HW#2

1 - IOC and DI means?

Inversion of control (IoC) is a design paradigm, a programming technique. It has a purpose of giving more control to the targeted components of the application. Dependency injection (DI) is a pattern for to create instances of objects that other objects rely on without knowing at compile time which class will be used to provide that functionality. So IoC uses dependency injection as a mechanism which activates the components providing the specific functionality.

2 - Spring Bean Scopes?

Bean scope is the life cycle and visibility of the bean that we use. There are six types of spring bean scopes at the latest version of the Spring framework:

- Singleton
- Prototype
- Request
- Session
- Application
- Websocket

Singleton: With this scope container creates a single instance of that bean and all requests for that bean name will return the same cached object. Singleton scope is the default value if no other scope is specified.

Prototype: This scope returns a different instance every time when it is requested from the container.

Request, Session, Application, Websocket: These four additional scopes that are only available in a web-aware application context. They are used less often in practice.

The *request* scope creates a bean instance for a single HTTP request, while the *session* scope creates a bean instance for an HTTP Session.

The *application* scope creates the bean instance for the lifecycle of a ServletContext and the *websocket* scope creates it for a particular WebSocket session.

3 - What does @SpringBootApplication do?

@SpringBootApplication annotation is available from Spring 1.2 onwards. It is an annotation which contains three other annotations of Spring. These annotations are @Configuration, @ComponentScan and @EnableAutoConfiguration. So when we use @SpringBootApplication annotation, we basically become like we used these three annotations.

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

AOP (aspect-oriented programming) is a programming paradigm that enables the application to be adaptable to changes. It isolates the concerns of the application and reduces code clutter. Also it improves the maintainability and readability of the code. It has all the benefits of OOP as well. In OOP we take the advantage of classes to achieve modularity. But in AOP we achieve modularity through aspects. The essence of AOP is encapsulating functionalities that are common while at the same time enabling the application to leverage those functionalities as need be. The most important benefit of using AOP is we just need to write our aspects once and then we can reuse it wherever we need to in our application. So with this way we can reduce the complexity of the source code of our application and make our code clean. That is the most important benefit why we should choose to use AOP. For to implement AOP in our application we need to isolate the aspects in our application from the business logic. The aspects should be independent and they shouldn't have any dependency on the application. After that we should apply these aspects wherever they are needed by the application.

5 - What is Singleton and where to use it?

Singleton is a creational design pattern in Java. Singleton ensures that only one object of its kind exists and provides a global point of access to it. We use it when we want to restrict the limit of the number of object creation to only one.

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

Spring Boot actuator is a Spring Boot module which allows us to monitor and manage our Spring Boot application. It contains the actuator endpoints. For to enable Spring Boot actuator endpoints, we need to add a dependency in our starter pom file. Just with this dependency, we are able to use Spring Boot actuator features.

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

Spring framework focuses on providing flexibility to build applications in Java. It has a lot of important features that shortens our application build but it requires us to deal with all configuration parts. The most important feature in Spring is dependency injection.

Spring Boot is an extension of the Spring Framework. It aims to shorten the code length and provide the easiest way to develop Web Applications. The most important feature of Spring Boot is Autoconfiguration. We don't need to deal with configuration, it automatically configures the classes based on the requirement. With less configuration it shortens our code length.

8 - Why to use VCS?

Version control systems (VCS) are software tools which help developers manage changes to source code during development process. They help us to track and manage changes to the files over time. Especially when more than one person works on a project, it is really important to keep track of changes and keep every team member working on the right version. With version control systems we can trace every change ever made. And if something goes wrong with new changes we can undo those changes with the help of version control systems features.

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

SOLID principles are design principles encourage us to create more maintainable, understandable and flexible software. They are object-oriented approach that are applied to software structure design.

Explanations of the five principles are:

- *Single Responsibility Principle (SRP):* "One class should have one and only one responsibility"
- *Open-Closed Principle (OCP):* "Software components should be open for extension, but closed for modification"
- *Liskov Substitution Principle (LSP):* "Derived types must be completely substitutable for their base types"
- *Interface Segregation Principle (ISP):* "Clients should not be forced to implement unnecessary methods which they will not use"
- *Dependency Inversion Principle (DIP):* "Depend on abstractions, not on concretions"

10 - What is RAD model?

Rapid Application Development (RAD) is a software development model. This model is generally based on prototyping and iterative model with no specific planning. So RAD approach concentrates on development and creating prototypes as soon as possible rather than planning tasks. This approach provides quickly to see and use and to provide feedback regarding the requirements.

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

Spring Boot starters are dependency management features. They make development easier and rapid. For example when we want to use a Spring Boot feature in our project we use the Spring Boot starter which is relevant with that feature and we are able to use it in our project after that moment. With a simple Spring Boot starter code we create access to important features and modules without any other action.

12 - What is Caching? How can we achieve caching in Spring Boot?

A cache is a local memory area which is used to temporarily store a copy of frequently accessed data. So that future requests for that data does not have to be retrieved again from the original storage location. Caching allows to efficiently reuse previously retrieved data and this process improves performance.

When we want to use cache mechanism in a Spring Boot project we need to add "*spring-boot-starter-cache*" dependency in the pom.xml file. With this cache starter we can use cache annotations in our code.

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

Logging is the process of collecting and storing data over a period of time in different systems or environments with the features that are specified previously. We may log every change occurred in a project or we can log only when errors occur or log every operation done by a user. So logging conditions may change by our purposes. The reason to use logging system is to be able to track every changes that have been made in a project. Is sounds like similar with version control systems but logging is just about to be able to see what happened in a project. We can not use logging for to change or undo an action.

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

Swagger is a specification for documenting REST API. It specifies the format (URL, method and representation) to describe REST web services. The format is both machine-readable and human-readable. Swagger allows us to describe the structure of our APIs so that machines can read them. It supports a wide range of frameworks.

In Spring Boot we need to add Swagger dependencies to our build configurations file for to be able to use Swagger. When we write web services using a framework Swagger scans the code and exposes the documentation on some URL. After that any client can consume this URL and learn how to use our REST web services.