

## Lecture 07: Asking Question to OpenAI Model

### Setting up the Controller

To connect Spring Boot with OpenAI using Spring AI, we create a controller:

- **Create a new controller class:** `OpenAIController`.
- **Mark it with** `@RestController`.
- **Define a method with** `@GetMapping("/api/{message}")`.
- **Accept user input with** `@PathVariable` String `message`.
- **Response:** return "Hello World" + `message` to confirm setup works.

```
@RestController
public class OpenAIController {

    private OpenAiChatModel chatModel;

    // Constructor Injection
    public OpenAIController(OpenAiChatModel chatModel) {
        this.chatModel = chatModel;
    }

    @GetMapping("/api/{message}")
    public String getAnswer(@PathVariable String message)
    {
        // Send message to OpenAI model
        String response = chatModel.call(message);
        return response;
    }
}
```

### Explanation of Code

- **@RestController** → Marks the class as a REST controller.
- **OpenAiChatModel** → Provided by Spring AI to talk to OpenAI.
- **Constructor Injection** → Spring automatically provides the `chatModel` bean.
- **@GetMapping("/api/{message}")** → Exposes an endpoint to accept user input.
- **chatModel.call(message)** → Sends the prompt to OpenAI and gets the response.

### How It Works:

- User hits URL: **`http://localhost:8080/api/what is Java?`**
- Controller captures "**`what is Java?`**" as a **`message`**.

- Spring AI sends it to the **OpenAI model**.
- Response returned directly as String.

### Output Example:

- **Input 1:** /api/what is Java
- **Response:** "Java is a high-level, object-orientated programming language..."
- **Input 2:** /api/tell me a joke
- **Response:** "Why do programmers prefer dark mode? Because light attracts bugs."