Introduction to Spring MVC

In this section, we explored Spring MVC, the web framework built into Spring Boot that allows us to build both traditional web applications and modern REST APIs. We covered key web concepts, the MVC design pattern, and how Spring Boot handles HTTP requests.

How the Web Works

When we browse a website, our browser (the client) sends a request to a server. The server processes the request and returns a response.

This communication happens using HTTP (HyperText Transfer Protocol), which defines how data is exchanged over the web.

Key Parts of an HTTP Request

- **Method** – Specifies what action we want to perform (GET, POST, PUT, DELETE).
- **URL** – The address of the resource being requested.
- **Headers** – Extra information (e.g., content type, authentication tokens).
- **Body** (Optional) – Contains data for the server (e.g., form submissions).

Key Parts of an HTTP Response

- **Status Code** – Indicates if the request was successful (200 OK, 404 Not Found).
- **Headers** – Metadata about the response.
- **Body** – The actual content returned by the server. It can contain
    - **HTML markup** (used in traditional web applications)
    - **Data in JSON format** (used in APIs)

How Web Pages Are Generated

Web pages are built using HTML (HyperText Markup Language). In web applications, HTML can be generated in two ways:

- **On the Server** – The server generates the full HTML page and sends it to the client. This technique is referred to as **Server-Side Rendering (SSR)**.
- **On the Client** – The server sends only raw data (JSON), and the client dynamically generates web page using JavaScript. This technique is referred to as **Client-Side Rendering (CSR)**.

What is JSON?

- **JSON (JavaScript Object Notation)** is a lightweight format used to structure data.
- It is commonly used in APIs because it is easy to read and process for both humans and computers.

What is an API?

- An **API (Application Programming Interface)** is a way for applications to communicate with each other.
- In web development, an API allows a client (browser, mobile app, or another system) to send/request data to/from a server.

Spring MVC

Spring MVC follows the Model-View-Controller (MVC) pattern, which organizes web applications into three parts:

- **Model** – Represents data and business logic. Typically, these are Java objects mapped to database entities.
- **View** – Defines how data is displayed. In traditional web apps, this is an HTML page generated using a template engine like Thymeleaf.
- **Controller** – Handles HTTP requests, processes data, and returns a response.

Handling Requests in Spring MVC

Spring MVC provides different types of controllers to handle HTTP requests.

- **@Controller** for returning HTML views.
- **@RestController** for returning data. Spring MVC automatically convert Java objects to JSON objects.

```
1  @Controller
2  public class HomeController {
3      @RequestMapping("/")
4      public String index() {
5          model.addAttribute("name", "Mosh");
6
7          return "index";
8      }
9  }
10
11 @RestController
12 public class MessageController {
13     @RequestMapping("/hello")
14     public Message sayHello() {
15         return new Message( "Hello World!");
16     }
17 }
```