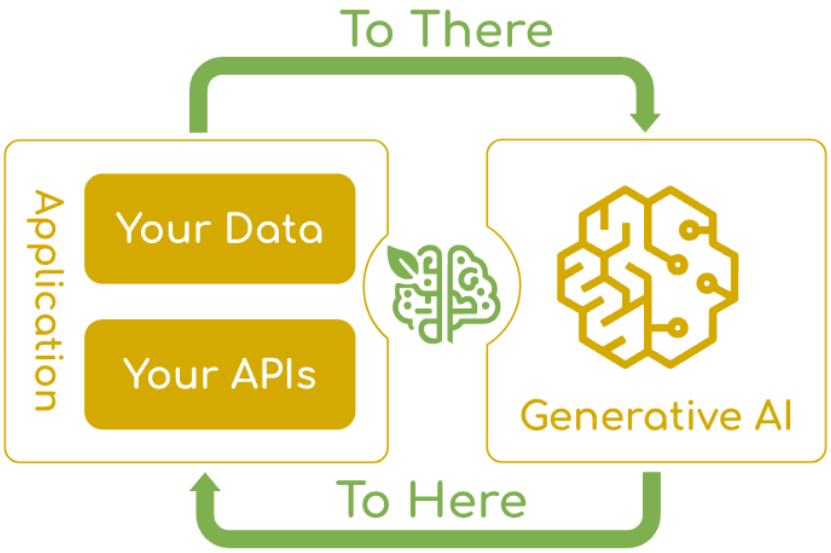
Spring AI

NOTE

Spring AI addresses the fundamental challenge of AI integration: `Connecting your enterprise Data and APIs with AI Models`.



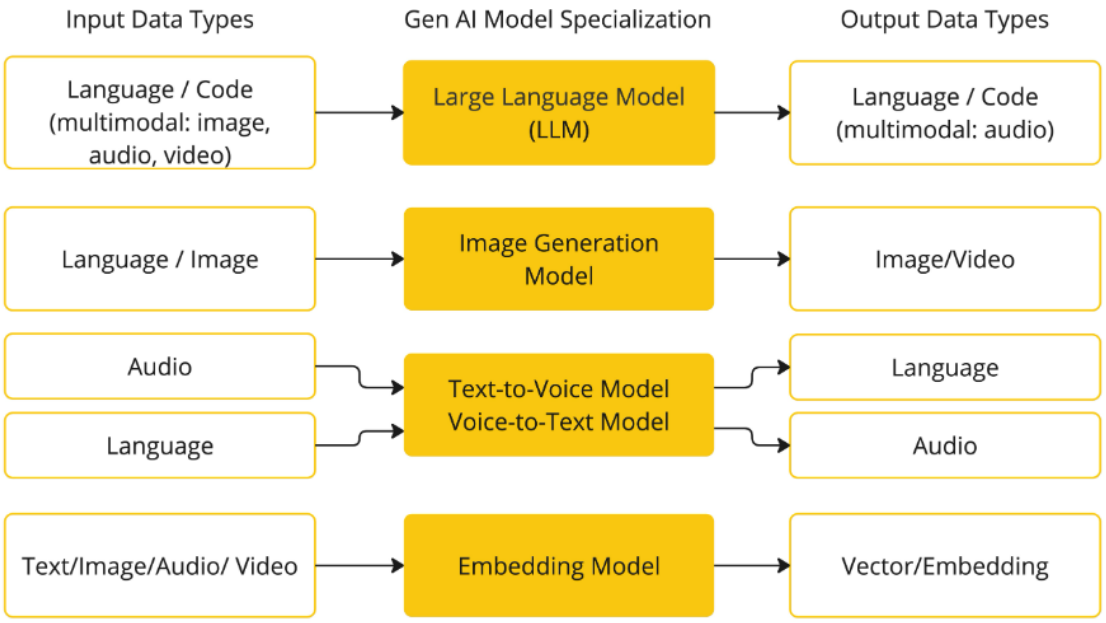
- Spring AI is a **Spring framework extension** that makes it simple for Java developers to add **AI features** (like chatbots, text generation, and document Q&A) into Spring Boot applications.
- It provides a **unified, Spring-friendly way** to connect to different AI models (OpenAI, Google, Hugging Face, etc.) and handles the **complex parts** of working with them, so you can focus on building features instead of learning new AI tools or managing complicated workflows.
- **Think of it as:** "Your Spring Boot app gets a smart AI brain that you can easily swap or upgrade without rewriting your code."
- "Spring AI is an extension of the Spring ecosystem that allows developers to easily integrate AI and machine learning capabilities—like chatbots, text generation, and document Q&A—into Spring Boot applications. It provides a unified API to work with different AI providers such as OpenAI, Hugging Face, and Azure OpenAI, so we can switch providers without major code changes. It also handles the

complexity of prompt management, API calls, and response handling, allowing us to focus on building application features rather than dealing with low-level AI integration details."

Features of Spring AI

1. **Unified & Portable API** – One consistent API for multiple AI tasks (chat, text-to-image, embeddings, audio) across different providers.
2. **Multi-Provider Support** – Works with OpenAI, Anthropic, Microsoft, Amazon, Google, Ollama, etc., with easy provider switching.
3. **Vector Database Integration** – Connects to major vector stores (Pinecone, Redis, Milvus, MongoDB Atlas, etc.) with a portable query API.
4. **Advanced AI Capabilities** – Supports structured outputs, function calling, conversation memory, and Retrieval Augmented Generation (RAG).
5. **Spring Boot Friendly** – Auto-configuration, starters, observability tools, and built-in evaluation utilities.

AI Concepts



1. Models

- **Definition:** Algorithms that process/generate information by learning patterns from data.
- **Purpose:** Generate text, images, audio, predictions, etc.
- **Types Supported by Spring AI:**
  - **Language models:** Chat, text generation
  - **Image models:** Text-to-image generation
  - **Audio models:** Speech recognition & synthesis
  - **Embeddings:** Numeric representations of text/images for similarity and semantic search
- **Key point:** Pre-trained models (e.g., GPT) allow developers to use AI without training from scratch.

---

2. Prompts

- **Definition:** Inputs that guide AI to produce desired outputs.
- **Details:**

    - Composed of multiple roles (system, user, assistant).
    - Crafting prompts effectively is called **Prompt Engineering**.
    - Well-designed prompts drastically improve AI responses.
- **Example:** Instead of just asking “Write a joke,” use structured text: system role + user role instructions.

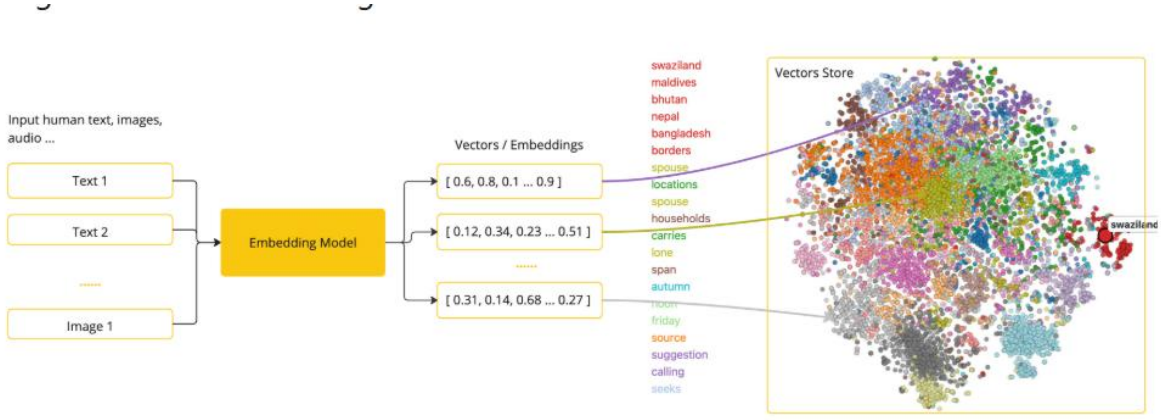
---

3. Prompt Templates

- **Definition:** Predefined text structures with placeholders for dynamic input.
- **Purpose:** Simplifies creating consistent prompts.
- **Implementation:** Spring AI uses **StringTemplate** for placeholders.
- **Analogy:** Like the “View” in Spring MVC — populates placeholders from data maps.
- **Example:**
- Tell me a {adjective} joke about {topic}.

---

4. Embeddings



- **Definition:** Numeric vector representations of text, images, or videos.

- **Purpose:** Capture relationships between inputs to measure similarity.
- **Key Use:**
    - Semantic search
    - Text classification
    - Retrieval Augmented Generation (RAG)
- **Analogy:** Think of a multi-dimensional space where similar concepts are close together.

---

5. Tokens

- **Definition:** Basic units AI models process (words → tokens → AI).
- **Details:**
    - ~75% of a word = 1 token
    - Important for cost calculation and context limits.
    - Models have **token limits** (context window).
- **Example:** ChatGPT-3: 4K tokens; GPT-4: 8K–32K tokens.

---

6. Structured Output

- **Definition:** Converting AI text outputs into usable data structures (like JSON or POJOs).
- **Purpose:** Makes AI output easy to use in applications.
- **Challenge:** Raw AI output is a string; structured output ensures correct formatting for processing.

---

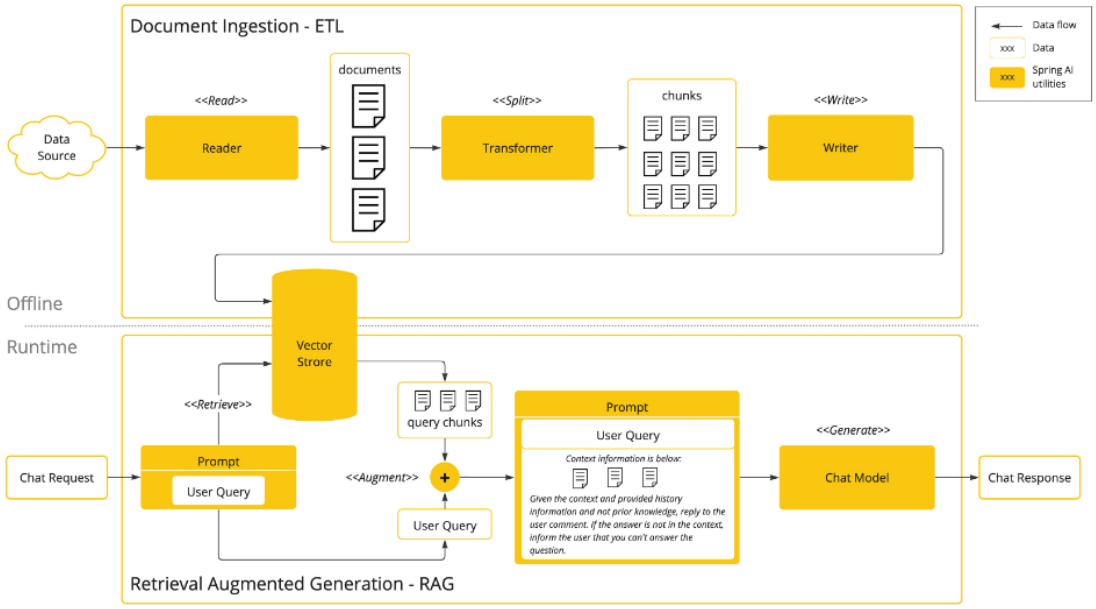
7. Bringing Your Data & APIs to AI

- **Problem:** Pre-trained models only know data up to a cutoff date.

- **Solutions:**
    1. **Fine-tuning:** Retrain model (resource-heavy).
    2. **Prompt Stuffing / RAG:** Embed relevant data in prompts.
    3. **Tool Calling:** Connect AI to external APIs for real-time data.

---

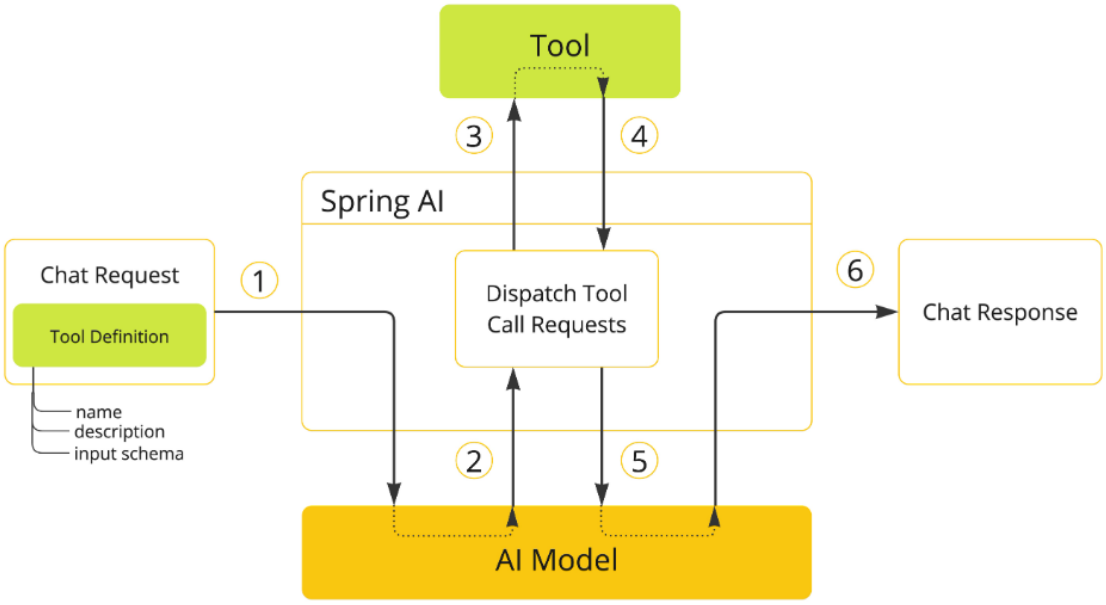
8. Retrieval Augmented Generation (RAG)



- **Definition:** Technique to fetch relevant data from a vector database to improve AI accuracy.
- **Steps:**
    1. ETL: Extract, transform, load unstructured data into vector DB.
    2. Split data carefully to preserve semantic meaning and token limits.
    3. At query time, retrieve similar content and send in AI prompt.

---

9. Tool Calling



- **Definition:** Allows AI models to call external tools/APIs for real-time info or actions.
- **How it works:**
    1. Register tools with name, description, input schema.
    2. Model decides to call a tool.
    3. Application executes tool and returns result.
    4. AI generates response using tool output as context.
- **Benefit:** AI can perform dynamic actions and access live data.

---

10. Evaluating AI Responses

- **Definition:** Ensuring AI outputs are accurate, relevant, and coherent.
- **Techniques:**
    - Compare response vs. user intent
    - Use vector database context for evaluation
    - Spring AI provides an **Evaluator API** to automate response checks

Models

1. Chat Models

- **Definition:** AI models designed to have human-like conversations. They can understand questions, provide answers, and follow instructions.
- **Purpose:** Customer support, virtual assistants, coding help, tutoring, and general Q&A.
- **How it works:**
    1. You give a **prompt** (input text).
    2. The model generates a **response** (output text) based on the prompt and its pre-trained knowledge.
- **Example:**
    - **Prompt:** "Explain Newton's first law in simple words."
    - **Output:** "An object stays still or keeps moving straight unless something pushes or pulls it."
- **Popular Models:** ChatGPT, Claude AI, Bard, etc.
- **Tip:** Chat models can use **memory** or context to continue conversations naturally.

---

2. Embeddings Models

- **Definition:** Convert text, images, or other data into numeric vectors that capture meaning.
- **Purpose:** Measure similarity, group related items, search semantically instead of keyword-based search.
- **How it works:**
    - Text → Embedding vector (array of numbers)

    - Compare vectors to find similarity (smaller distance = more similar)
- **Example:**
    - Input texts:
        1. “I love programming in Java”
        2. “Java coding is fun”
    - Embedding vectors of both texts will be close → model knows they are similar.
- **Applications:** Semantic search, recommendations, RAG (Retrieval-Augmented Generation).

---

3. Image Generation Models

- **Definition:** AI models that generate images from text descriptions.
- **Purpose:** Create visual content, concept art, marketing material, game design, or AI art.
- **How it works:**
    1. Input a **prompt** describing the image.
    2. Model generates a **new image** that matches the description.
- **Example:**
    - Prompt: “A cat wearing sunglasses sitting on a beach.”
    - Output: AI generates an image showing exactly that scene.
- **Popular Models:** MidJourney, Stable Diffusion, DALL-E, etc.

---

4. Transcription Models

- **Definition:** Convert spoken audio into text.

- **Purpose:** Voice-to-text conversion, meeting notes, captioning, or accessibility features.
- **How it works:**
    1. Upload audio file (speech).
    2. Model processes audio → outputs **text transcription**.
- **Example:**
    - Audio: Someone says “Hello, how are you today?”
    - Output: “Hello, how are you today?”
- **Popular Models:** Whisper, Azure Speech-to-Text, Amazon Transcribe.

---

5. Text-To-Speech (TTS) Models

- **Definition:** Convert written text into spoken audio.
- **Purpose:** Assistive technology, audiobook creation, voice assistants, accessibility tools.
- **How it works:**
    1. Input a text prompt.
    2. Model generates an audio file that reads the text aloud.
- **Example:**
    - Input: “Welcome to our website!”
    - Output: Audio saying “Welcome to our website!” in a natural-sounding voice.
- **Popular Models:** Amazon Polly, Google Text-to-Speech, Microsoft Azure TTS.

---

6. Vector Databases

- **Definition:** Specialized databases that store **embedding vectors** for fast similarity search.
- **Purpose:** Make AI apps faster and smarter by finding related items quickly.
- **How it works:**
    1. Store embeddings of text, images, or other data.
    2. Query with a new embedding → returns the **most similar stored vectors**.
- **Example:**
    - Store embeddings of 10,000 product descriptions.
    - Query embedding: “wireless headphones with noise cancellation”
    - Database finds products with the closest matching description.
- **Popular Vector Databases:** Pinecone, Milvus, Weaviate, Qdrant, PostgreSQL+PGVector.