**Lab 3 Task 2 – Answer the questions**

1. Spring is an umbrella open-source framework that is specifically designed for Java programming language. It provides a set of tools and features to develop Java applications, particularly enterprise-level ones.
2. Spring Boot is one of the Spring projects that is used to create stand-alone, production-ready, and Spring-based applications and helped configure Spring’s project. Spring Boot enables developers to focus more on application features rather than boilerplate code and configuration. Here are some Spring Boot features:

* Starter POMs: provides a set of starter dependencies that make it easy to add common features to the Spring application. Example: Spring starter web dependency includes all of the dependencies that we need to create a web application.
* Dependency Management: Manage dependencies and ensure version compatibility among Spring and third-party libraries.
* Auto-Configuration: Automatically configure the Spring application based on the jar dependencies we have added. Example: When you add Spring starter web dependency, Spring boot will automatically set up the DispatcherServlet to dispatch the request to the controller and also an embedded server like Tomcat to run your application out of the box without deploying to an external web server.
* Convention over configuration: Allow the developer to customize any part of the auto-configuration. For example, if you want to change the server port from the default 8080 to 9090, you just need to add server.port=9090 to your application.properties file. If you have a complex configuration you can do the configuration class by using annotation @Configuration.

1. The relation between the Spring platform and Spring Boot is:

* Spring platform refers to the entire ecosystem of Spring projects that build on the Spring framework. This ecosystem includes Spring Boot, Spring Data, Spring Cloud…
* Spring Boot is built on top of the Spring framework. It aims to simplify the configuration and deployment of Spring-based applications by applying convention over configuration.

1. The relation between the Spring platform and Spring framework is:

* Spring platform refers to the entire ecosystem of Spring projects that build on the Spring framework. This ecosystem includes Spring Boot, Spring Data, Spring Cloud…
* Spring framework is a comprehensive programming and configuration model for building modern Java-based enterprise applications. It provides foundational support for different application architectures, including messaging, data access, and web applications.

1. Dependency injection is the mechanism that allows the class to use another class’s object without creating it in the class which enables us to develop loosely coupled applications. It's a form of inversion of control where an object's dependencies are not created by the object itself but are instead passed in (injected) by an external entity at runtime. This makes your code more modular, easier to test, and flexible. For example, Imagine a scenario where a class Car depends on another class Engine. Instead of the Car creating an instance of Engine directly, the Engine instance is created elsewhere (e.g., by the Spring container) and provided (injected) into the Car at runtime.

This is how DI is done in Spring, Spring manages DI through its IoC (Inversion of Control) container. When your application starts, Spring creates and configures the objects (known as beans) and manages their entire lifecycle, including injecting dependencies where needed.

* Component Scanning: Spring automatically detects classes annotated with @Component (and similar annotations like @Service, @Repository, @Controller) during the classpath scanning.
* Wiring Beans: Through the use of annotations like @Autowired, Spring wires objects together by injecting the declared dependencies.
* Configuration: Beyond annotations, Spring also supports XML-based or Java-based configurations for defining beans and their dependencies explicitly.

1. Inversion of Control (IoC) is a principle in software engineering where the control over the flow of execution of a program is inverted from the traditional approach. Instead of the application code managing the creation and flow of objects and their dependencies, this responsibility is delegated to an external entity (such as a framework or container). This external entity creates objects, wires them together by setting necessary dependencies, and manages their lifecycle but at a certain point it give back control to application code.

The Spring Framework implements the IoC principle through its IoC container. The container manages the instantiation, configuration, and assembly of objects, known as beans, within an application. This management is done based on metadata that the application provides. This metadata can be supplied in XML, annotations, or Java-based configuration.