Lecture 02: Why Spring AI

Direct Use of LLMs:

- You can directly use OpenAI or Anthropic SDKs in your Java application.
- **Example with OpenAI:**
  - Create an account, and add credits.
  - Set **OPENAI_API_KEY** as an environment variable.
  - Add the **openai-java dependency** to Maven.
  - Write simple code to send a prompt (e.g., "Tell me a joke") → get a response.
- This works fine if you only plan to use one model.

The Problem:

- Different providers have different SDKs and code styles.
  - OpenAI code looks one way.
  - Anthropic code looks another way.
- If you want to:
  - Switch from OpenAI → Anthropic → Gemini in the future.
  - Use multiple models in the same project, and you'll need to rewrite large parts of your code.
- Writing your own abstraction layer is possible but time-consuming and repetitive.

Why Spring AI:

- Spring AI provides abstraction out-of-the-box.
- Instead of writing model-specific code, you use a common API.
- **Benefits:**
  - Cleaner and more maintainable code.
  - Easy to switch providers in the future.
  - **Supports multiple providers:** OpenAI, Anthropic, Microsoft, Amazon, Google, and Ollama.
- Widely adopted because companies prefer abstraction to keep projects flexible and easier to manage.

Key Takeaway:

- Yes, you can connect to models directly.
- But for scalability, flexibility, and maintainability, → Spring AI is the better choice.