Created by:-Lalit Ingale

Spring-Boot

1)What is Spring Boot?

➔-Spring Boot is an open-source framework built on top of the spring framework for java developers.
-Spring Boot was developed by Pivotal Software(now the part of VMware), It was first released in 2014.
-In Spring Boot, 'Boot' refers to the concept of rapidly initializing a spring application by handling all essentials set-ups and configuration.

2)Where is the use of Spring Boot?

➔i)Web Application
ii)RESTful Web Services
iii)Microservices
iv)Batch Processing
v)Integration with Other Technologies
vi)Testing
vii)Monitoring and Management

2)What is Framework?

➔Framework is a software that is built on the top of one or more technologies to simplify the application development process by generating the common logics of the application dynamically.

3)Spring Boot Core Principle.

➔**i)Opinionated Defaults:-** Spring Boot gives you smart, ready-to-use default settings so you can get started quickly without needing to configure everything yourself.

ii)Convetion Over Configuration:- If you follow the standard rules or naming conventions, Spring Boot will automatically configure things for you — no need to write extra setup code.

iii)Auto-Configuration:- Spring Boot automatically sets up your application based on the libraries you have added and the settings it finds — you don't have to configure everything manually.

iv)Embedded Servers:- Spring Boot comes with a built-in (embedded) web server, so you don't need to install or configure a separate server like Tomcat or Jetty — it runs right inside your application.

4)Spring Boot Features?

➔i)We can do coding for spring boot Apps in multiple jvm based languages like java, groovy, Kotlin….since all are habituated with…java, so java is recomanded.

ii)Spring Boot and its module gives new annotation by combining multiple relevant annotations to reduce the burden on the programmer.

-@RestController=@Controller+@ResponseBody

-@SpringBootApplication=@Configuration+@EnableAutoConfiguration +@ComponentScan

iii)We can develop and test spring boot Apps without installing external DB s/w and webserver/Application server s/w becoz the spring boot app execution itself give Embedded Db s/w and embedded servers based on the starter we add.

iv)Spring boot starters reduce no of jar files that should be added by programmer to the project..the each starter give main jar files, dependent jar file and relevant jar files.

4)What is Spring Boot Starters?

➔Spring Boot Starters are the "Dependency Descriptors"

-Dependency→External libraries or JARs

-Descriptor→Configure specific JAR's and their version
-Spring Boot Starters Combine all the necessary libraries (JAR files) for a particular feature or technology into a single dependencies.
-**For ex:-"**Spring-boot-starter-Web" dependency will provide all the Web features in our application.

5)What is the use of Spring Boot Starters?
➔**i)Simplifies Configuration:-**Starters automatically include the right libraries and settings we need, so that we write code faster.
ii)Boost Productivity:-Saves time and effort by providing sensible defaults and configuration, allowing developers to focus on writing business logic rather then managing dependencies.
iii)Simplifies Configuration:-Ensures that all included dependencies are compatible with each other and the Spring Boot version being used, reducing the risk of conflicts.

6)Which are the most important Spring Boot Starters?
➔**i)Spring-boot-starter:-**It is the core starter that includes auto-configuration support, logging, and YAML processing.
ii)Spring-boot-starter-web:-It provides dependencies to build web applications, including Spring MVC, Restful Services, and embedded Tomcat.
iii)Spring-boot-starter-data-jpa:-It provide dependencies for Spring Data JPA, Hibernate and database connection pool.
iv)Spring-boot-starter-security:-It includes Spring Security and it's dependencies for securing application.
v)Spring-boot-starter-thymeleaf:-It integrates the Thymeleaf templating engine with Spring MVC.
vi)Spring-boot-starter-test:-It includes testing libraries like Junit, Hamcrest and Mockito.

vii)Spring-boot-starter-actuator:-It provides production-ready features to help monitor and manage your application
viii)Spring-boot-starter-aop:-It brings in support for aspect-oriented programming using Spring AOP and Aspect J.

5)What is the difference between dependent jar file and relevant jar file?
➔**-Dependent JAR:-** A JAR file that your application must have to run or compile — like Spring libraries if you're using Spring Boot.
-Relevant JAR:- A JAR file that is related or useful to your project, but not always required — like an email library if your app sends emails.

6)Define @SpringBootApplication?
➔**i)@Configuration:-**This annotation in Spring indicates that the class can be used to define bean configuration within the Spring application context.
ii)@ComponentScan:-It tells Spring to scan the specified package and it's sub packages for component, configuration and service to manage.
iii)@EnableAutoConfiguration:-It automatically configures the Spring application based on its dependencies and the contents of the classpath.
-@SpringBootApplication annotation serves as the "Entry point for a SpringBoot application.

7)Which are the important components are used in spring Boot?
➔**i)Spring Boot Auto Configure:-** Auto Configuration means Spring Boot automatically configures your application based on the libraries (dependencies) you add.
ii)Spring Boot CLI:-A commond line interface, based on ruby, to start/stop spring boot created application.
iii)Spring Boot Actuator:- Spring Boot Actuator is a set of tools that helps you monitor and manage your Spring Boot application. It's like a health monitor for your app, providing useful information about how the app is performing.

iv)Spring Boot Starters:- Spring Boot Starters are pre-configured sets of dependencies that help you quickly set up a specific type of application. They are like "starter kits" that come with all the necessary tools (libraries) to perform a particular task.

v)Spring Boot Tools:-The Maven and Gradle build tool as well as the custom Spring Boot Loader is included in this project.

8)Different Categories of Starters.

➔i)Cloud Starter
ii)Web Starter
iii)Data Store Starter
iv)Spring Core Starter
v)Netflix Oss Starter

vi)Message Starter
vii)Batch Starter

9)What is “static folder” in Spring Boot Project?

➔-The “static folder” is typically used to store static web resources such as HTML, CSS, JavaScript, Images and other assets.
-These files are served directly by the Web Server without any processing by the backend application.

10)What is “templates folder” in Spring Boot Project?

➔-The “templates folder” is commonly used to store dynamic web pages or templates, Which are typically processed by a templating engine like Thymeleaf, FreeMarker or Velocity.
-These templates are rendered dynamically by the server side code before being sent to the client.

11)Define “application.properties file” in Spring Boot Project?

➔-In “application.properties file”, We typically provide configuration settings for various aspect of the Spring Boot application.
-for example database settings, Server port configuration, logging levels, and other application-specific properties.

12)What is Conditional annotation?

➔Conditional annotation in Spring Boot’s auto-configuration determine whether specific configuration should be applied based on conditions like the presence of classes, properties or beans. This enables automatic configuration tailored to the application’s environment and dependencies.

13)What is the difference between Spring and Spring Boot?

➔-The following features are defined as the main difference between Spring famework and Spring Boot:-

Features	Spring Framework	Spring Boot
What is it?	A core framework for building Java application	A tool built on top of Spring to make development faster and easier
Setup	Manual setup—you write a lot of configuration	Auto setup—most configuration is done for you
Web Server	You have to set up Tomcat/ Jetty separately	Comes with built-in Tomcat, ready to run apps
Dependencies	You add each dependeny manually	Comes with Spring Boot Starter(pre-made set of dependencies)
Project Structure	You have to create and organize folders,files	Spring Boot creates Structure automatically
Learning Curve	Require more time to learn and configure	Easier for beginners and faster development
Focus	Gives full control, more	Focuses on productivity and

	flexible	simplicity

Run method	Needs XMl/config files+WAR deployment	Just run your main() method ,like a normal Java program
Focus	Gives full control, more flexible	Focuses on productivity and simplicity

17)What is Stereo type annotation?

➔Multiple annotations doing the same work of configuring java class as spring bean are called stereo type annotation.

18)Define @Named, @Inject and @Resource?

➔**i)@Named:-**To configure the java class as spring bean and also to resolve ambiguity problem.

ii)@Inject:-Alternate to @Autowired

iii)@Resource:-Alternate to @Autowired+@Qualifier

19)What is the advantage of spring/spring boot frameworks compare to Struts or JSF?

➔Struts or JSF are just web frameworks…which can be used to just develop the web application…where as spring/spring boot are application framework/JEE frameworks which can be used to develop to standalone Apps, webapplication, distributed Apps, Microservice architecture applications and etc.

20)Spring/Spring boot alternate to EJB?

➔No, EJB is given to develop the Distributed Apps, where as Spring/spring boot is given to develop the all kinds of apps including the distributed apps.

21)What happens if we configure service, DAO classes using @Component annotation?

➔Yes, we can configure….But additional behaviour required for the spring bean should be added manually/explicitly.

22)What is JDBC Connection Pool?

➔A JDBC Connection Pool is like a group of reusable connections that are already open and ready to use.

-Instead of opening a new connection every time, the application borrows a connection from the pool.

-After it finishes using the connection, it returns it back to the pool for others to use.

23)What is HikariCp in JDBC?

➔HikariCP is a high-performance JDBC connection pooling library for Java. It's one of the fastest and most lightweight connection pool implementations available.

24)What is the difference between pool and cache/buffer?

➔-Pool means set of items i.e gives the reusability of same items.

-cache/buffer means a set of different items i.e gives the reusability of different items.

25)What is lombok?

➔Lombok is a Java tool that helps you write less code. It automatically adds things like getters, setters, constructors, and more, so you don't have to write them yourself.

26)Define @Data?

➔@Data is a shortcut from Lombok that saves you from writing a lot of boring code in Java classes.

-When you use @Data it automatically creates Getters,Setters,toString(), equals() and constructor.

27)What is Serializable interface?

➔In Java, Serializable is a marker interface used to tell the program.

-"Hey, this object can be saved, sent, or converted into a format (bytes) that can be restored later."

28)When we add the starters like spring-boot-starter-JDBC/JPA/…what is default DataSource obj that comes pointing to JDBC con pool?

➔We will get HikariDataSource object.

29)We configure DAO class with @Service annotation and Service class with @Repository Annotation?

➔ No, we should not configure a DAO class with @Service and a Service class with @Repository. That's a wrong usage of annotations.

- @Repository → used for DAO (Data Access Object) classes.
  These classes interact with the database.
- @Service → used for Service classes.
  These contain business logic.

30)If we place all the 4 datasources(Tomcat cp,hikari cp,apache dbcp2, apache dbcp) related dependencies at a time in spring boot App then what happens?

➔As per Algorithm priority the Hikari CP Data Source will be taken for AutoConfiguration.

31)How does the @EnableAutoConfiguration annotation of @SpringBootApplication knows which classes of the currently added jar files should be configured as spring beans through auto configuration?

➔The @EnableAutoConfiguration annotation searches for spring factories files in META-INF folders of all the jar files added to the CLASSPATH but finds in spring-boot-autoconfigure<ver>jar file…This file internally linked with another file called META-INF\spring\org.springframework.boot.auto-configure.AutoConfiguration.imports➔This file contain all the Configuration classes that needs to executed for Autoconfiguration activity➔All these configuration classes and their nested and imported configuration classes related @Bean methods execute to get the objects of java classes as spring beans through AutoConfiguration activity.

32)When spring-boot-starter-jdbc is added to CLASSPATH what is default DataSource we get?

➔HkariCP DataSource

33)Which is industry standard DataSource to use in real projects?

➔Hikari CP as of now becoz its performance(especially speed of creating jdbc con objects in the jdbc con pool)

34)What is the difference between getForEntity() and getForObject() methods?

➔**-getForEntity()** return type is ResponseEntity<T> which contains response body(output),headers,status code and etc.
-getForObject() return type is Object which gives only response body(output)i.e It does not give response headers,status code and etc.

35)What is RestTemplate?

➔ RestTemplate is a helper class provided by Spring that makes it easy to call REST APIs (HTTP services) from your Spring Boot application.

36)What is webClient?

➔ WebClient is a modern and powerful tool in Spring used to make HTTP calls to other APIs, just like RestTemplate, but with more features and better performance.

37)Define @ConfigurationProperties?

➔- It's a way to read values from your application.properties or application.yml file and put them into a Java class automatically.
- So you don't have to write @Value again and again for every property. You just create one class, and Spring fills in all the values for you.

38)What is the difference between @Value and @ConfigurationProperties annotations?

➔

@Value	@ConfigurationProperties
a)Given by spring f/w,so it can be used in both spring Apps and spring boot Apps.	a)Given by Spring boot,So it can be used only in spring boot Apps.
b)Perform Field Level Injection by acc-essing private properties of spring bean class through reflection api.	b)Perform setter Injection by calling of the spring bean class.
c)placing setter methods in spring bean is not required.	c)required
d)It is field/property level annotation.	d)It is class level,method level annotation.
e)Perform single value injection to single property of spring bean.	e)Support Bulk Injection.

f)Common prefix for keys of properties is not required.	f)Common prefix for keys in properties file is required.
g)The keys in properties file and the property name in spring bean class need not to match.	g)The keys in properties file and the property name in spring bean class needs to match.
h)If we specify wrong key in @Value annotation we get IllegalArgumentException.	h)If the keys in properties file is not matching with spring bean property name..then No Exception will be raised it just ignore value injection to that spring bean property.
i)Can be used to inject the values to simple type,wrapper type,String type properties.	i)Can be used to inject the values to simple type,wrapper type,String type array,collection HAS-A type spring bean properties.

39)If @Value and @ConfigurationProperties tries to inject two different values to same spring bean property then which will be taken as final value?

➔The @ConfigurationProperties annotation assigned/injected value will be taken as final value becoz @Value performs fields Injection right after spring bean instantiation where as @ConfigurationProperties performs Setter injection by calling the setter method…So setter method injected value overrides the field injection value.

40)How to configure user-defined properties file in spring boot application?

➔Using @PropertySource annotation as spring bean class level annotation.

41)If application.properties and user-defined properties file contains same key with different value and if we try to inject that key related

value to spring bean property by using @Value or @ConfigurationProperties then which value will be taken?

➔If user-defined properties file is configured using @PropertySource annotation then the value kept application.properties file will come as the final value.

➔If user-defined properties file is configured using spring.config.import key of application.properties/yml file then the value kept user-defined properties file will be taken as the final value.

42)Can we use @ConfigurationProperties and @Value annotation in single bean class?

➔Yes, can be done ,Infact few values to spring bean properties can be injected using @Value and the values to few other properties using @ConfigurationProperties annotation(Only in spring boot apps)

43)What is YML/YAML?

➔-YAML stands for Yet Another Markup Language.
-It is a simple, human-readable format used for writing configuration files.

44)If we place both application.properties and application.yml having same keys and different values can you tell me what happens?

➔The value given to application.properties will override the values given in application.yml

45)What is the difference between properties file and yml file?

➔

Properties file	Yml file
a)There is no specification providing rules and guidelines to develop proper-	a)There is specification providing rules and guidelines to develop the

ties files It is just keys=values.	yml file(www.yaml.org)
b)can be used only in java	b)can be used in java,python,ruby, groovy and etc.
c)No way related to JSON format.	c)Super set of JSON
d)can be used in both spring f/w and spring boot f/w.	d)cannot be used in spring f/w..i.e only supported in spring boot.
e)nodes/levels in the keys may have duplicates.	e)same nodes/levels will not repeat.
f)It is not hierarchal data.	f)It is hierarchal data.
g)Custom properties file can be configured in spring boot App directly by using @PropertySource and no PropertySource Factory class is required.	g)Custom yml file can be configured in spring boot App by using @PropertySource and by developing specifying PropertySource Factory class.
h)while working with profiles in spring /spring boot we can not place multiple Profiles in single properties file.	h)we can place multiple profiles in single yml file having separation with "---".
i)Spring or Spring boot App directly loads and reads the properties file content.	i)every yml file will be converted to properties file content before loading and reading.
j)Use properties file when no of keys are less and the nodes/levels in keys are not repeating.	j)Use yml file when no of keys are more and nodes/levels keys are repeating.
k)Give bit extra performance.	k)Give bit less performance.
l)takes more time for typing becoz of repeated nodes in the keys.	l)takes less time becoz of the no repeated nodes in the key.

46)How to configure user-defined yml/yaml file to the spring boot application?

➔use spring:

config:

import:<filename> key in application.yml to specify the user-defined yml file name/properties file name

➔use spring.config.import key in application.properties file to specify the user-defined yml/yaml file/properties file name.

47)How can we configure custom properties file in spring/spring boot Application?

➔In Spring Apps using @PropertySource Annotation, In Spring boot Apps using @PropertySource annotation or using spring.config.import key of the application.properties file or spring:

```
config:
  import: key of the application.yml file
```

48)What is Profiles?

➔A profile is like a label or a tag that tells Spring which settings or code to

use depending on the environment.

i) Development (dev)
ii) Testing (test)
iii) Production (prod)

49)If one target class is having multiple possible dependents to inject then how can we inject specific dependent of our choice without disturbing the source code of the Application/Project?

➔Using spring boot profiles(Best and recomanded)

50)If we activate two different profiles using application.properties and using system property then which will be taken as the final profile?

➔The “profile specified” in the System property will be taken as the final value.

51)How many ways are there to activate profiles in spring boot app?

➔There are multiple ways to use…but mainly 3 approaches can be considered.
i)using application.properties/yml(Best)
ii)using System property(-Dspring.profiles.active=dev)
iii)using Programatic approach(by using the methods of SpringApplication class.

52)What is the meaning of @Profile(“default”) or what is default profile?
➔- @Profile("default") is used in Spring to mark a bean that should only be created when no profile is explicitly active. It acts as the default configuration when spring.profiles.active is not set. This is useful for providing fallback settings for general use.

53)What is the difference between fallback profile and default profile and also child profile?
➔**-Fallback Profile:-**The profile that executes when requested active profile is not available.

-default profile:-The profile that executes when there is no activate profile in the application.
-child profile:-The profile that is included in application.properties by specifying the profile name to reuse keys and values of specific profile in application.properties indirectly as fallback profile.

54)What is the difference between not adding @Profile for spring bean and adding @Profile(“default”) for spring bean?
➔-If @Profile is not added for spring bean that will work for all profiles when profile is activated..It will even work if no profile activated and also for child profile that is activated eg:service,controller class.
-@Profile(“default”) based spring bean will work only when active profiles are not specified.

55)What is Runners?

➔ Runners are special blocks of code that run automatically when your Spring Boot application starts — right after everything is set up.

56)What is the difference between placing static block in spring bean class and working with Runner class?

➔

Static block of spring bean class	Runner class
a)It is java feature, can be used in both spring and spring boot Apps.	a)It is a feature of spring boot can be used only in spring boot app.
b)Executes automatically when the IOC container Loads the spring bean class.	b)Executes automatically as part of spring boot Application startup activity performed by the SpringApplication.run()method.
c)Does not allow to pass data/args from outside.	c)Allows to pass data/args to run() method.
d)Non-static data can not be accessed in this static block.	d)Allows us to access both static and non-static data.
e)useful to place class level one-time executing logics.	e)Useful to place spring boot Application level one time executing logic.
f)Static block does not support Exception Propagation.	f)run() method of Runner class supports the Exception Propagation.

57)What is the difference between CommandLineRunner and ApplicationRunner?

➔Both are one time execution classes/spring bean having run() method..But the way they get cmd line args to the run() is different.

-CommandLineRunner gets the given cmd line args in the form of String[] where as ApplicationRunner gets the given cmd line args in the form of ApplicationArguments object where we can categorize the cmnd line args into optional and non-optional args.

58)Though, we are not adding any starters to Spring boot project, How does it gets multiple jar files as the maven/gradle dependencies to the Project?

➔In every Spring boot Project, One parent spring boot project will be imported from the maven central repo using<parent>tag.

➔Based on the parent spring boot project's version, lots of things will come to our spring boot project which is nothing but a child project.

a)Basic spring,spring boot jar files and dependent, relavent jar files.

b)required plugins

c)required Project Properties.

59)Define Debugging the Application using Eclipse IDE?

➔-Debugging is useful to feel application code execution line by line

-Using debugging, we can identify the errors, fix the errors

-Using Debugging, we can feel the data flow

-Using Debugging, we can understand the code written by others

-Using Debugging, we can enable monitoring/watching on the variable data, object data, and expression.