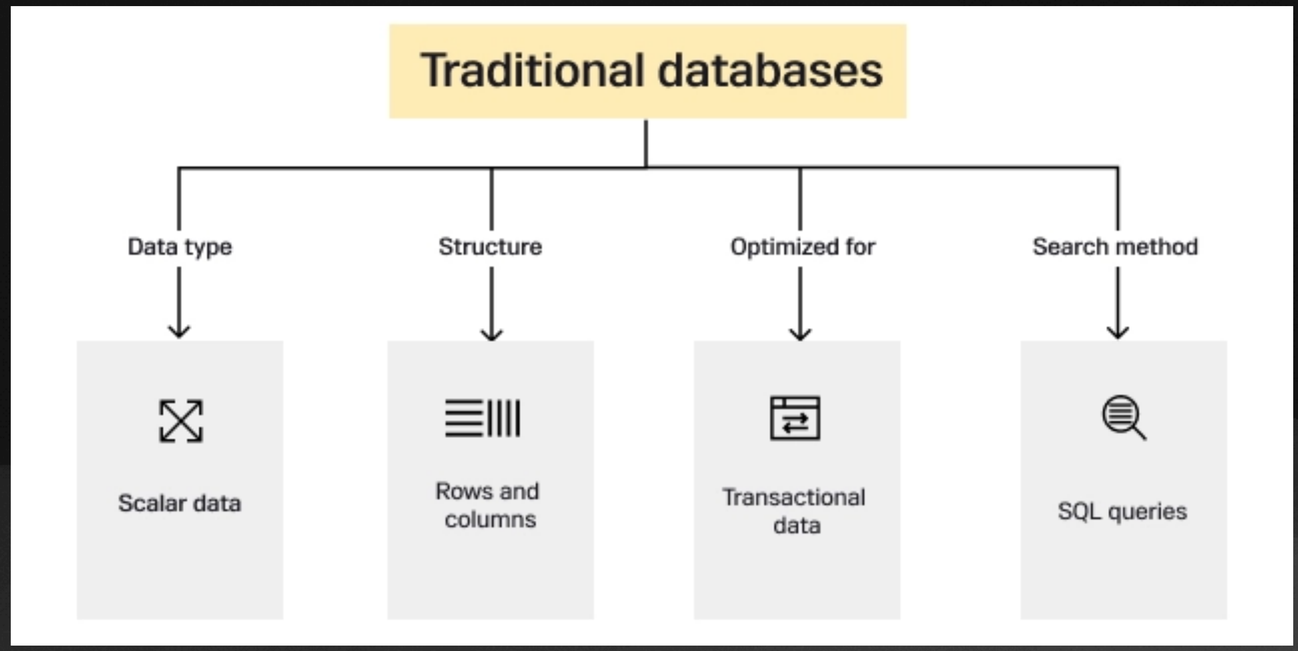
Cosine Similarity

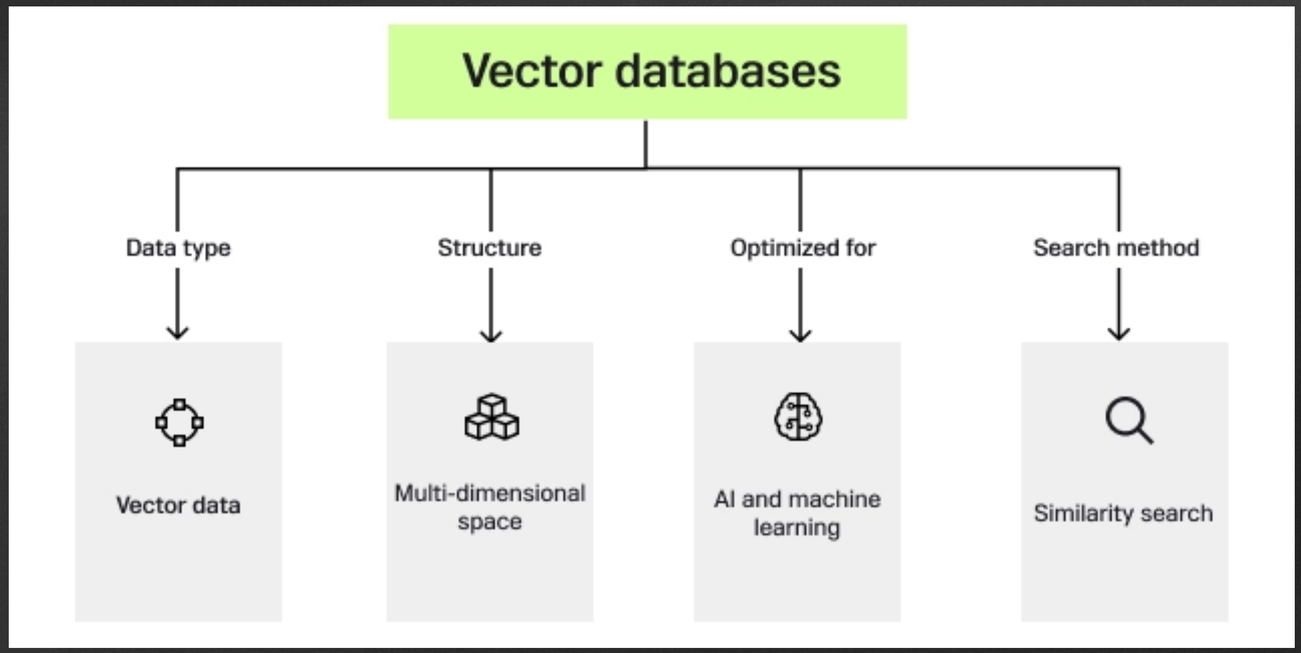
$$\cos(\mathbf{t}, \mathbf{e}) = \frac{\mathbf{te}}{\|\mathbf{t}\|\|\mathbf{e}\|} = \frac{\sum_{i=1}^{n} \mathbf{t}_i \mathbf{e}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{t}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{e}_i)^2}}$$

Source: https://mlops.community/vector-similarity-search-from-basics-to-production/

RAG

Traditional DB
vs
Vector DB





Source: https://www.singlestore.com/blog/a-complete-guide-to-vector-databases/

RAG

Prompts

Prompts - are the inputs that guide an AI model to generate specific outputs

Prompt Types:

SYSTEM

USER

ASSISTANT

TOOL

RAG

System prompt vs User prompt

System Prompt

- System prompt is a fixed prompt providing context and instructions to the model.
- The system prompt is always included in the provided input to the LLM, regardless of the user prompt.
- The idea is that the system prompt should not be editable by any party other than the developers.

User Prompt

- User prompts are the input queries from the user.
- This is how the customer interacts with LLM application, and it enables the user to shape this application’s input.

Spring AI

Spring AI

When was it made?

In **mid-2023**

Spring revealed that it was working on a module that makes it easier to create AI applications using the Spring framework

12.2023

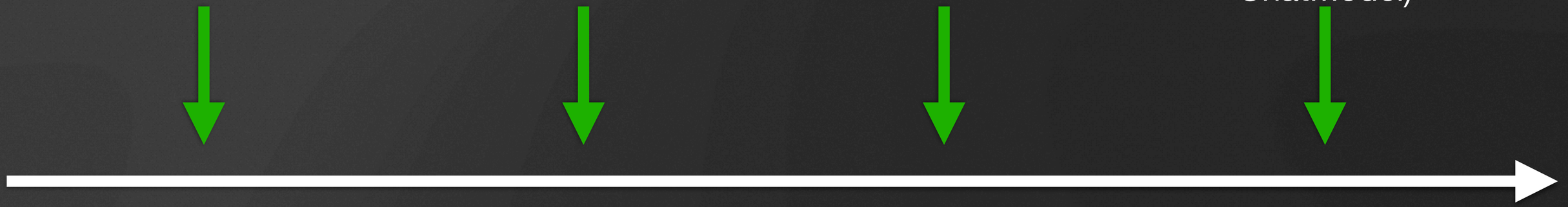
Spring AI left the experimental phase and was officially released

01.2024

Version 0.8.0 was released (most often used in courses, YT videos, etc.)

04.2024

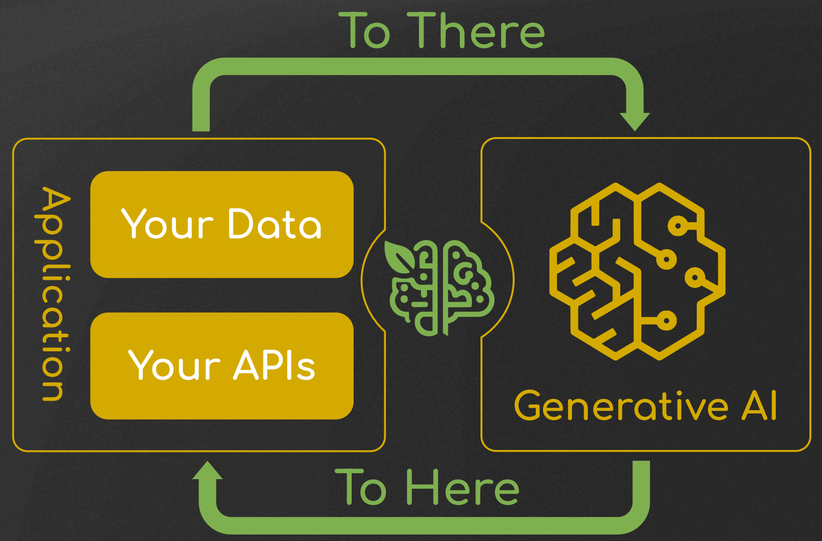
Version 1.0.0 (1.0.0RC) was released, introducing changes causing backward incompatibility (mainly ChatClient/ ChatModel)



Spring AI

What is it used for?

- **Spring AI is designed to integrate AI models with applications written in Spring**
- **It allows us to connect our data and API with AI models**
- **Spring AI is an abstraction on LLMs**



Source: https://spring.io/projects/spring-ai

Spring AI
Supported LLMs

- OpenAI Chat Completion (streaming, multi-modality & function-calling support)
- Microsoft Azure Open AI Chat Completion (streaming & function-calling support)
- Ollama Chat Completion (streaming, multi-modality & function-calling support)
- Hugging Face Chat Completion (no streaming support)
- Google Vertex AI PaLM2 Chat Completion (no streaming support)
- Google Vertex AI Gemini Chat Completion (streaming, multi-modality & function-calling support)
- Amazon Bedrock (Cohere, Llama, Titan, Anthropic, Jurassic2)
- Mistral AI Chat Completion (streaming & function-calling support)
- Anthropic Chat Completion (streaming & function-calling support)
- watsonx.AI

Spring AI

Supported Vector DB

- **SimpleVectorStore** - A simple implementation of persistent vector storage, good for educational purposes.
- **PgVector Store**
- **Azure Vector Search**
- **Apache Cassandra**
- **Elasticsearch Vector Store**
- **GemFire Vector Store**
- **Milvus Vector Store**
- **MongoDB Atlas Vector Store**
- **Neo4j Vector Store**
- **OpenSearch Vector Store**
- **Oracle Vector Store**
- **Pinecone Vector Store**
- **Qdrant Vector Store**
- **Redis Vector Store**
- **SAP Hana Vector Store**

Spring AI

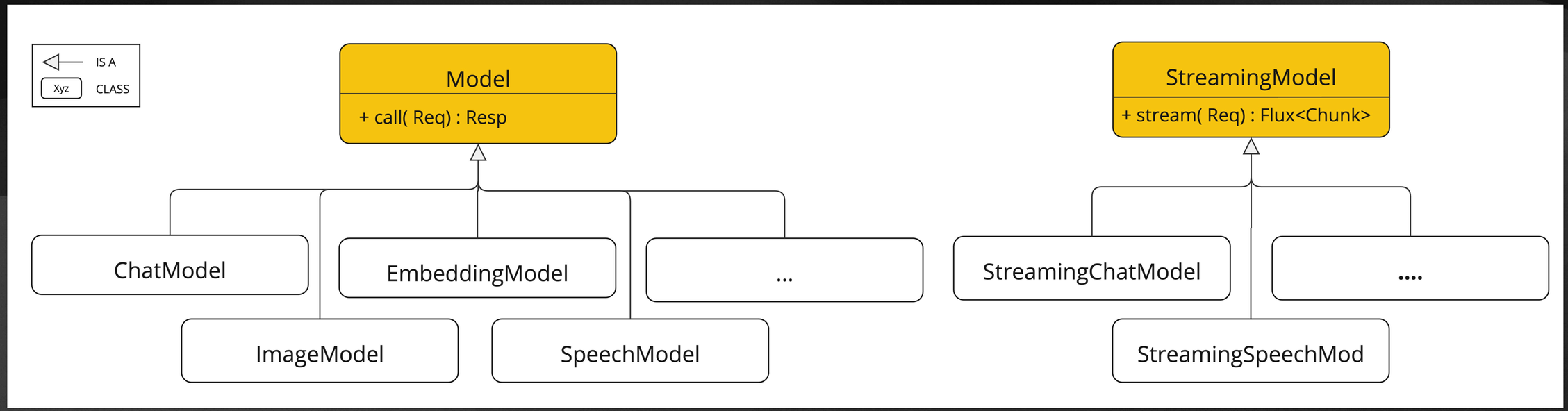
Alternatives for Spring AI

	JDK	GraalVM	Release	RAG support	Doc	Dev Activity
LangChain	8, 11, 17, 21	✅	General Availability	✅	👌	🚀🚀🚀
spring AI	17	❌	Experimental	✅	👍	🚀🚀
Semantic Kernel	17, 21	❌	Alpha	✅	👀	🚀

February 2024

Source: https://www.linkedin.com/pulse/leading-java-ai-frameworks-langchain4j-vs-spring-custom-miliari-gz7je

Spring AI
API



Source: https://spring.io/projects/spring-ai

Spring AI
Simple app

```
spring.application.name=spring-ai-ud
server.port=8080
spring.ai.openai.api-key=${OPEN_AI_KEY}
spring.ai.openai.chat.options.model=gpt-4o
```

```
@Service  no usages  1 inheritor  pskwiercz *
public class OpenAiServiceExample {

    private final ChatClient chatClient;  2 usages

    public OpenAiServiceExample(ChatClient.Builder chatClient) {  new *
        this.chatClient = chatClient.build();
    }

    Rename usages
    public String ask(String question) {  new *

        return chatClient
                .prompt() ChatClientRequest
                .user(question)
                .call() CallResponseSpec
                .content();

    }
}
```

```
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.3.0</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.pskwiercz</groupId>
    <artifactId>spring-ai-ud</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>war</packaging>
    <name>spring-ai-ud</name>
    <description>spring-ai-ud</description>
    <properties>
        <java.version>21</java.version>
        <spring-ai.version>1.0.0-M1</spring-ai.version>
    </properties>
    <dependencies> Add Starters...
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.ai</groupId>
            <artifactId>spring-ai-openai-spring-boot-starter</artifactId>
        </dependency>
    </dependencies>
```

SPRING AI

Vector DB Configuration

- Vector DB are used to integrate customer data with AI models.
- When a user query is to be sent to the AI model, a set of similar documents is first retrieved. These documents then serve as the context for the user's question and are sent to the AI model, along with the user's query.

```java
public interface VectorStore {

    void add(List<Document> documents);

    Optional<Boolean> delete(List<String> idList);

    List<Document> similaritySearch(String query);

    List<Document> similaritySearch(SearchRequest request);
}
```

Source: https://spring.io/projects/spring-ai

SPRING AI

Vector DB configuration

```yaml
ai:
  openai:
    api-key: ${OPENAI_API_KEY}
    embedding:
     options:
       model: text-embedding-ada-002
    chat:
      options:
        model: gpt-4-turbo

  vectorstore:
    milvus:
      initialize-schema: true
      client:
        host: "localhost"
        port: 19530
        username: "root"
        password: "milvus"
      databaseName: "default"
      collectionName: "vector_store"
      embeddingDimension: 1536
      indexType: IVF_FLAT
      metricType: COSINE
```

SPRING AI

Prompts

```java
public class Prompt implements ModelRequest<List<Message>> {

    private final List<Message> messages;

    private ChatOptions chatOptions;
}
```

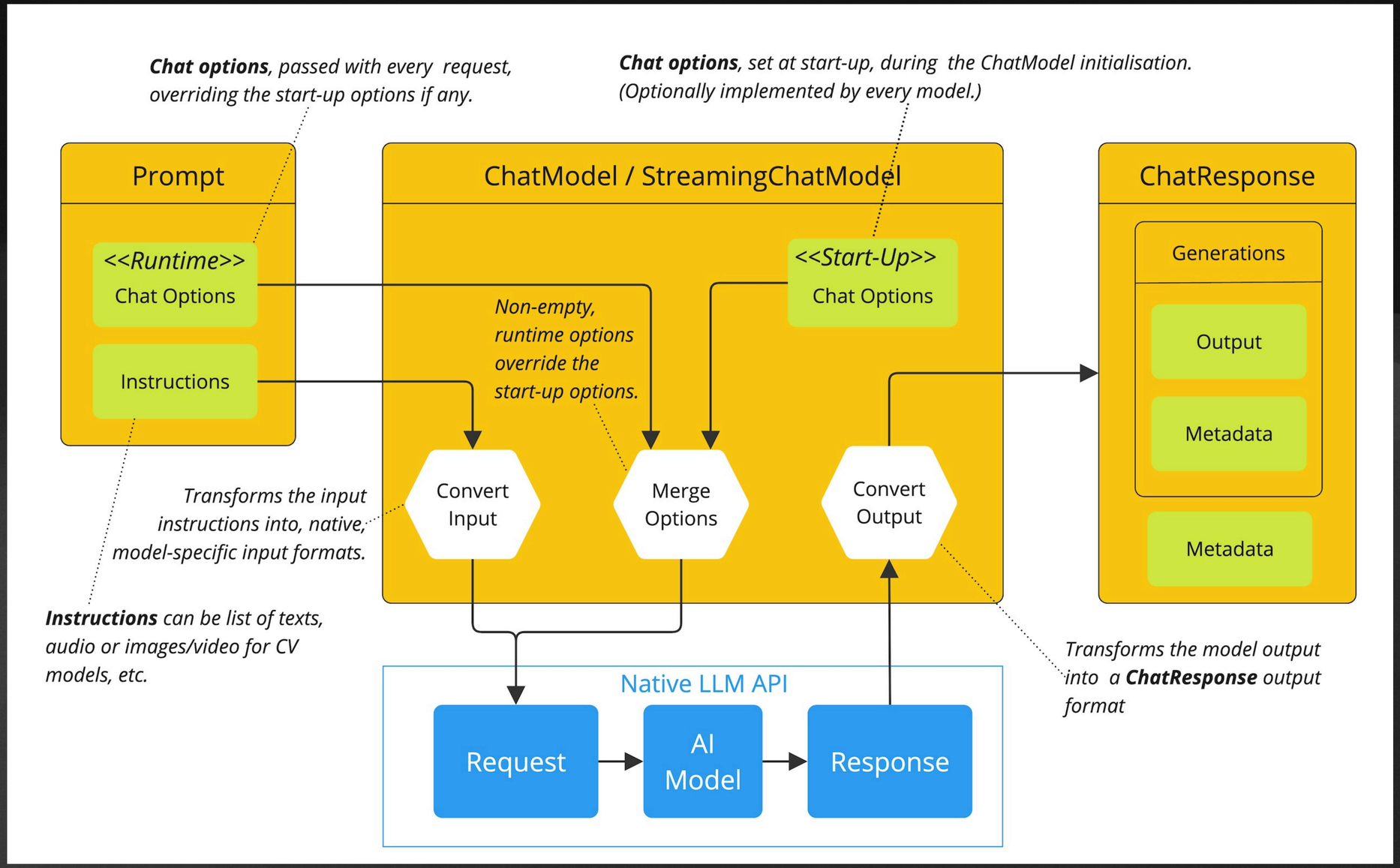
```java
public interface ChatOptions extends ModelOptions {

    String getModel();
    Float getFrequencyPenalty();
    Integer getMaxTokens();
    Float getPresencePenalty();
    List<String> getStopSequences();
    Float getTemperature();
    Integer getTopK();
    Float getTopP();
    ChatOptions copy();

}
```

Source: https://spring.io/projects/spring-ai

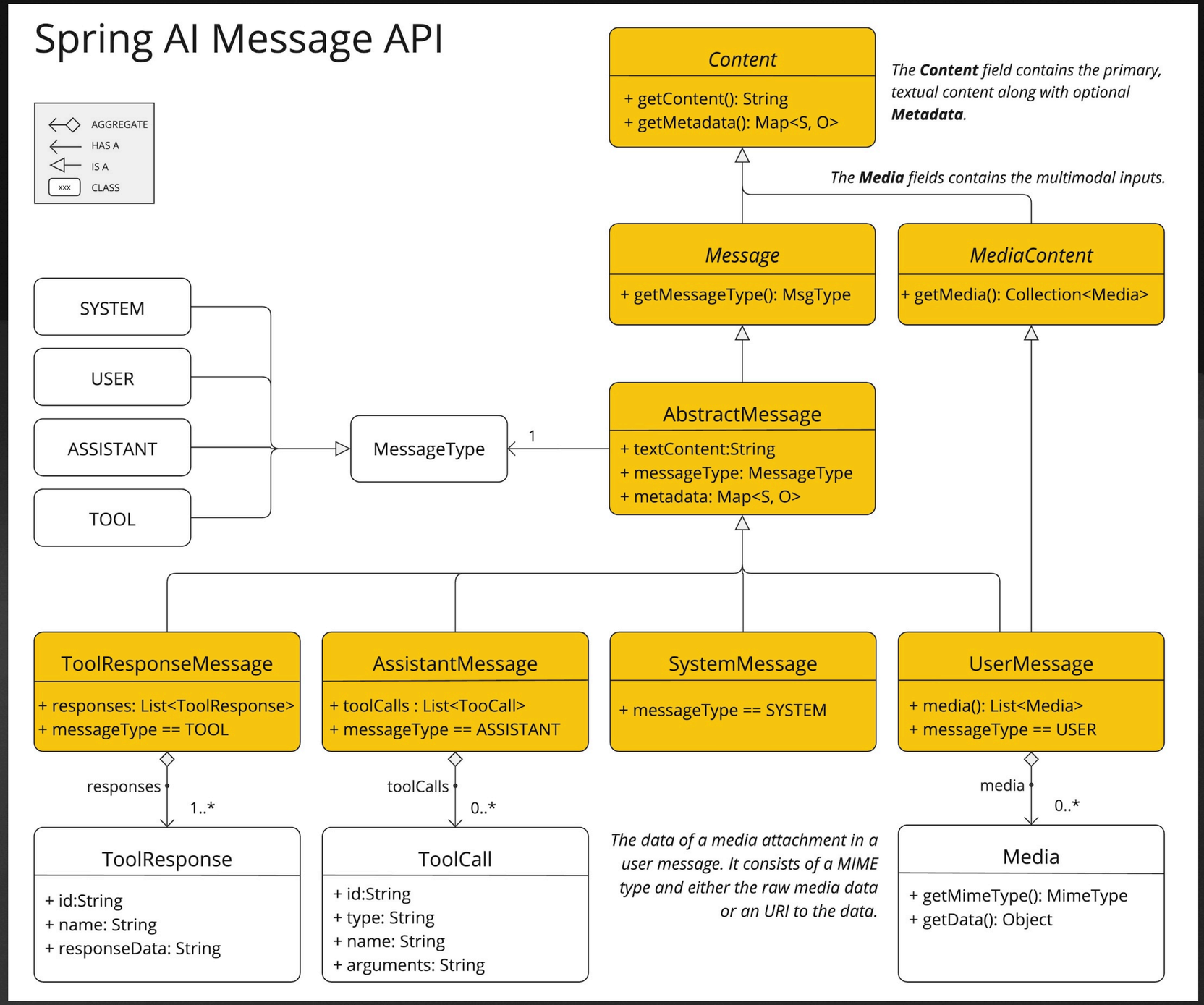
Spring AI
Chat Model



Source: https://spring.io/projects/spring-ai

SPRING AI

Prompts



Source: https://spring.io/projects/spring-ai

SPRING AI

String Templates

river-prompt.st

```
What is the longest river in {country}? Response in JSON with the name of the river in a property named
Do not wrap the json codes in JSON
```

OpenAiService.java

```java

@Service  2 usages  pskwiercz *
public class OpenAiService {

    private final ChatClient chatClient;  2 usages

    @Value("classpath:templates/river-prompt.st")
    private Resource riverPrompt;

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

    public Answer getRiver(GetRiverRequest getRiverRequest) {  1 usage  new *

        Prompt prompt = new PromptTemplate(riverPrompt)
                .create(Map.of( k1: "country", getRiverRequest.country()));

        String answer = chatClient
                .prompt(prompt) ChatClientPromptRequest
                .call() CallPromptResponseSpec
                .content();

        return new Answer(answer);
    }
}
```

Demo

https://github.com/pskwiercz/jug-rag-spring-ai

Spring AI

What Next

Advisors

Function Calling

Observability

Embedding Models

Q&A