**Phase 3 : 7 days**

**Day 1**

**20-03-2023**

**Spring framework and spring boot**

**Framework. Framework is a collection of classes and interfaces which internally connected to each other to perform a specific task. If we develop any enterprise application without framework we need to do more configuration. But if we develop the application with the help of any framework take core those all configuration. Design pattern. Design pattern means best practise or solution for repeating problems. If we develop any application using any framework we are following standards because all framework internally follow design pattern.**

**Collection framework is like a data structure.**

**MVC : Model View Controller**

**View 🡪 html or jsp**

**Controller 🡪 servlet**

**Model -🡪 javabean class or entity class**

**Service class**

**Dao class**

**Resource class**

**Jsp and servlet object creation is taken care by web container. Web container is a part of web service ie tomcat.**

**But web container doesn’t create object of java bean class, service class, dao class as well as resource class etc.**

**To improve the model class we were use EJB (Enterprise Java Bean).**

**View 🡪jsp and html**

**Controller 🡪 servlet**

**Model 🡪 EJB**

**To run the ejb application we required ejb container. Ejb container is a part of application server application server are heavy weighted server. Ejb is very complex technologies**

**Framework**

**Struts : struts is open source web framework which internally follow mvc architecture. Struts is known as controller centric framework.**

**JSF (Java Server Faces) : jsf is open source web framework part of oracle. It internally follow mvc architecture. JSF is known as View centric framework.**

**JSF Vs Angular / React JS / View JS**

**Hibernate : hibernate is replacement of JDBC.**

**Spring Framework : spring is open source layered architecture or onion architecture framework. Spring is light weighted framework.**

**Spring framework provided lot of modules which help to develop any type of application on demand.**

**Spring framework modules**

**Spring core**

**Spring context**

**Spring web**

**Spring mvc : spring mvc internally follow mvc architecture framework.**

**Spring mvc is known as model centric framework.**

**Spring MVC Vs EJB**

**Spring mvc allow use to integrate with struts or JSF.**

**Spring jdbc**

**Spring orm spring orm allow us to integrate with existing orm tool like hibernate, jpa or iBaties etc.**

**Spring data**

**Spring rest**

**Spring boot**

**Spring security**

**Spring micro service**

**Spring core and context**

**IOC : Inversion of control**

**DI : Dependency Injection**

**IOC is a concept. IOC is a design programming pattern.**

**In place of creating or maintaining any resources explicitly allow to maintain by container. If container create it will maintain the life cycle of those resources. Whenever we need pull from container, use it and leave it. If we create we need to maintain properly.**

**DI : DI is an implementation of IOC.**

**Using two ways we can do the DI in spring framework**

1. **Constructor base di**
2. **Setter base di**

**Using DI we pull the resource from container. Like objects**

**Spring container is light weighted container. If we want to achieve only ioc and di we don’t need to web server or application need to add few jar files inside our project can we can achieve di and ioc. Container are part of those jar files.**

**To achieve these concept we need to configuration using**

1. **Using Xml file**
2. **Using annotation**

**DI using constructor base and setter base with XML Configuration**

**Singleton design pattern : if we want to create only one object of that class but more than one reference then we have to use singleton design pattern.**

**Using constructor base we have to achieve fully dependencies.**

**Using setter base di we can achieve partial dependencies.**

**In parameterized constructor base di number of parameter as well as type of parameter must be match.**

**In setter base number of parameter as well non mandatory to match.**

**Day 2**

**21-03-2023**

**Auto wiring : Auto wired is one the great features of spring framework which help to enable you to inject the complex object dependency implicitly rather than doing explicitly with help ref attribute on property or constructor-arg tags.**

**If we use autoWired=”byteType”. Spring framework automatically scan the xml file if it found bean definition for that complex object it automatically to the di for that complex object.**

**In bytype we need only one bean definition for that complex object to achieve di.**

**If we have more than one bean definition then we have to use typeName.**

**In typename bean id name and complex property reference name must be match.**

**Spring framework do the di for primitive property implicitly. If we want to do the di for complex property we need to use ref attribute.**

**DI using Annotations**

1. **We can use xml file for enable the annotation**
2. **We can use configuration class to enable the annotation.**

**@Component : it is type of annotation provided by spring framework we need to use on pojo or javabean class.**

**It is equal to <bean class=”packagename.className”></bean>**

**By default id is classname in camelnaming rule. Means if class name is one word then id is classname in lower case like**

**Address class name then id is address.**

**If class name more than one word then**

**AddressInfo class name then id is addressInfo**

**@Autowired annotation we need to write on complex object.**

**@Value : this annotation we need to use to primitive property to set default value.**