Spring Framework

1. What is Spring Framework?

- Answer: Spring is a comprehensive framework for enterprise Java development. It provides a set of services for building Java applications such as dependency injection (DI), aspect-oriented programming (AOP), transaction management, and more.

2. What is Dependency Injection (DI)?

- Answer: Dependency Injection is a design pattern used in Spring to achieve loose coupling between components. In DI, an object's dependencies are provided (injected) rather than the object creating them itself.

3. What are the types of Dependency Injection in Spring?

- Answer: There are three types of Dependency Injection in Spring:
    - Constructor Injection
    - Setter Injection
    - Field Injection

4. What is Inversion of Control (IoC)?

- Answer: Inversion of Control is a core principle of Spring where the control of object creation and dependency management is inverted from the developer to the Spring container.

5. What is Spring Bean?

- Answer: A Spring Bean is an object that is managed by the Spring IoC container. Beans are defined in the configuration file (XML or Java-based) and can be injected wherever required.

6. Explain Spring AOP (Aspect-Oriented Programming).

- Answer: AOP is used to separate cross-cutting concerns (such as logging, security, etc.) from the business logic. It works by defining "aspects" and "advice" and applying them to target methods.

7. What is the Spring MVC framework?

- Answer: Spring MVC is a web framework in Spring that follows the Model-View-Controller design pattern for building web applications.

Spring MVC

1. What is Spring MVC, and how does it work?

- Answer: Spring MVC is a web framework that follows the Model-View-Controller pattern. It separates the application into three components:
  - Model: Represents data
  - View: Represents UI (JSP, Thymeleaf, etc.)
  - Controller: Handles user requests and updates the model.

2. What is the DispatcherServlet in Spring MVC?

- Answer: The DispatcherServlet is the front controller in Spring MVC. It handles all incoming HTTP requests and forwards them to appropriate handlers (controllers).

3. What is the role of @Controller and @RestController annotations?

- **Answer:**
  - @Controller: Indicates a class as a Spring MVC controller that returns a view (usually JSP).
  - @RestController: A specialized version of @Controller that returns data directly (JSON, XML) instead of a view.

4. What is @RequestMapping?

- Answer: The @RequestMapping annotation is used to map HTTP requests to handler methods in Spring MVC controllers.

5. What are Spring MVC Interceptors?

- Answer: Interceptors allow you to pre-process or post-process HTTP requests before and after the controller execution.

6. How to configure Spring MVC in a web application?

- Answer: Spring MVC can be configured using an XML configuration file or Java configuration using @Configuration and @EnableWebMvc**.**

Spring Boot

1. What is Spring Boot, and how is it different from Spring Framework?

- Answer: Spring Boot is a framework built on top of Spring that simplifies the setup and development of Spring-based applications. It eliminates boilerplate code and configuration by using sensible defaults, allowing developers to focus on building applications.

2. What are Spring Boot Starters?

- Answer: Starters are a set of pre-configured dependencies that can be included in a project to get started quickly with Spring Boot. Examples include spring-boot-starter-web, spring-boot-starter-data-jpa, etc.

3. What is the Spring Boot auto-configuration?

- Answer: Auto-configuration is a feature of Spring Boot that automatically configures beans based on the libraries available in the classpath. For example, if Spring Boot detects that a database driver is present, it will automatically configure a DataSource.

4. What is the Spring Boot Actuator?

- Answer: The Spring Boot Actuator provides production-ready features such as monitoring and managing Spring Boot applications, including health checks, metrics, and application info.

5. What are the advantages of using Spring Boot?

- Answer: Some advantages of Spring Boot include:
    - Simplified configuration
    - Embedded web server support (Tomcat, Jetty)
    - Fast development time
    - Embedded database support
    - No need for an external application server

6. How do you create a Spring Boot application?

- Answer: A Spring Boot application can be created by using the @SpringBootApplication annotation, which is a combination of @Configuration, @EnableAutoConfiguration, and @ComponentScan.

Comparison: Spring vs Spring MVC vs Spring Boot

Feature	Spring	Spring MVC	Spring Boot
Purpose	Core framework for Java apps	Web framework based on MVC pattern	Simplified framework for building Java applications
Setup	Requires more configuration (XML/Java-based)	Requires web.xml configuration & controller setup	Zero configuration, uses sensible defaults
Ease of Use	Moderate	Moderate	Very easy, out-of-the-box functionality
Web Support	No specific web support	Built-in web support via DispatcherServlet	Built-in embedded web server (Tomcat, Jetty)
Configuration	More complex	Requires configuration files	Minimal configuration with auto-configuration
Deployment	Needs external server (e.g., Tomcat)	Needs external server (e.g., Tomcat)	No need for an external server, embedded server support

Other Possible Questions Based on Experience

1. How would you handle exception handling in Spring MVC?
2. What are Spring Profiles, and how are they used in Spring Boot?
3. Explain the concept of Spring Boot's "CommandLineRunner" and "ApplicationRunner" interfaces.
4. What is a Spring Boot "Starter"?
5. What is the difference between @Component, @Repository, @Service, and @Controller annotations?
6. How do you handle security in Spring Boot applications?
7. What are the main advantages of using Spring Boot over Spring Framework?

8. How do you manage application properties in Spring Boot?
9. Explain the difference between @GetMapping, @PostMapping, and @RequestMapping.
10. How do you handle database transactions in Spring?