UNDERSTANDING IMPORTANT TERMINOLOGIES

1. **@Component**
    a. **@Component** is a Spring framework annotation used to indicate that a class is a candidate for auto-detection as a Spring-managed bean.
    b. By annotating a class with **@Component**, you are telling the Spring container to create an instance of that class and manage its lifecycle, including its initialization and destruction. This instance is known as a bean and is stored in the Spring application context.
2. **Dependency**
    a. A dependency is an object that another object depends on. A dependency can be another Spring-managed bean, a service, a data access object, or any other object that provides a needed functionality.
3. **@ComponentScan**
    a. @ComponentScan is a Spring framework annotation used to specify the base package for Spring to scan for annotated components.
    b. When the Spring container starts up, it searches for classes annotated with @Component, @Controller, @Service, @Repository, and other user-defined annotations within the specified package and its sub-packages.
4. **Dependency Injection**
    a. The Spring DI framework is responsible for creating and injecting the dependencies between objects in a Spring-based application.
    b. Identify beans, their dependencies and wire them together.
5. **Inversion of Control (IoC)**
    a. The term "inversion of control" (IoC) refers to the inversion of the traditional flow of control in a program. In a traditional program, the application code controls the flow of execution, deciding when to create objects and call methods on them. However, in an IoC environment, the control flow is inverted, and the framework or container takes control of the object creation and method calling.
    b. In other words, with IoC, the framework or container manages the lifecycle of objects and their dependencies, and it is responsible for calling the methods of these objects as needed, rather than the application code directly controlling these operations.