Lecture 13: Spring AI with Ollama

Why Use Ollama in Spring AI?

- Earlier, we connected Spring AI with **OpenAI**.
- Now, we want to use **Ollama** (local models) inside our Spring AI application.
- Benefit: no API key, no cost, runs directly on your machine.

Steps to Integrate Ollama

- **Create a new Controller**
  - Create the OllamaController class.
  - This controller will use OllamaChatModel.
- **Add Dependency**
  - In pom.xml, add the Ollama dependency from start.spring.io or the official docs.
  - Now you'll have both: OpenAI and Ollama.

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

<dependency>
    <groupId>org.springframework.ai</groupId>
    <artifactId>spring-ai-starter-model-ollama</artifactId>
</dependency>
```

- **Reload Maven**
  - After updating pom, reload Maven to apply changes.

Implementation:

```java
@RestController
public class OllamaController {
    private ChatClient chatClient;

    public OllamaController(OllamaChatModel chatModel) {
        this.chatClient = ChatClient.create(chatModel);
    }

    @GetMapping("/api/{message}")
    public ResponseEntity<String> getAnswer(@PathVariable String message) {
        ChatResponse chatResponse = chatClient.prompt(message)
                .call()
                .chatResponse();

        System.out.println(chatResponse.getMetadata().getModel());

        String response = chatResponse
                .getResult()
                .getOutput()
                .getText();
        return ResponseEntity.ok(response);
    }
}
```

Key Points

- **Default Model**: If no model is specified, Spring AI with Ollama uses **Mistral**.
- **Missing Models**: If Mistral is not installed, you'll get errors.
- **Custom Model:** Set model via property:
    - spring.ai.ollama.chat.options.model=deepseek-7b
- **No Keys Needed:**Local Ollama models don't require API keys.

How It Works

- Run 'ollama list' → check installed models.
- Default model is used unless overridden in properties.
- To switch, install your desired model (e.g., DeepSeek) and update the Spring property.

Takeaways

- Ollama makes Spring AI work with **local open-source LLMs**.
- Add dependency → configure controller → choose model in properties.
- Easy to test: simply **hit /api/{message}** and see responses.
- More control, free usage, but it depends on your machine's hardware.