1. @SpringBootApplication : it indicates that it is a configuration class and also triggers auto-configuration .

Combination of : - >

1. @EnableAutoConfiguration: - it will help our application automatically configures based on the dependency we added in pom.xml file
2. @ComponentScan : - This enables the spring bean dependency injection feature by using @Autowired . Components which are annoted with @Component,@Service,@@Repository or @Controller automaticallyj registered as spring beans . These beans can be injected by using @Autowired annotation .
3. @Configuration : - it enables the java based configuration for spring boot application .

The class with @SpringBootAplicaiton is considered as main class . It kicks starts the application by invoking SpringApplication.ru(Class name , args) method .

@AutoWired condition:

Constructor dependency injection for mandatory dependency and setter injection for optional dependency

Condition 1:

The dependencies which are injected using @Autowired should be available in spring container , if container does not find bean then it throws NoSuchBeanDefinationException .

Condition 2: if more than one bean of the same type are available in the container then the framework throws exception indicating more than one bean is available for autowiring . to handle it we use @Qualifier . @Qualifier helps to give cusotme name to bean . eg:

@AutoWire

@Qualifier(“custoRepo)

Private cutomerRepository customerRepository;

Q. what is @PostConstruct and its uses ?

Ans : When the constructor is called , the bean is not yet initialized i.e No dependencies are injected . In the @PostConstruct method , bean is fully initialized and we can use the dependency .

This @PostConsruct method will be invoked only once in the bean life cycle .

It might be bean instantiate multiple times by the container in its internal working but it guarantees that @PostConsgtruct be invoked only once .

* because when the constructor is called, the bean is not yet initialized - i.e. no dependencies are injected. In the @PostConstruct method the bean is fully initialized and you can use the dependencies.
* because this is the contract that guarantees that this method will be invoked only once in the bean lifecycle. It may happen (though unlikely) that a bean is instantiated multiple times by the container in its internal working, but it guarantees that @PostConstruct will be invoked only once.

Q. What is domain objects ?

Ans : It is nothing but Entity class object like Employee ;

Q. What is life time of bean and scope ? By default scope is always Singleton

Lifetime of bean depends on its scope . its scope can be defined while declaring it in the configuration metadata file .

A bean can be singleton or prototype . A bean with singleton scope is initialized during the container starts up and same bean used whenever request comes from Application .

For prototype, a new instance is created for every bean request from the application .

Q. Logging API ?

JDK logging API 2. Apache Log4j 3. Common Logging API

By default Springboot Configures logging via Logback to log the activities of libraries that our application use even we don’t use any loger in our application .

If we dont add any logger dependency still we can use our custom log in application .

Logger loger = LogerFactory.gerLogger(Employee.class); >> this will do logger work for student class .

Q . How to write log in file?

We need to add loggin.file.name and logging.file.path property in out application.properties file . (See google)