Answer the following

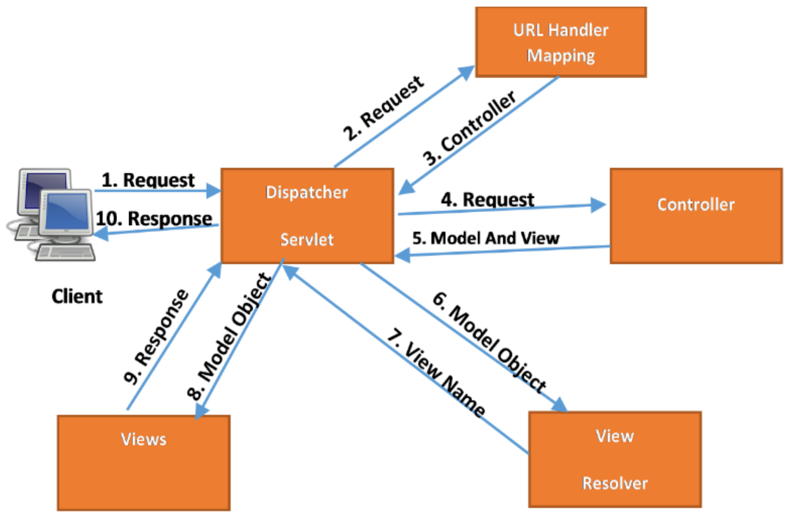
1. **Explain Spring Request-Response Life Cycle.**

Ans –

Spring Request-Response Life Cycle

The Spring Request-Response Life Cycle outlines how an incoming HTTP request is processed and responded to within a Spring or Spring Boot application. Below are the main stages:

1. **Client Request**
   The client sends an HTTP request (GET, POST, PUT, DELETE, etc.) to the server.
2. **DispatcherServlet**
   The DispatcherServlet is the central controller in Spring's Web MVC architecture. It intercepts the incoming request and delegates it to the appropriate handler.
3. **Handler Mapping**
   The DispatcherServlet consults the HandlerMapping to determine the appropriate controller (or handler) based on the URL pattern.
4. **Controller Execution**
   The matched controller method is invoked. It processes the request, performs business logic, interacts with services, and typically returns a ModelAndView or a ResponseEntity.
5. **View Resolver (for MVC)**
   If the controller returns a ModelAndView, the ViewResolver maps the logical view name to a physical view (e.g., a JSP or Thymeleaf template).
6. **Response Construction**
   The DispatcherServlet assembles the model data and the view to construct the HTTP response. If the controller returns a ResponseEntity, it directly serializes the response data to JSON, XML, or plain text.
7. **Response Dispatch**
   The constructed response is sent back to the client, completing the life cycle.



2. **What is the use of Maven in the Spring Boot Framework?**

Ans –

Use of Maven in the Spring Boot Framework

Maven is a build automation and dependency management tool widely used in Spring Boot projects. It simplifies project configuration, dependency resolution, and build processes.

Key Uses of Maven in Spring Boot

1. **Dependency Management:**
   Maven allows developers to include required libraries (dependencies) in a simple, declarative way using the pom.xml file. Spring Boot projects typically rely on Maven to include Spring Boot Starter dependencies**.**
2. **Build and Packaging:**
   Maven automates the compilation, testing, and packaging of Spring Boot applications into JAR or WAR files. The Spring Boot plugin for Maven (spring-boot-maven-plugin) makes it easy to package applications into executable JAR files.
3. **Spring Boot Plugins:**
   Maven integrates Spring Boot plugins to simplify tasks like running **the**

application (mvn spring-boot:run), creating executable JARs, and building Docker images.

4. **Version Control for Dependencies:**
   Maven uses the spring-boot-dependencies BOM (Bill of Materials) to manage dependency versions, ensuring compatibility across all libraries.
5. **Custom Profiles:**
   Maven profiles allow Spring Boot developers to manage different environments (development, testing, production) by providing environment-specific configurations.
6. **Integration with CI/CD Tools:**
   Maven integrates easily with CI/CD tools like Jenkins, GitHub Actions, and others, enabling automated builds and deployments.