HOMEWORK 2

1 – IOC and DI means ?

IoC (**I**nversion **o**f **C**ontrol) : It's a generic term and implemented in several ways (events, delegates etc).

DI (**D**ependency **I**njection) : DI is a sub-type of IoC and is implemented *by* constructor injection, setter injection or Interface injection.

The advantages of IOC and DI are:

- Allows us to create loosely coupled classes
- Easy unit test writing
- Modular programs
- Manageability

2 – Spring Bean Scopes?

When you create a bean definition what you are actually creating is a *recipe* for creating actual instances of the class defined by that bean definition. The idea that a bean definition is a recipe is important, because it means that, just like a class, you can potentially have many object instances created from a single recipe.

- **Singleton:** Scopes a single bean definition to a single object instance per Spring IoC container.
- **Prototype:** Scopes a single bean definition to any number of object instances.
- **Request:** Scopes a single bean definition to the lifecycle of a single HTTP request; that is each and every HTTP request will have its own instance of a bean created off the back of a single bean definition.
- **Session:** Used for web applications. An instance is created every time an HTTP session occurs.
- **Global-session:** This scopes a bean definition to a global HTTP session**.**

3 – What does @SpringBootApplication do ?

The @SpringBootApplication annotation is a convenience annotation that combines the **@EnableAutoConfiguration**, **@Configuration** and the **@ComponentScan** annotations in a Spring Boot application. These annotations do the following:

@EnableAutoConfiguration – This enables Spring Boot's autoconfiguration mechanism Auto-configuration refers to creating beans automatically by scanning the classpath.

@ComponentScan – Typically, in a Spring application, annotations like @Component, @Configuration, @Service, @Repository are specified on classes to mark them as Spring beans. The @ComponentScan annotation basically tells Spring Boot to scan the current package and its sub-packages in order to identify annotated classes and configure them as Spring beans.

@Configuration – Designates the class as a configuration class for Java configuration.

4 – What is Spring AOP? Where and How to use it?

One of the key components of Spring Framework is the **Aspect oriented programming (AOP)** framework. Aspect-Oriented Programming entails breaking down program logic into distinct parts called so-called concerns. The functions that span multiple points of an application are called **cross-cutting concerns** and these cross-cutting concerns are conceptually separate from the application's business logic.

5 – What is Singleton and where to use it?

Singleton" is a pattern, not a class. A given class may be a singleton - a class that is designed to allow only one instantiation. It's similar to a static class in that it creates a single point of invocation. A common use for a singleton is to create a single point of access to some resource, to force centralization of access, for example a logger class that you want all log messages passed to, so it can decide which to output and which to ignore.

All implementations of the Singleton have these two steps in common;

● Make the default constructor private, to prevent other objects from using the new operator with the Singleton class.

● Create a static creation method that acts as a constructor. Under the hood, this method calls the private constructor to create an object and saves it in a static field. All following calls to this method return the cached object.

6 – What is Spring Boot Actuator and Where to use it?

Spring Boot Actuator automatically activates production-ready features (health check, disk usage, heap dump etc.) of applications and offers a structure that allows interacting with different HTTP endpoints.

7 -What is the primary difference between Spring and Spring Boot?

Basis	*Spring*	*Spring Boot*
Where it's used?	Spring framework is a java EE framework that is used to build applications.	Spring Boot framework is mainly used to develop REST API's
Key feature	The primary or most important feature of the Spring framework is dependency injection(Dependency Injection (DI) is a design technique that removes dependencies from computer code, making the application easier to maintain and test).	The main or primary feature of the Spring Boot is Autoconfiguration(Simply described, Spring Boot autoconfiguration is a method of automatically configuring a Spring application based on the dependencies found on the classpath.)

Why it's used	Its goal is to make Java EE (Enterprise Edition) development easier, allowing developers to be more productive.	Spring Boot provides the RAD(Rapid Application Development) feature to the Spring framework for faster application development.
Type of Application Development	Spring framework helps to create a loosely coupled application.	Spring Boot helps to create a stand-alone application.

8 – Why to use VCS?

It increases the traceability of the development process of our projects. VCS also enables the collaboration of team members and makes sure everyone is working on the same version of the project.

9 – What are SOLID Principles? Give sample usages in Java?

SOLID principles are object-oriented design concepts relevant to software development. SOLID is an acronym for five other class-design principles: **S**ingle Responsibility Principle, **O**pen-Closed Principle, **L**iskov Substitution Principle, **I**nterface Segregation Principle, and **D**ependency Inversion Principle.

Principle	Description
Single Responsibility Principle	Each class should be responsible for a single part or functionality of the system.
Open-Closed Principle	Software components should be open for extension, but not for modification.
Liskov Substitution Principle	Objects of a superclass should be replaceable with objects of its subclasses without breaking the system.
Interface Segregation Principle	No client should be forced to depend on methods that it does not use.
Dependency Inversion Principle	High-level modules should not depend on low-level modules, both should depend on abstractions.

10 - What is RAD model?

RAD Model or Rapid Application Development model is a software development process based on prototyping without any specific planning. In RAD model, there is less attention paid to the planning and more priority is given to the development tasks. It targets at developing software in a short span of time.

11 - What is Spring Boot starter? How is it useful?

Spring Boot provides a number of **starters** that allow us to add jars in the classpath. Spring Boot built-in **starters** make development easier and rapid. **Spring Boot Starters** are the **dependency descriptors**.

In the Spring Boot Framework, all the starters follow a similar naming pattern: **spring-boot-starter-***, where * denotes a particular type of application. For example, if we want to use Spring and JPA for database access, we need to include the **spring-boot-starter-data-jpa** dependency in our **pom.xml** file of the project.

12 – What is Caching? How can we achive caching in Spring Boot?

It reduces the required number of requests to a database, thus it increases the performance of applications as sending requests to databases is costly. Spring Boot provides an auto-configured cache manager. It can be configured manually and further with spring annotations.

13 – What & How & Where & Why to logging ?

Logging is keeping a record of all data input, processes, data output, and final results in a program. Logging is important for software developing, debugging, and running. With logging, you can leave a trail of breadcrumbs so that if something goes wrong, we can determine the cause of the problem. There are a number of situations like when we are expecting an integer, we have been given a float and we can a cloud API, the service is down for maintenance, and much more. Such problems are out of control and are hard to determine.

14 - What is Swagger? Have you implemented it using Spring Boot?

Swagger is an open source project used to describe and document RESTful APIs. It is language-agnostic and is extensible into new technologies and protocols beyond HTTP. The current version defines a set HTML, JavaScript, and CSS assets to dynamically generate documentation from a Swagger-compliant API. These files are bundled by the Swagger UI project to display the API on browser. Besides rendering documentation, Swagger UI allows other API developers or consumers to interact with the API’s resources without having any of the implementation logic in place.