Introduction

**1. Spring Framework: Spring MVC và Spring Boot**

* **Spring MVC**: Là mô hình kiến trúc MVC (Model-View-Controller) của Spring Framework giúp xây dựng các ứng dụng web một cách linh hoạt. Spring MVC phân tách rõ ràng các thành phần và logic xử lý giữa Model (dữ liệu), View (giao diện) và Controller (xử lý điều hướng), giúp mã nguồn dễ bảo trì và mở rộng.
* **Spring Boot**: Là một phần mở rộng của Spring Framework giúp phát triển ứng dụng Spring nhanh chóng, đơn giản bằng cách tự động cấu hình các thành phần cần thiết (Auto-Configuration). Spring Boot cung cấp các dependency và tích hợp sẵn các công cụ như embedded server (Tomcat, Jetty), giúp việc triển khai ứng dụng dễ dàng hơn mà không cần cấu hình nhiều.

**2. Khái niệm Dependency Injection (DI) và Inversion of Control (IoC)**

* **Inversion of Control (IoC)**: Là một nguyên lý mà trong đó việc kiểm soát sự khởi tạo và vòng đời của các đối tượng sẽ được giao cho Spring Container thay vì chính bản thân chúng tự kiểm soát. Điều này giúp giảm sự phụ thuộc giữa các đối tượng, giúp mã nguồn dễ bảo trì và kiểm thử.
* **Dependency Injection (DI)**: Là một kỹ thuật của IoC, trong đó các dependency (phụ thuộc) của một đối tượng sẽ được "tiêm" (injected) vào từ bên ngoài thay vì tạo trực tiếp bên trong đối tượng. Spring hỗ trợ DI để dễ dàng quản lý các phụ thuộc và tạo các đối tượng một cách hiệu quả.

**3. Spring Bean, Lifecycle, Scope và các Annotation khai báo Bean**

* **Spring Bean**: Là một đối tượng được quản lý bởi Spring IoC Container. Bean có thể được khai báo bằng các annotation hoặc cấu hình XML.
* **Lifecycle của Bean**: Spring quản lý vòng đời của bean từ khi nó được khởi tạo, thiết lập dependency, sử dụng, và cuối cùng là hủy bỏ. Các phương thức phổ biến để tùy chỉnh vòng đời là @PostConstruct và @PreDestroy.
* **Bean Scope**: Là phạm vi tồn tại của một bean, bao gồm:
  + singleton: chỉ có một instance duy nhất cho mỗi container.
  + prototype: tạo một instance mới mỗi khi được yêu cầu.
  + request, session, application, websocket là các scope cho ứng dụng web.
* **Các Annotation để khai báo Bean**:
  + @Component: Đánh dấu một lớp là bean.
  + @Service: Đánh dấu một lớp như một service layer.
  + @Repository: Đánh dấu lớp như là data access layer.
  + @Controller: Đánh dấu lớp xử lý request trong mô hình MVC.
  + @Bean: Khai báo một phương thức trả về một bean (thường dùng trong @Configuration class).

**4. Khái niệm cơ bản về HTTP Request, Response, HTTP Method, JSON và API**

* **Request**: Là yêu cầu từ client tới server, chứa thông tin như URL, headers, và body (dữ liệu kèm theo yêu cầu).
* **Response**: Là phản hồi từ server trả về client, chứa các thông tin như status code, headers, và body (dữ liệu phản hồi).
* **HTTP Method**: Các phương thức để thực hiện thao tác với tài nguyên trên server:
  + GET: Lấy dữ liệu.
  + POST: Tạo mới dữ liệu.
  + PUT: Cập nhật dữ liệu.
  + DELETE: Xóa dữ liệu.
* **JSON**: (JavaScript Object Notation) là định dạng dữ liệu nhẹ và dễ đọc, thường dùng trong trao đổi dữ liệu qua API.
* **API**: Application Programming Interface – một giao diện cho phép các phần mềm giao tiếp với nhau.

**5. Demo: Cài đặt môi trường và tạo Project Spring Boot trên IntelliJ**

* **Bước 1**: Cài đặt **Java SDK** (Java 17 trở lên) và **IntelliJ IDEA**.
* **Bước 2**: Mở IntelliJ, chọn **File > New > Project**.
* **Bước 3**: Chọn **Spring Initializr**, nhập thông tin dự án và chọn các dependency cần thiết (VD: Spring Web, Spring Data JPA, H2 Database).
* **Bước 4**: Bấm **Finish** để tạo project.

Các dependency quan trọng:

* **Spring Web**: Cung cấp các API để phát triển ứng dụng web.
* **Spring Data JPA**: Hỗ trợ làm việc với database qua JPA.
* **H2 Database**: Một database nhúng dùng cho phát triển và kiểm thử.

**6. Sơ lược về Maven và cách sử dụng**

* **Maven**: Là công cụ quản lý dự án Java, giúp quản lý dependency, build và deploy ứng dụng.
  + **pom.xml**: Là file cấu hình chính của Maven, chứa các dependency, plugin và cấu hình build.
  + Các lệnh Maven cơ bản:
    - mvn clean: Xóa các file được build trước đó.
    - mvn compile: Compile mã nguồn.
    - mvn package: Đóng gói ứng dụng thành file .jar hoặc .war.
    - mvn spring-boot:run: Chạy ứng dụng Spring Boot.

**7. Hướng dẫn sử dụng Postman**

* **Parameters**: Thêm các tham số vào URL để gửi yêu cầu.
* **Body Request**: Chọn Body và chọn raw, sau đó chọn định dạng JSON để gửi dữ liệu JSON.
* **Thực hiện yêu cầu**:
  + Chọn HTTP Method (GET, POST, PUT, DELETE).
  + Thêm endpoint URL và các thông tin cần thiết trong body hoặc parameters.
  + Nhấn **Send** để gửi yêu cầu và xem kết quả trong Response.