Getting Started

Getting Started

This section offers jumping off points for how to get started using Spring AI.

You should follow the steps in each of the following section according to your needs.

Spring CLI

The Spring CLI, simplifies creating new applications directly from your terminal. Like the 'create-react-app' command for those familiar with the JavaScript ecosystem, Spring CLI provides a `spring boot new` command to create Spring-based projects. Spring CLI also offers features to integrate external code bases into your current project, and many other productivity features.

Note

Note

It is important to understand that the "Spring CLI" is a distinct project from the "Spring Boot CLI", each with its own set of functionalities.

To begin creating a Spring AI application, follow these steps:

1. Download the latest Spring CLI Release and follow the installation instructions.
2. To create a simple OpenAI-based application, use the command:

   ```shell
   spring boot new --from ai --name myai
   ```

3. Consult the generated `README.md` file for guidance on obtaining an OpenAI API Key and running your first AI application.

To add the same simple AI application to an existing project, execute:

```shell
spring boot add ai
```

Note

Spring CLI allows users to define their own project catalogs that define which projects you can create or add to your existing code base.

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.

Add Milestone and Snapshot Repositories

If you prefer to add the dependency snippets by hand, follow the directions in the following sections.

To use the Milestone and Snapshot version, you need to add references to the Spring Milestone and/or Snapshot repositories in your build file.

For Maven, add the following repository definitions as needed:

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

For Gradle, add the following repository definitions as needed:

GROOVY
```groovy
repositories {
  mavenCentral()
  maven { url 'https://repo.spring.io/milestone' }
  maven { url 'https://repo.spring.io/snapshot' }
}
```

Dependency Management

The Spring AI Bill of Materials (BOM) declares the recommended versions of all the dependencies used by a given release of Spring AI. Using the BOM from your application's build script avoids the need for you to specify and maintain the dependency versions yourself. Instead, the version of the BOM you're using determines the utilized dependency versions. It also ensures that you're using supported and tested versions of the dependencies by default, unless you choose to override them.

If you're a Maven user, you can use the BOM by adding the following to your pom.xml file -

```xml
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>org.springframework.ai</groupId>
            <artifactId>spring-ai-bom</artifactId>
            <version>0.8.1-SNAPSHOT</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
```

Gradle users can also use the Spring AI BOM by leveraging Gradle (5.0+) native support for declaring dependency constraints using a Maven BOM. This is implemented by adding a 'platform' dependency handler method to the dependencies section of your Gradle build script. As shown in the snippet below this can then be followed by version-less declarations of the Starter Dependencies for the one or more spring-ai modules you wish to use, e.g. spring-ai-openai.

```gradle
dependencies {
  implementation platform("org.springframework.ai:spring-ai-bom:0.8.1-
SNAPSHOT")
  // Replace the following with the starter dependencies of specific modules
you wish to use
  implementation 'org.springframework.ai:spring-ai-openai'
}
```

Add dependencies for specific components

Each of the following sections in the documentation shows which dependencies you need to add to your project build system.

Embeddings Models

- Embeddings API
  - Spring AI OpenAI Embeddings
  - Spring AI Azure OpenAI Embeddings
  - Spring AI Ollama Embeddings
  - Spring AI Transformers (ONNX) Embeddings
  - Spring AI PostgresML Embeddings
  - Spring AI Bedrock Cohere Embeddings
  - Spring AI Bedrock Titan Embeddings
  - Spring AI VertexAI Embeddings
  - Spring AI MistralAI Embeddings

Chat Models

- Chat Completion API
  - OpenAI Chat Completion (streaming and function-calling support)
  - Microsoft Azure Open AI Chat Completion (streaming and function-calling support)
  - Ollama Chat Completion
  - HuggingFace 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 Chat Completion
    - Llama2 Chat Completion
    - Titan Chat Completion
    - Anthropic Chat Completion

    - MistralAI Chat Completion (streaming and function-calling support)

Image Generation Models

- Image Generation API
    - OpenAI Image Generation
    - StabilityAI Image Generation

Vector Databases

- Vector Database API
    - Azure Vector Search - The Azure vector store.
    - ChromaVectorStore - The Chroma vector store.
    - MilvusVectorStore - The Milvus vector store.
    - Neo4jVectorStore - The Neo4j vector store.
    - PgVectorStore - The PostgreSQL/PGVector vector store.
    - PineconeVectorStore - PineCone vector store.
    - QdrantVectorStore - Qdrant vector store.
    - RedisVectorStore - The Redis vector store.
    - WeaviateVectorStore - The Weaviate vector store.
    - SimpleVectorStore - A simple (in-memory) implementation of persistent vector storage, good for educational purposes.

Sample Projects

You can clone these projects on GitHub to get started.

OpenAI

- github.com/rd-1-2022/ai-openai-helloworld

Azure OpenAI

- github.com/rd-1-2022/ai-azure-openai-helloworld
- github.com/Azure-Samples/spring-ai-azure-workshop