Lab 3-4, Suleman Asghar - StudentID:613820

1. What is Spring?
Answer:
Spring is a widely adopted open-source framework tailored for constructing enterprise-level Java applications. Renowned for its versatility, it offers a robust infrastructure facilitating the development of scalable and resilient Java applications across various layers.

2. What is Spring Boot?
Answer:
Spring Boot serves as an extension of the Spring Framework, simplifying the creation and deployment of production-ready applications with Spring. Its appeal lies in its convention-over-configuration approach, minimizing the need for boilerplate code and configuration. Spring Boot excels in developing microservices and standalone applications.

3. What is the relation between the Spring platform and Spring Boot?
Answer:
Spring Boot operates as a specialized project within the broader Spring platform, streamlining the development of Spring-based applications. Dependent on the core capabilities of the Spring Framework, Spring Boot embraces an opinionated methodology for application development, particularly for stand-alone and microservices-oriented applications.

4. What is the relation between the Spring platform and the Spring framework?
Answer:
The Spring Framework forms the foundational cornerstone of the Spring platform, introducing pivotal concepts like Inversion of Control (IoC). Projects within the Spring platform, including Spring Boot, Spring Data, and Spring Security, are constructed upon this framework, extending its functionalities to cater to specific application needs.

5. What is Dependency Injection and how is it done in the Spring platform/framework?
Answer:
Dependency Injection (DI) is a design pattern where a class's dependencies are provided externally rather than created within the class. In the Spring Framework, DI is facilitated through the Inversion of Control (IoC) container. The most common approaches include Constructor Injection, Setter Injection, Field Injection, and Interface-Based Injection. For instance, in Constructor Injection:

java
Copy code

```
@Configuration
public class AppConfig {
  @Bean
  public UserService userService() {
      return new UserServiceImpl(); // Creating a UserService bean
  }

  @Bean
  public OrderService orderService() {
    return new OrderServiceImpl(userService());
  }
}
```

6. What is Inversion of Control (IoC) and how is it related to Spring?

Answer:

Inversion of Control (IoC) is a fundamental design principle shifting control over object creation and management from the application code to a framework or container. Spring, deeply intertwined with IoC, incorporates tools and mechanisms within its framework, notably through its IoC container and Dependency Injection capabilities. This allows developers to focus on business logic while Spring manages object creation and assembly.