1. What is Maven primarily used for?

A. Web server configuration
B. Version control
C. Java project build and dependency management
D. Operating system installation

---

2. Which file contains configuration for a Maven project?

A. settings.xml
B. build.gradle
C. pom.xml
D. application.properties

---

3. Which tag in pom.xml is used to inherit a parent project?

A. <inherit>
B. <base>
C. <parent>
D. <extends>

---

4. What is the default directory where Maven stores downloaded JAR files?

A. /usr/lib/maven
B. target/
C. .m2/repository/
D. src/main/java/

---

5. What does the clean lifecycle do in Maven?

A. Cleans source code
B. Deletes test cases
C. Removes generated files like compiled classes
D. Cleans local repository

---

6. What is the final output after all parent and child POMs are merged?

A. Merged POM
B. Final Build File
C. Effective POM
D. Complete XML

---

7. Which Maven lifecycle phase compiles the source code?

A. validate
B. test
C. package
D. compile

---

8. Which command is used to build a Maven project and install it to the local repo?

A. mvn build
B. mvn package
C. mvn compile
D. mvn install

---

9. What does the site lifecycle do?

A. Runs the project
B. Cleans the project
C**. Generates project documentation**
D. Installs dependencies

---

10. Which of the following is NOT a Maven repository type?

A. Local
B. Central
C. Remote
D**. Private (It's a subtype, not official core type)**

---

11. What does the <build> tag in pom.xml define?

A. Project metadata
B. Source directories
C. Build-related plugins and configuration
D. Java version

---

12. What is the main purpose of build plugins in Maven?

A. Store project info
B. Add user-defined build actions
C. Run database migrations
D. Compile JAR to WAR

13.Who is the original creator of the Spring Framework?
a) James Gosling
b) **Rod Johnson**

c) Josh Long
d) Craig Walls

14.In which year was the Spring Framework initially released?
a) 2000
b) 2005
c) **2003**
d) 2010

15.What was the main goal of introducing the Spring Framework?
a) To create mobile applications
b) To replace servlets
c) To simplify database access
d) To simplify enterprise Java development by replacing EJB

16.Which version of Spring introduced full annotation support?
a) Spring 2.x
b) **Spring 3.0**
c) Spring 4.x
d) Spring 5.x

17.Which version of Spring added support for Reactive Programming with WebFlux?
a) Spring 4.x
b) Spring 2.x
c) **Spring 5.x**
d) Spring 6.x

18.Which Spring version introduced native support for Jakarta EE and AOT compilation?
a) Spring 4
b) Spring 5
c) Spring 3
d) **Spring 6**

19.Which of the following is true about Spring 1.x?
a) Introduced WebFlux
b) Supported full annotations
c) **Based on XML configuration and BeanFactory**
d) Introduced ApplicationContext

20.Which version of Spring started support for annotations?
a) Spring 1.x
b) **Spring 2.x**
c) Spring 3.0
d) Spring 4.0

---

21. What is the primary purpose of using the Spring framework in Java development?
a) To replace the Java compiler
b) To manage operating system tasks
c) **To simplify Java development by providing lightweight and comprehensive infrastructure**
d) To convert Java to C++

---

22. Which of the following best describes the term "Inversion of Control (IoC)" in Spring?
a) A design where objects control their own creation
b) A process of compiling Java classes at runtime
c) **A design principle where the control of object creation is delegated to the Spring container**
d) A Java keyword for memory management

---

23. Which code snippet shows Spring's approach to dependency injection?
a) Car c = new Car();
b) Car c = new Car(engine, wheels);
c) **Car c = ctx.getBean(Car.class);**
d) Car c = new SpringCar();

---

24. What is a Spring Bean?
a) A JavaScript function in Spring
b) A static class in Spring
c**) A simple Java object managed by the Spring container**
d) A servlet in Spring

---

25. What does the Spring container primarily manage?
a) HTML rendering
b) Lifecycle of servlets
c) **Lifecycle of Spring beans**
d) Java memory allocation

---

26. Which of the following is NOT a feature of Spring?
a) Unified transaction management
b) **Heavyweight and monolithic design**
c) Smooth integration with technologies like JDBC, JNDI
d) Elimination of boilerplate code

---

27. What annotation is used in Spring for unified transaction management?
a) @Autowired
b) @Service
c) **@Transactional**
d) @Component

---

28. What kind of transaction management does Spring support?
a) Only local
b) Only global
c) Only manual
d) **Both local and global**

---

29. Which of the following is an example of boilerplate code that Spring helps eliminate in JDBC?
a) Declaring main method
b) Creating GUI
c) **Managing database connection lifecycle**
d) Importing Java packages

---

30. In Spring, which principle is followed for better decoupling and testability?
a) Global variables
b) Singleton pattern
c) **Programming to interfaces and using POJOs**
d) Hard coding dependencies

---

31. Which of the following is not a module in Spring 5?
a) spring-context
b) spring-core
c) **spring-browser**
d) spring-data

---

32. What is the purpose of the Spring Core module?
a) Provides UI components
b) Handles REST API
c) **Provides IoC container and dependency injection**
d) Manages database connections directly

33. What is true about Dependency Injection in Spring?
a) Objects create their own dependencies
b) **Dependencies are injected externally by the container**
c) It increases tight coupling
d) It prevents testing

---

34. Which of the following is NOT a valid way to inject dependencies in Spring?
a) Constructor Injection
b) Setter Injection
c) Field Injection
d**) Loop Injection**

---

35. In which Spring configuration style are beans typically defined using @Bean methods inside @Configuration classes?
a) XML config
b) **Java config**
c) Mixed config
d) Properties config

---

36. Which annotation is used for field injection in Spring?
a) @Bean
b) @InjectBean
c**) @Autowired**
d) @ComponentScan

37. What is true about Spring stereotype annotations?
a) They prevent bean registration
b) They are used only in XML
c) **They help Spring auto-detect and register classes as beans**
d) They replace Java classes with XML tags

38. Which one is not a Spring stereotype annotation?
a) @Controller
b) @Component
c) **@Bean**
d) @Service

39. Which configuration method was introduced after Spring 2.5 and uses annotations extensively?
a) **Java Config**
b) XML Config
c) Hibernate Config
d) YAML Config

40. What is the role of ClassPathXmlApplicationContext in Spring?
a) It starts a servlet container
b) It loads Java configuration classes
c) **It reads XML bean configuration files**
d) It connects to a database

41. What does @SpringBootApplication combine?
a) @ComponentScan only
b) **@Configuration, @ComponentScan, @EnableAutoConfiguration**
c) @Autowired, @Bean, @EnableWebMvc
d) @Service, @Repository, @RestController

---

42. What is the default behavior of @ComponentScan?
a) Scans only external JARs
b) Scans full project
c) **Scans current package and sub-packages**
d) Doesn't scan anything by default

---

43. Which annotation is used to mark a class as Java-based configuration?
a) @Component
b) @Bean
c) **@Configuration**
d) @Qualifier

---

44. What does @EnableAutoConfiguration do?
a) Automatically creates a web server
b) Turns off bean creation
c) **Automatically configures beans based on classpath and properties**
d) Scans XML files

---

45. Which Spring container interface provides lazy initialization?
a) ApplicationContext
b) **BeanFactory**
c) WebApplicationContext
d) SpringBootApplication

46. What is eager bean loading?
a) Beans are created only when required
b) Beans are created at runtime
c) **Beans are created at startup**
d) Beans are created using XML only

---

47. Which interface is the base for Spring IoC container?
a) BeanManager
b) **BeanFactory**
c) BeanContext
d) ApplicationManager

---

48. Which of these is used to define a bean manually in Java config?
a) @Autowired
b) **@Bean**
c) @Component
d) @Service

---

49. What does @Component do?
a) Declares a database connection
b) Defines a web controller
c) **Registers a class as Spring bean**
d) Creates configuration file

---

50. Which annotation is specialization of @Component?
a) **@Service**
b) @Autowired
c) @Bean
d) @Configuration

---

51. What is the default scope of a Spring bean?
a) Prototype
b) Request
c) **Singleton**
d) Session

---

52. When is @Autowired on constructor optional?
a) When using XML config
b) When field is private

c) **When there is only one constructor**
d) When bean is prototype

---

53. What does @Qualifier do?
a) Deletes beans
b) **Specifies bean name when multiple same-type beans exist**
c) Defines bean scope
d) Adds versioning

---

54. Which context class is used for Java-based configuration?
a) ClassPathXmlApplicationContext
b) WebApplicationContext
c) **AnnotationConfigApplicationContext**
d) FileSystemXmlApplicationContext

---

55. In constructor injection, Spring injects dependencies:
a) After object creation
b) Through XML
c) **At the time of bean creation**
d) Via field setters

56. In Spring, what does the term *"bean scope"* refer to?
a) The inheritance of a bean
b) The type of bean used
c) **The number of instances and their lifecycle**
d) The method used to define the bean

---

57. Which of the following is **true** about *singleton* bean scope in Spring?
a) A new bean is created for each HTTP request
b) **One shared bean instance per Spring container**
c) One bean per user session
d) One bean per class loader

---

58. How is a **Singleton Class** different from a **Singleton Bean** in Spring?
a) Singleton bean can have multiple instances
b) Singleton class uses Spring configuration
c) **Singleton class enforces only one instance in JVM, while Spring allows one per container**
d) Singleton bean must use the Singleton design pattern

---

59. Which of the following is **not** a valid Spring bean scope?
a) singleton

b) request
c) prototype
d) **public**

---

60. Which scope will create a **new bean instance every time** it is requested from the container?
a) singleton
b) session
c) **prototype**
d) application

---

61. What is the lifecycle phase where Spring container creates the bean and injects its dependencies?
a) **Init**
b) Construct
c) Use
d) Destroy

---

62. In a Spring bean lifecycle, the @PostConstruct method is called:
a) Before the constructor
b) **Right after dependencies are injected**
c) After bean is destroyed
d) Before bean is created

---

63. What method is called by the Spring container **before destroying** a bean?
a) @Init
b) @Stop
c) **@PreDestroy**
d) @AfterConstruct

---

64. In Java Servlets, which lifecycle method is used to release resources?
a) init()
b) service()
c) paint()
d) **destroy()**

---

65. What is the purpose of lifecycle callback methods like init() and destroy()?
a) To manage HTTP sessions
b) To let the programmer handle creation of beans
c) **To allow the container to manage the object lifecycle**
d) To build a singleton pattern

---

66. Why is the init() method preferred over constructor for accessing container services in Spring?
a) Constructors are faster
b) Constructor injection is not supported
c) **Container services are not available during constructor execution**
d) init() is executed before the constructor

67. What is the **first** phase in the Spring bean lifecycle?
a) Initialization
b) **Instantiation**
c) Dependency Injection
d) Bean Post Processing

68. During which lifecycle phase does **dependency injection** happen?
a) After @PostConstruct
b) After initialization
c) **During Populate Properties phase**
d) During Bean destruction

---

69. If a bean implements BeanNameAware, what does Spring provide to it?
a) Its scope
b**) The bean name**
c) The ApplicationContext
d) Its lifecycle methods

---

70. Which interface must be implemented to get access to the **BeanFactory**?
a) **BeanFactoryAware**
b) BeanNameAware
c) ApplicationContextAware
d) InitializingBean

---

71. When does the method postProcessBeforeInitialization() get called?
a) After bean destruction
b) **Before @PostConstruct**
c) After properties are set
d) Right after bean instantiation

---

72. Which of the following is a valid **BeanPostProcessor**?
a) ApplicationBeanProcessor
b) **AutowiredAnnotationBeanPostProcessor**

c) RequestScopeProcessor
d) BeanFactoryProcessor

---

73. What annotation is used to specify a **custom init method** in Java configuration?
a) @InitMethod
b) @Initialization
c) **@Bean(initMethod="...")**
d) @StartMethod

---

74. Which interface method is called if a bean implements InitializingBean?
a) afterBeanCreated()
b) **afterPropertiesSet()**
c) onInit()
d) initialize()

---

75. In the Spring Bean lifecycle, when does postProcessAfterInitialization() run?
a) Before constructor
b**) After all initialization steps**
c) During dependency injection
d) After @PreDestroy

---

76. What happens at the final stage of a Spring bean lifecycle?
a) Bean is injected
b) Bean is ready to use
c) **Bean is destroyed**
d) Bean is scanned

---

77. Which method is called during bean destruction if the bean implements DisposableBean?
a) **destroy()**
b) stop()
c) cleanUp()
d) dispose()

---

77. What annotation is used to define a method that should run before the bean is destroyed?
a) @BeforeDestroy
b) @Cleanup
c) **@PreDestroy**
d) @DestroyMethod

78. What does **Autowiring** in Spring do?
a) Creates a new database connection
b) **Automatically injects dependencies**
c) Compiles the Spring application
d) Starts a thread

---

79. Which annotation is used to auto-inject dependencies in Spring?
a) @InjectBean
b) @**Autowired**
c) @ComponentScan
d) @BeanConfig

---

80. What is the key difference between **Autowiring** and **Explicit Wiring**?
a) Autowiring requires XML only
b) Explicit wiring requires annotations
c) **Autowiring injects dependencies automatically**
d) There is no difference

---

81. Which annotation helps resolve conflicts when multiple beans of the same type exist?
a) @BeanName
b) **@Qualifier**
c) @PrimaryOnly
d) @Unique

---

82. What does the @Primary annotation do?
a) Marks the bean as deprecated
b) Sets bean priority to low
c) **Marks a bean as the default when multiple candidates exist**
d) Prevents a bean from being autowired

---

83. Where can the @Autowired annotation be applied?
a) Only in XML
b) Only on constructors
c) **On constructors, fields, or setter methods**
d) Only on interface methods

---

84. What is the major drawback of **field injection** using @Autowired?
a) Code becomes non-compilable
b) Requires more XML configuration
c) **Hard to test and mock in unit testing**
d) Cannot be used in Spring Boot

---

85. Which form of autowiring is best for **mandatory dependencies** and **immutability**?
a) Field injection
b) Setter injection
c) **Constructor injection**
d) Runtime injection

---

86. In Spring 4.3+, which annotation is **optional** if there's only one constructor?
a) @Bean
b) **@Autowired**
c) @Component
d) @Inject

---

87. What is a benefit of **setter-based dependency injection**?
a) Promotes immutability
b) **Allows runtime reconfiguration**
c) Reduces memory usage
d) Avoids dependency injection altogether

---

88. When is **explicit wiring** preferred over autowiring?
a) In unit testing
b) In lightweight mobile apps
c) **When full control over dependencies is needed**
d) In prototype-scoped beans

89. What is the primary feature of Spring Boot?

a) Simplifies data access

b)Automates code generation

c) Simplifies project setup

d) Enhances UI design

Explanation :- Spring Boot simplifies the development of new Spring applications through convention over configuration and automatic setup.

90. What does the @SpringBootApplication annotation do?

a) Enables JDBC

b) Configures a web application

c) Declares a configuration class

d) Starts a Spring context

91. How does Spring Boot handle configuration?

a)Through YAML files

b) Through properties files

c) Through XML files

d) Manually via code

92.What embedded servers does Spring Boot support?

a) jetty
b) **Tomcat**
c) Undertow
d) Netty

93. Which module of Spring Framework is the foundation for Spring Boot?

a) Spring MVC
b) **Spring Core**
c) Spring AOP
d) Spring ORM

94. Spring Boot is best suited for creating what type of applications?

a) Batch applications
b) **Web applications**
c) Enterprise applications
d) Desktop applications

95. What role does Spring Boot DevTools play?

a) Improving build speed
b) **Automatic restart**
c) Code generation
d) Dependency management

96.What annotation is used to define a Spring Boot application's main class?

a) @SpringApplication
b) @**EnableAutoConfiguration**
c) @SpringBootConfiguration
d) @SpringBootApplication

97.If a Spring Boot application fails to start due to a port conflict, what is the typical solution?

a) **Change the port in the properties file**
b) Reinstall Spring Boot
c) Update Spring dependencies
d)Change the server

98. What is the purpose of the application.properties file in a Spring Boot application?

a) **To configure the application's properties**
b) To store application resources
c) To manage external dependencies
d) None of the above

99. How does Spring Boot allow overriding default configurations?

a) Using external properties files
b) Using command-line arguments
c) Using environment variables
d) **All of the above**

100. Which annotation is used to specify the location of the application.properties file in a Spring Boot application?

a) **@PropertySource**
b) @ConfigurationProperties
c) @Value
d) @SpringBootApplication

101. What is the purpose of the @ConfigurationProperties annotation in a Spring Boot application?

a) **To map properties from external sources to JavaBean properties**
b) To enable Spring's auto-configuration
c) To define application-specific properties
d) None of the above

102. How can you specify a custom location for the application.properties file in a Spring Boot application?

a) **Using the spring.config.location property**
b) Using the @PropertySource annotation
c) Using the @ConfigurationProperties annotation
d) Using the @Value annotation

103. How does an Introduction advice do this in Spring?
a) web proxy
b) dynamic proxy
c) implements org.springframework.net.bundle interface
d) none of the mentioned

Explanation: Introduction works by adding an interface to the dynamic proxy.

104. Which method is used to gracefully shut down all the bean processes after closing the spring container?
a) destroy method
b) shutdownHook
c) none of the mentioned
d) all of the mentioned

Explanation: ShutdownHook method gracefully shuts down each bean process before closing the container.

105. Spring Security algorithms to secure passwords.
a) MD5
b) SHA
c) None of the mentioned
d**) All of the mentioned**

106. What does Spring Boot provide to simplify application development?
a) A built-in web server
b) Starter dependencies
c) Automatic configuration
d) All of the mentioned

107. Which of the following annotations is used to indicate a controller in Spring MVC?
a) @Component
b**) @Controller**
c) @Service
d) @Repository

108. Which of the following is a characteristic of Spring's Bean Lifecycle?
a) Beans are created at runtime only
b) All beans are created as singletons
c) **Beans can be configured to execute custom initialization methods**
d) Beans are destroyed when the application shuts down only

109. Which Spring annotation is used to mark a class as a repository?
a) @Service
b**) @Repository**
c) @Component
d) @Controller

110. What is the difference between @Component and @Service?

a) **No difference, just semantic**
b) @Service is for DAO layer

c) @Component can’t be scanned
d) @Service creates a prototype bean

111. Which scope creates a new bean instance every time it is requested?

a) singleton
b) request
c) session
d) **prototype**

112. What is the purpose of the @Primary annotation?

a) It disables a bean
b**) It gives the highest priority for bean selection**
c) It prevents autowiring
d) It sets scope to singleton

113. Which interface provides a callback method after all properties are set in a Spring bean?

a) BeanFactoryAware
b) ApplicationContextAware
c) **InitializingBean**
d) DisposableBean

---

114. Which annotation allows you to specify a method to be called before a bean is destroyed?

a) @PostConstruct
b) @**PreDestroy**
c) @Init
d) @BeforeDestroy

---

115. In which phase does Spring call postProcessBeforeInitialization()?

a) After bean creation
b) Before property population
c**) Before @PostConstruct**
d) After @PreDestroy

---

116. Which of the following best describes @Autowired?

a) Injects dependency by name
b**) Injects dependency by type**
c) Used only on methods
d) Only works with XML configuration

---

117. Which is not a valid Spring bean scope?

a) session
b) view
c) **global**
d) request

---

118. Which BeanPostProcessor is responsible for handling @Autowired annotations?

a) InitDestroyAnnotationBeanPostProcessor
b) BeanValidationPostProcessor
c) **AutowiredAnnotationBeanPostProcessor**
d) ApplicationListenerProcessor

---

119. Which bean wiring approach offers more control and clarity?

a) Auto-wiring
b) **Explicit wiring**
c) Field injection
d) Constructor injection

---

120. What is the main disadvantage of field injection?

a) Easy to test
b) More boilerplate code
c) **Difficult to mock in unit testing**
d) Doesn’t support immutability

---

121. What is the recommended approach for injecting mandatory dependencies?

a) Field injection
b) Setter injection
c) **Constructor injection**
d) Static method injection

---

122. What is the main use of @Bean annotation?

a) To define an autowired bean
b) **To declare a bean in Java config class**
c) To mark class for scanning
d) To inject dependency

123. What happens if multiple beans match the type during autowiring and no @Qualifier is specified?

a) **Spring throws an exception**
b) Spring injects the first one
c) Spring creates a new bean
d) Spring skips injection

124. What is true about singleton scope in Spring?

a) A new instance is created every time
b) One instance per request
c) One instance per session
d) **Only one instance per container**

125. In which file do we define bean scope in XML-based configuration?

a) **beans.xml**
b) application.properties
c) spring-config.yaml
d) web.xml

126. What is the purpose of the BeanFactoryAware interface?

a) To allow bean to modify other beans
b) To access the bean name
c) **To get reference to BeanFactory**
d) To listen for application events

127. Which scope is best suited for web applications where a new bean is needed per HTTP request?

a) Singleton
b) Prototype
c) Session
d) **Request**

128. What is the main disadvantage of prototype scope?

a) **Memory overhead due to multiple instances**
b) Shared state among threads
c) Not suitable for services
d) Limited to web applications

129. What should you check first if changes to application.properties are not reflecting in your Spring Boot application?

a) **The file is located in the correct directory**
b) The properties are correctly named

c) The application is restarted after changes
d) The file has the correct file permissions

130. What is a common cause of a Spring Boot configuration issue where a specific bean is not being injected as expected?

a) Incorrect bean definition in the context
b**) Conditional annotations blocking bean creation**
c) Incorrect profile settings activated

d)Missing component scanning configuration

131. The core interface of Spring email support is:
a) **MailSender**
b) EMail
c) All of the mentioned
d) None of the mentioned

132. Spring Batch works with a system scheduler:
a) autosys
b) cron
c) none of the mentioned
d) **all of the mentioned**

133.Which special type of advice is used to implement an interface?
a) **Introduction**
b) Before
c) After
d) AfterSpecial

Explanation: It allows your objects to implement an interface dynamically by providing an implementation class for that interface.

134. Introduction advice helps in implementing multiple inheritance

a) **True**
b) False

Explanation: You are able to introduce multiple interfaces with multiple implementation classes to your objects at the same time.

135. In introduction advice you have to modify class to introduce new methods
a) True
b) False

Explanation: You can introduce methods to your existing classes even without source code available.

136. Annotation used to declare an introduction

a) Before
b) After
c) @DeclareParents
d) None of the mentioned

Explanation: In this aspect, you can declare an introduction by annotating an arbitrary field with the @DeclareParents annotation.

137. How to introduce counter field to original bean class?

a) **Using Spring AOP**
b) Implementing interface
c) AspectJ Pointcut
d) None of the mentioned

138. A bean can be requested by:-

a) getBean method
b) reference from another bean using autowiring, property etc
c**) all of the mentioned**
d) none of the mentioned

Explanation: When a bean is requested by the getBean() method or a reference from other beans, Spring will decide which bean instance should be returned according to the bean scope.

139. Which attribute is used to set the scope of the bean?

a) setScope
b) **scope**
c) getScope
d) none of the mentioned

Explanation: Scope attribute defines the scope of a bean.

140. Which one is the default scope of the beans?

a) Prototype
b) Session
c) Request
d) **Singleton**

141. Which scope creates a new bean instance each time when requested?

a) Singleton
b) **Prototype**
c) Session
d) Request

142. Session Creates a single bean instance per HTTP request, only valid in the context of a web application?
a) True
b) **False**

Explanation: Session Creates a single bean instance per HTTP session, only valid in the context of a web application.

143. Which of the following are considered valid beans?
a) Singleton
b) Prototype
c) **All of the mentioned**
d) None of the mentioned

144. Which interface is used to perform initialization of beans?
a) **InitializingBean**
b) Disposablebean
c) None of the mentioned
d) All of the mentioned

Explanation: Spring allows your bean to perform initialization callback methods afterPropertiesSet() by implementing the InitializingBean and interfaces.

145. Which interface is used to perform destruction of beans?
a) InitializingBean
b) **Disposablebean**
c) None of the mentioned
d) All of the mentioned

Explanation: Spring allows your bean to perform destroy callback methods destroy() by implementing the DisposableBean and interfaces

146. Alternate way of initialization method is:-
a) **init-method attribute**
b) afterPropertiesSet
c) destroy-method attribute
d) none of the mentioned

147. Alternate way of destruction method is:-
a) init-method attribute
b) afterPropertiesSet
c) **destroy-method attribute**
d) none of the mentioned

Explanation: A better approach of specifying the destroy callback methods is by setting the destroy-method attributes in your bean declaration.

148. Which annotation is used as a substitute of initialization method?
a) **@PostConstruct**
b) @PreDestroy
c) None of the mentioned
d) All of the mentioned

Explanation: Using JSR annotation.

149. Which configuration can be used for Dependency Injection?
a) XML Configuration
b) Annotation Configuration
c) Java Based Configuration
d) **All of the mentioned**

150. Which of the following is a characteristic of Spring's Bean Lifecycle?
a) Beans are created at runtime only
b) All beans are created as singletons
c) **Beans can be configured to execute custom initialization methods**
d) Beans are destroyed when the application shuts down only

Explanation: Spring allows configuring custom initialization methods for beans, which can be executed during the bean lifecycle.

151.Which of the following is NOT a feature provided by Spring Boot?
A. Auto-configuration
B. Embedded Servers
C. XML-based configuration by default
D. Opinionated starter dependencies

Answer: C
Spring Boot prefers convention over XML configuration and encourages Java-based config.

152. What happens if you have both application.properties and application.yml in your Spring Boot project?
A. Only application.properties is used
B. Only application.yml is used
C. Both are merged with application.yml taking precedence
D. Both are merged with application.properties taking precedence

Answer: D
Spring Boot loads both, but .properties has higher precedence over .yml.

153. Which annotation is equivalent to combining @Configuration, @EnableAutoConfiguration, and @ComponentScan?
A. @SpringBootApplication
B. @EnableSpringBoot
C. @SpringConfiguration
D. @AutoConfigApplication

Answer: A
@SpringBootApplication is a meta-annotation that combines the three.

154. If a Spring Boot application does not find a main() method with @SpringBootApplication, what happens?
A. It compiles but fails at runtime
B. It still runs using a default launcher
C. It runs in test mode
D. It throws an exception during build

Answer: A
Spring Boot needs a main() method to start the embedded server; absence leads to a runtime error.

155. Which of these statements is FALSE about Spring Boot starters?
A. They simplify dependency management
B. spring-boot-starter-data-jpa includes Hibernate by default
C. Starters always include embedded databases
D. Starters are pre-defined dependency descriptors

Answer: C

Not all starters include embedded databases; it depends on the specific starter.

156. What is the purpose of the spring-boot-devtools module?
A. Production monitoring
B. Logging only
C. Auto-restart and LiveReload during development
D. Compiling classes at runtime

Answer: C
DevTools helps in development by auto-reloading the app on changes.

157. What will happen if you exclude spring-boot-starter-web from your Spring Boot application?
A. Only REST APIs will stop working
B. The embedded server wonâ€™t start
C. Spring Boot will throw an exception
D. It has no impact

Answer: B
The embedded server like Tomcat/Jetty is part of the web starter. Excluding it disables web capabilities.

158. Which embedded server is used by default in Spring Boot if no other is specified?
A. Jetty
B. Tomcat
C. Undertow
D. Netty

Answer: B
Tomcat is the default embedded server unless overridden in the dependencies.

159. What is the effect of placing @SpringBootApplication(scanBasePackages = "com.example") in a class located in com.app.main?
A. It scans only com.app.main
B. It scans both com.example and com.app.main
C. It scans only com.example
D. It throws an error at startup

Answer: C
It overrides the default scan location, scanning only com.example.

160. Which command is used to create a new Spring Boot project using Spring CLI?
A. spring start new
B. spring init

C. spring create boot
D. spring new project

Answer: B
spring init is the correct Spring CLI command to create a new project.

161. In a basic Spring Boot application, what is the role of SpringApplication.run()?
A. Initializes only the Spring context
B. Starts the application and the embedded server
C. Compiles and packages the application
D. Only triggers dependency injection

Answer: B
SpringApplication.run() bootstraps the application, starts the Spring context, and embedded server if needed.

162. Which of the following statements is TRUE about the main() method in a Spring Boot application?
A. It is not required in Spring Boot
B. It must be in a class with @RestController
C. It is the entry point for the JVM
D. It should be avoided for microservices

Answer: C
main() is the standard JVM entry point, essential for running a Spring Boot app as a Java SE application.

163. What is the default behavior of SpringApplication.run() if no application.properties or application.yml is present?
A. It throws an error
B. It uses only environment variables
C. It uses sensible defaults and still runs
D. It shuts down the application

Answer: C
Spring Boot uses default configurations and can run without property files.

164. What will happen if a Spring Boot Java SE application does NOT include any web dependency (like spring-boot-starter-web)?
A. It fails at startup
B. It runs as a non-web CLI-style application
C. It automatically adds spring-boot-starter-web
D. It logs a warning and exits

Answer: B
Without web dependencies, it acts as a non-web Spring application (e.g., batch or console app).

165. Where should you place the @SpringBootApplication annotated class for component scanning to work effectively?
A. In any package
B. At the base package level
C. Inside a test class
D. In the resources directory

Answer: B
It should be in the root (or base) package to allow component scanning of subpackages.

166. What does the CommandLineRunner interface help achieve in a Java SE Spring Boot app?
A. Scheduling tasks in background
B. Executing code at application startup
C. Managing REST controllers
D. Handling database transactions

Answer: B
CommandLineRunner.run() executes code right after the Spring context is initialized.

167. In a Hello World Spring Boot console app, where does the System.out.println() typically go?
A. Inside the main() method
B. Inside a @Component implementing CommandLineRunner
C. Inside a @Service method
D. Inside application.properties

Answer: B
To execute logic after startup, use CommandLineRunner or ApplicationRunner.

168. Which of the following will correctly print "Hello Spring Boot" during startup in a Java SE app?
A. A method inside a @RestController
B. Code inside a class implementing CommandLineRunner
C. Code inside application.properties
D. A bean with @Autowired

Answer: B
CommandLineRunner is designed to execute custom logic after context is loaded.

169. What will happen if there are multiple beans implementing CommandLineRunner?
A. Only one is executed at random
B. Spring throws an exception
C. All are executed in no specific order unless ordered

D. None are executed

Answer: C
All CommandLineRunner beans run; their order can be controlled with @Order.

170. How do you prevent a Spring Boot Java SE application from starting the embedded server?
A. Use spring.main.web-application-type=none
B. Exclude spring-boot-starter-web
C. Extend SpringBootServletInitializer
D. Both A and B

Answer: D
Setting web-application-type=none and/or excluding web starters prevents web environment initialization.

171. Which of the following annotations is NOT a stereotype annotation in Spring?
A. @Controller
B. @Service
C. @Bean
D. @Repository

Answer: C
@Bean is used in configuration classes to define beans, but it is not a stereotype annotation.

172. Which annotation would you use for a class that contains business logic in a layered architecture?
A. @Repository
B. @Service
C. @Component
D. @Controller

Answer: B
@Service is specifically used for service/business logic layer components.

173. Which stereotype annotation provides additional exception translation for persistence layer errors?
A. @Component
B. @Repository
C. @Service
D. @Persistence

Answer: B
@Repository enables automatic exception translation for database exceptions into Springâ€™s DataAccessException.

174. All stereotype annotations are meta-annotated with which core annotation?
A. @Autowired
B. @SpringBootApplication
C. @Component
D. @Bean

Answer: C
All stereotype annotations like @Service, @Repository, and @Controller are themselves annotated with @Component.

175. If you annotate a class with both @Component and @Service, what will happen?
A. Spring will throw a conflict error
B. Both annotations cancel each other out
C. Spring will register it as a component once
D. The class will not be detected by component scanning

Answer: C
The class will be registered only once; having multiple stereotypes is redundant but harmless.

176. Which annotation is most appropriate for a class that handles HTTP requests in a Spring MVC application?
A. @Repository
B. @Service
C. @Component
D. @Controller

Answer: D
@Controller is used to handle HTTP requests and map them to handler methods.

177. What is the main reason to use @Service or @Repository over @Component, even though all are detected?
A. They consume less memory
B. They provide specific behavior and clarity in design
C. They require fewer dependencies
D. They automatically log method calls

Answer: B
These annotations add semantic meaning and in some cases, like @Repository, additional behavior (like exception translation).

178. Which stereotype annotation is typically used with RESTful web services?
A. @Controller
B. @RestController
C. @Service

D. @WebService

Answer: B
@RestController is a composed annotation: @Controller + @ResponseBody.

179. Can a class annotated with @Component be injected into a class that requires a @Service bean?
A. Yes, if they share the same bean ID
B. No, Spring checks annotation type
C. Yes, if the type matches
D. Only with manual wiring

Answer: C
Spring uses type-based injection primarily. The stereotype is for classification/documentation.

180. Which of the following statements is TRUE regarding stereotype annotations?
A. They are optional; Spring works without them
B. They help in automatic component scanning and bean registration
C. They are only used in Spring Boot
D. They disable manual bean registration

Answer: B
Stereotype annotations help Spring identify and automatically register classes as beans during classpath scanning.

181. What is the primary purpose of an ORM framework?
A. To optimize SQL queries
B. To allow objects to directly interact with databases without SQL
C. To manage UI rendering
D. To create database backups

Answer: B
ORM abstracts the database interaction by allowing developers to work with objects instead of writing raw SQL.

182. In ORM, what does the term â€œentityâ€ typically refer to?
A. A database column
B. A SQL query
C. A Java object mapped to a database table
D. A JSON object used in REST APIs

Answer: C
An entity is a Java class that represents a table in the database.

183. Which of the following annotations is used to mark a class as an entity in JPA?
A. @Persistent

B. @MappedClass
C. @Table
D. @Entity

Answer: D
@Entity is the core JPA annotation to denote a persistent class.

184. What will happen if two entity classes are mapped to the same table?
A. The one with a lower ID is preferred
B. JPA will throw an error at runtime
C. It is allowed, but can cause data integrity issues
D. JPA merges both entities automatically

Answer: C
While technically allowed, mapping multiple entities to the same table can lead to design and data issues.

185. Which of the following is TRUE about the @Id annotation in JPA?
A. It marks a class as a primary key generator
B. It is optional in entity classes
C. It defines the primary key field of the entity
D. It is used only for foreign key mapping

Answer: C
@Id is mandatory for marking the primary key field in an entity.

186. In ORM, what is the effect of lazy loading?
A. All related entities are loaded immediately
B. The main entity is not loaded until explicitly asked
C. Related entities are loaded only when accessed
D. Database transactions are skipped

Answer: C
Lazy loading defers the retrieval of associated entities until they are accessed.

187. Which of the following is used in JPA to specify the column name for a field if it differs from the variable name?
A. @Column(name = "column_name")
B. @Field("column_name")
C. @Table(name = "column_name")
D. @Join(name = "column_name")

Answer: A
@Column(name = "db_column") lets you customize the mapping of a field to a specific database column.

188. What is the purpose of the @GeneratedValue annotation in JPA?
A. It sets a default value

B. It marks a column as optional
C. It auto-generates the primary key value
D. It creates a foreign key constraint

Answer: C
@GeneratedValue tells JPA to auto-generate the value of the primary key (e.g., using sequences or identity columns).

189. What is the role of the persistence context in ORM frameworks like JPA?
A. It stores table metadata
B. It caches entities and tracks their state
C. It generates DDL scripts
D. It converts SQL to HQL

Answer: B
The persistence context is like a first-level cache where JPA tracks and manages entity states (new, managed, detached, removed).

190. Which of the following relationships is correctly represented in ORM using @ManyToOne?
A. One customer has many orders
B. One order has many items
C. Many orders belong to one customer
D. One item belongs to many orders

Answer: C
@ManyToOne means many entities refer to the same parent entity â€” e.g., many orders belong to one customer.

191. What happens if an entity class in JPA does not define an @Id field?
A. It uses a default primary key
B. JPA assigns a UUID automatically
C. It results in a runtime exception
D. It compiles and runs normally

Answer: C
Every JPA entity must have an @Id; otherwise, it throws a runtime error during persistence.

192. Which of the following annotations allows JPA to auto-generate primary keys using a database identity column?
A. @GeneratedValue(strategy = GenerationType.SEQUENCE)
B. @GeneratedValue(strategy = GenerationType.AUTO)
C. @GeneratedValue(strategy = GenerationType.IDENTITY)
D. @Id

Answer: C
GenerationType.IDENTITY uses the databaseâ€™s auto-increment/identity feature.

193. Which JPA annotation is used to define a custom table name different from the entity class name?
A. @Entity(name = "custom_table")
B. @Column(name = "custom_table")
C. @Table(name = "custom_table")
D. @JoinTable(name = "custom_table")

Answer: C
@Table(name = "...") changes the table name that the entity maps to.

194. Which of the following is NOT a valid JPA relationship annotation?
A. @OneToOne
B. @ManyToOne
C. @JoinEntity
D. @OneToMany

Answer: C
There is no @JoinEntity in JPA; the correct annotation is @JoinTable for mapping join tables.

195. Which of the following statements is TRUE about @ManyToMany relationships in JPA?
A. It does not require a join table
B. It always requires a mappedBy attribute
C. It requires an intermediate join table
D. It cannot be bidirectional

Answer: C
@ManyToMany relationships require a join table to map the association between two entities.

196. What does the mappedBy attribute indicate in bidirectional relationships?
A. It specifies the table column
B. It identifies the owner of the relationship
C. It names the mapped entity
D. It initializes the foreign key

Answer: B
mappedBy tells JPA which side owns the relationship and prevents redundant join tables.

197. What is the default fetching strategy for @ManyToOne and @OneToOne associations in JPA?
A. Lazy for both
B. Eager for both
C. Eager for @ManyToOne, Lazy for @OneToOne
D. Eager for @ManyToOne, Eager for @OneToOne

Answer: D
By default, both @ManyToOne and @OneToOne are eagerly loaded unless specified otherwise.

198. What is the purpose of the @Transient annotation in JPA?
A. Makes a field persistent but encrypted
B. Persists a field as null
C. Excludes a field from persistence
D. Stores a field in memory and database

Answer: C
@Transient marks a field that should not be persisted to the database.

199. What does the EntityManager do in JPA?
A. It manages tables
B. It provides direct SQL access
C. It manages entity lifecycle and persistence operations
D. It creates entity classes at runtime

Answer: C
EntityManager is responsible for operations like persist, merge, remove, and querying entities.

200. Which method is used by EntityManager to retrieve an entity by its primary key?
A. findById()
B. get()
C. retrieve()
D. find()

Answer: D
entityManager.find(Class<T>, primaryKey) retrieves an entity using its primary key.

201. What is the correct annotation to customize the database column name for a field in an entity?
A. @Column(name = "column_name")
B. @Field(name = "column_name")
C. @JoinColumn(name = "column_name")
D. @Property(name = "column_name")

Answer: A
@Column is used to map a Java field to a specific column name in the database.

202. Which JPA annotation marks a class as a database-mapped entity?
A. @MappedSuperclass
B. @Entity

C. @Table
D. @Persistence

Answer: B
@Entity marks a class as a JPA entity that will be mapped to a database table.

203. What does the @JoinColumn annotation specify in a JPA relationship?
A. The name of the table used in a join
B. The foreign key column name
C. The table index
D. The primary key for the entity

Answer: B
@JoinColumn specifies the foreign key column for an entity relationship.

204. Which annotation is used to map a @ManyToMany relationship with a custom join table?
A. @JoinEntity
B. @ManyToMany
C. @JoinTable
D. @MappedBy

Answer: C
@JoinTable is used with @ManyToMany to define the intermediate join table and its join columns.

205. Which annotation tells JPA to not persist a specific field of an entity to the database?
A. @Ignore
B. @Optional
C. @Skip
D. @Transient

Answer: D
@Transient tells JPA to exclude the field from persistence.

206. What does the @Temporal annotation do?
A. Converts temporal fields to strings
B. Specifies the date format for serialization
C. Defines how a java.util.Date or Calendar should be stored (DATE, TIME, TIMESTAMP)
D. Marks a field as required

Answer: C
@Temporal specifies the temporal precision (DATE, TIME, or TIMESTAMP) for date/time fields.

207. Which JPA annotation allows inheritance mapping in entities?

A. @Inheritance
B. @Entity
C. @Embedded
D. @SuperClass

Answer: A
@Inheritance defines the strategy for mapping an inheritance hierarchy to database tables.

208. What is the purpose of the @MappedSuperclass annotation?
A. Marks a class as non-entity
B. Allows fields in a superclass to be mapped to database columns in a subclass entity
C. Prevents subclass from overriding fields
D. Binds a superclass as an embedded object

Answer: B
@MappedSuperclass lets non-entity superclasses provide persistent fields to their entity subclasses.

209. What happens if a collection field in an entity is annotated with @ElementCollection?
A. It stores only primitive values
B. It maps a collection of simple types or embeddable objects to a separate table
C. It is stored as a JSON field
D. It is treated like a @ManyToOne relationship

Answer: B
@ElementCollection is used to store collections of value types (like Strings or Embeddables) in a separate table.

210. Which of the following is TRUE about @Embedded and @Embeddable in JPA?
A. @Embedded creates a foreign key
B. @Embeddable marks a class that can be embedded in multiple entities
C. @Embedded makes the class act like an entity
D. Both are used to define many-to-many relationships

Answer: B
@Embeddable marks a class whose fields are mapped into the owning entity's table.

211. What is the primary benefit of using Spring Data?
A. It provides automatic logging of queries
B. It eliminates the need for SQL joins
C. It reduces boilerplate code in data access layers
D. It replaces the need for a database

Answer: C

Spring Data minimizes boilerplate code by providing repository interfaces and auto-implemented methods.

212. Which interface must be extended to get basic CRUD operations in Spring Data JPA?
A. JpaManager
B. CrudOperations
C. CrudRepository<T, ID>
D. EntityManager

Answer: C
CrudRepository<T, ID> provides built-in methods for basic CRUD operations.

213. What is the role of @Repository in Spring Data?
A. It connects entities with service classes
B. It marks interfaces for dynamic proxy creation and exception translation
C. It maps a table to a class
D. It creates REST endpoints

Answer: B
@Repository is used to mark data access components and enable Spring Data features like proxying and exception translation.

214. Which of the following is NOT a Spring Data repository interface?
A. CrudRepository
B. PagingAndSortingRepository
C. JpaRepository
D. JdbcTemplate

Answer: D
JdbcTemplate is part of Spring JDBC, not Spring Data repository abstraction.

215. What does Spring Data use to implement repository interfaces at runtime?
A. APT (Annotation Processing Tool)
B. Java Reflection and Proxies
C. Classloaders
D. Static compilation

Answer: B
Spring Data uses runtime proxies and reflection to implement repository methods dynamically.

216. Which annotation enables JPA repositories in a Spring Boot application?
A. @EnableJpa

B. @EnableJpaRepositories
C. @RepositoryScan
D. @EnableRepositories

Answer: B
@EnableJpaRepositories tells Spring to scan and create implementations for repository interfaces.

217. What does the method name findByUsernameAndStatus in a Spring Data repository imply?
A. It returns a list of all users
B. It requires a custom SQL query
C. It uses method name conventions to generate a query by username and status
D. It performs a native SQL join

Answer: C
Spring Data parses method names to create queries automatically, like findByUsernameAndStatus.

218. Which method is used to check the existence of an entity by ID in CrudRepository?
A. exists()
B. isPresent()
C. existsById(ID id)
D. checkIfExists(ID id)

Answer: C
existsById(ID id) checks whether an entity exists with the given primary key.

219. What does the @Query annotation allow you to do in Spring Data repositories?
A. Log query performance
B. Define a native SQL or JPQL query manually
C. Automatically fetch all entities
D. Auto-generate schema

Answer: B
@Query lets you write custom JPQL or native SQL when method names are insufficient.

220. Which repository interface provides methods like findAll(Pageable pageable) and findAll(Sort sort)?
A. CrudRepository
B. JpaRepository
C. PagingAndSortingRepository
D. QueryRepository

Answer: C

PagingAndSortingRepository extends CrudRepository and adds pagination and sorting methods.

221. What is the main purpose of the Spring Data Repository abstraction?
A. To automatically configure the database schema
B. To eliminate the need for SQL altogether
C. To simplify data access by creating implementations at runtime
D. To define REST controllers for data

Answer: C
Spring Data creates repository implementations at runtime based on interface definitions.

222. In Spring Data architecture, what is the role of Repository interface?
A. It defines entity relationships
B. It provides advanced transaction management
C. It is a marker interface that enables auto-proxying of data access
D. It handles pagination logic

Answer: C
Repository is a marker interface. All repository interfaces extend it to enable dynamic proxy creation.

223. Which of the following is the correct hierarchy in Spring Data JPA repository architecture (from base to most feature-rich)?
A. Repository â†’ JpaRepository â†’ PagingAndSortingRepository
B. Repository â†’ CrudRepository â†’ JpaRepository
C. JpaRepository â†’ Repository â†’ CrudRepository
D. Repository â†’ JpaRepository â†’ CrudRepository

Answer: B
The correct hierarchy is:
Repository â†’ CrudRepository â†’ PagingAndSortingRepository â†’ JpaRepository

224. What is the role of SimpleJpaRepository in Spring Data JPA?
A. It provides connection pooling
B. It is a proxy factory
C. It is the default runtime implementation of repository interfaces
D. It maps entities to DTOs

Answer: C
SimpleJpaRepository is the default implementation class that Spring Data generates at runtime.

225. Which component is responsible for translating method names into queries in Spring Data?
A. Spring Boot Autoconfiguration

B. QueryDerivationEngine
C. JpaEntityManager
D. RepositoryQueryFactory

Answer: B
The Query Derivation Engine in Spring Data parses method names and translates them into queries.

226. What does the Spring Data RepositoryFactoryBean do?
A. It loads external SQL scripts
B. It generates bean definitions for entity classes
C. It creates proxy instances of repositories
D. It initializes connection pools

Answer: C
RepositoryFactoryBean is responsible for creating proxy instances of repositories at runtime.

227. What is the role of @EnableJpaRepositories in Spring Data architecture?
A. It enables automatic creation of JPA repositories and scans for interfaces
B. It loads JPA entity classes
C. It generates schema definitions
D. It enables RESTful endpoints

Answer: A
@EnableJpaRepositories triggers scanning and creation of repository proxies for JPA.

228. What does the CrudRepository interface provide?
A. Query performance metrics
B. Only read methods
C. Basic CRUD operations like save(), findById(), delete()
D. Native query mapping only

Answer: C
CrudRepository provides basic CRUD methods that are auto-implemented.

229. Which of the following is not a core component of Spring Data architecture?
A. Repository Abstraction
B. Domain Model
C. EntityManager
D. JSP Engine

Answer: D
JSP (Java Server Pages) is unrelated to Spring Data. The other three are integral.

230. In the context of Spring Data, what is RepositoryMetadata used for internally?
A. It stores database table information
B. It provides runtime metadata about a repository interface
C. It maps SQL queries
D. It tracks method invocations

Answer: B
RepositoryMetadata is an internal Spring Data interface used to get metadata about a repository interface, such as its domain type and ID type.

231. In a Java SE application using Spring Data JPA, what is a mandatory configuration step?
A. Autowiring of controllers
B. Enabling scheduling
C. Manually setting up EntityManagerFactory and TransactionManager
D. Using @SpringBootApplication

Answer: C
In Java SE (without Spring Boot), you must manually configure EntityManagerFactory and PlatformTransactionManager.

232. Which file is typically used to configure Spring Data in a Java SE project?
A. application.properties
B. pom.xml
C. spring-context.xml or Java Config class
D. web.xml

Answer: C
In Java SE, you usually use a Spring context XML file or Java-based configuration to set up Spring beans manually.

233. Which of the following annotations is NOT typically used in a Spring Data Java SE application?
A. @EnableJpaRepositories
B. @Entity
C. @Autowired
D. @SpringBootApplication

Answer: D
@SpringBootApplication is Spring Boot-specific and not used in Java SE Spring Data apps.

234. What is the main purpose of @EnableJpaRepositories in a Java SE application?
A. Enables scheduling for JPA
B. Triggers scanning and creation of Spring Data repository proxies

C. Registers a JPA database
D. Compiles SQL files

Answer: B
It scans for repository interfaces and creates implementations using Spring proxies.

235. How do you typically create the EntityManagerFactory in a Spring Data Java SE application?
A. Use @EntityManager
B. Configure LocalContainerEntityManagerFactoryBean
C. Autowire it
D. Use JpaRepositoryFactory

Answer: B
You use LocalContainerEntityManagerFactoryBean to manually define the EntityManagerFactory bean.

236. Which transaction manager is used in Spring Data JPA for Java SE apps?
A. JpaTransactionManager
B. HibernateTransactionManager
C. JdbcTransactionManager
D. SessionTransactionManager

Answer: A
JpaTransactionManager manages JPA transactions.

237. Which dependency is essential for using Spring Data JPA in a Java SE application?
A. spring-boot-starter-data-jpa
B. spring-jdbc
C. spring-data-jpa
D. spring-web

Answer: C
spring-data-jpa is the core dependency; Spring Boot starter is not needed in Java SE.

238. What is the correct way to define a repository interface in a Java SE Spring Data app?
A. Extend JpaRepository and annotate with @Repository
B. Annotate the interface with @Entity
C. Implement all CRUD methods manually
D. Extend EntityManager

Answer: A
Define a repository by extending JpaRepository and optionally marking it with @Repository.

239. In a Java SE context, which configuration class or XML must register the JPA DataSource?
A. JpaRepository
B. spring-data-config.xml or @Configuration class
C. JpaEntity
D. application.yaml

Answer: B
You must register the DataSource in either a Java config class or XML config.

240. What happens if you do not configure a transaction manager in Spring Data Java SE?
A. Spring throws a compile-time error
B. It defaults to JDBC
C. Repository operations may fail with No transaction or rollback issues
D. It will use Hibernate's default commit

Answer: C
Without a transaction manager, data operations may fail due to the absence of transactional context.

241. What does the @Transactional annotation primarily do in Spring?
A. Logs database activity
B. Declares that a method should run within a transactional context
C. Locks the database row
D. Maps an entity to a table

Answer: B
@Transactional ensures that a method runs inside a transactional context â€” commits on success, rolls back on failure.

242. Which of the following method visibility levels will NOT be intercepted by @Transactional by default?
A. public
B. protected
C. private
D. All of the above

Answer: C
Private methods are not proxied, so @Transactional has no effect on them by default.

243. What will happen if a checked exception (e.g., IOException) is thrown inside a @Transactional method by default?
A. The transaction rolls back
B. The transaction commits anyway
C. The method is retried
D. A compile-time error occurs

Answer: B
By default, only unchecked exceptions (RuntimeException and Error) trigger rollback.

244. How can you ensure that a method rolls back for a specific checked exception?
A. Use @Rollback
B. Set @Transactional(rollbackFor = IOException.class)
C. Catch the exception manually
D. Set propagation = Propagation.NEVER

Answer: B
Use rollbackFor to specify checked exceptions that should trigger rollback.

245. What is the default propagation setting for @Transactional?
A. REQUIRES_NEW
B. NESTED
C. REQUIRED
D. MANDATORY

Answer: C
The default is Propagation.REQUIRED, which means use the existing transaction, or create a new one if none exists.

246. If a method marked with @Transactional(propagation = Propagation.REQUIRES_NEW) is called inside another transaction, what happens?
A. It joins the outer transaction
B. It throws an error
C. It suspends the outer transaction and creates a new one
D. It executes without any transaction

Answer: C
REQUIRES_NEW suspends the current transaction and starts a completely new one.

247. Which attribute of @Transactional defines how data visibility is handled in concurrent transactions?
A. propagation
B. readOnly
C. rollbackFor
D. isolation

Answer: D
isolation defines the transaction isolation level, which affects visibility of uncommitted data.

248. What is the effect of setting @Transactional(readOnly = true)?
A. It prevents all SELECT operations
B. It disables transaction management
C. It hints the persistence provider to optimize read operations
D. It guarantees consistent reads across tables

Answer: C
readOnly = true is a performance hint for optimizing queries; it may also restrict write operations in some providers.

249. What happens if an exception is caught and not re-thrown in a @Transactional method?
A. The transaction rolls back
B. The transaction commits
C. A compile error occurs
D. The transaction is paused

Answer: B
If the exception is caught and not re-thrown, Spring considers the method successful, so it commits the transaction.

250. Where is the @Transactional annotation typically placed for effective transaction management?
A. On repository classes only
B. On private utility methods
C. On public service layer methods
D. On controller methods

Answer: C
Service layer is the correct place for @Transactional, as Spring proxies service beans and handles business logic here.

251. What is the main responsibility of the Spring Repository (DAO) layer?
A. Handling HTTP requests
B. Performing business logic
C. Abstracting and encapsulating data access and persistence operations
D. Managing application configurations

Answer: C
The Repository/DAO layer encapsulates database interactions and data persistence logic.

252. In Spring, which annotation is typically used to mark a DAO/repository class?
A. @Service
B. @Controller
C. @Repository
D. @Component

Answer: C
@Repository marks the DAO layer and enables exception translation.

253. What additional feature does the @Repository annotation provide beyond @Component?
A. Automatic transaction management
B. Automatic exception translation of persistence exceptions to Springâ€™s DataAccessException hierarchy
C. Enables RESTful endpoints
D. Enables scheduling support

Answer: B
@Repository adds exception translation for persistence exceptions.

254. If you use Spring Data JPA repositories, is it mandatory to implement DAO methods manually?
A. Yes, always
B. No, Spring Data generates implementations automatically
C. Only for complex queries
D. Only when using XML configuration

Answer: B
Spring Data JPA auto-generates implementations for repository interfaces.

255. Which interface is commonly extended by DAO interfaces to gain CRUD functionality in Spring Data JPA?
A. EntityManager
B. JpaRepository
C. JdbcTemplate
D. BeanFactory

Answer: B
Extending JpaRepository provides CRUD and paging/sorting methods.

256. Which of these is a correct advantage of using the DAO pattern with Spring?
A. Tight coupling between business logic and persistence code
B. Improved testability and separation of concerns
C. Reduced memory usage
D. Direct SQL access without ORM

Answer: B
DAO pattern improves separation of concerns and testability.

257. What happens if you omit the @Repository annotation on a DAO class?
A. Spring will fail to inject dependencies
B. Spring will not recognize the class as a Spring bean
C. Spring wonâ€™t translate exceptions, but the class can still be a bean if annotated with @Component

D. The application will crash immediately

Answer: C
Without @Repository, exceptions wonâ€™t be translated, but @Component can still make it a bean.

258. How does Spring Data repositories interact with the database?
A. Through native JDBC only
B. By using EntityManager and JPA internally
C. By direct socket connection
D. By calling REST APIs

Answer: B
Spring Data uses JPAâ€™s EntityManager internally to interact with the database.

259. Which statement about custom repository implementations in Spring Data is TRUE?
A. You cannot customize Spring Data repositories
B. Custom methods must be implemented in a separate class and wired with the interface
C. You override the JpaRepository interface directly
D. You must disable Spring Data repositories to add custom logic

Answer: B
Custom repository methods are implemented in a separate class and linked via naming conventions.

260. When using Spring Data JPA repositories, what happens if you define a method with a name that does not follow the query derivation rules and no @Query is specified?
A. Spring Data generates a default query ignoring the method name
B. The application fails to start with an error
C. Spring Data throws an exception at runtime when the method is called
D. The method returns null silently

Answer: C
If method name cannot be parsed into a query and no @Query is present, an exception is thrown at runtime.

261. What is the primary role of the Spring Service layer?
A. To handle HTTP requests
B. To implement business logic and coordinate between controllers and repositories
C. To manage database connections
D. To perform entity-to-DTO mapping only

Answer: B
The Service layer contains business logic and acts as a bridge between controllers and data repositories.

262. Which annotation is commonly used to mark a Service class in Spring?
A. @Controller
B. @Service
C. @Repository
D. @Component

Answer: B
@Service marks the service layer bean and makes it a candidate for component scanning.

263. What is a typical reason to place @Transactional on service layer methods rather than on the repository layer?
A. Repositories cannot handle transactions
B. Service layer methods often define business transactions spanning multiple repositories
C. Transactions at repository level cause conflicts
D. Transaction management is not possible on repositories

Answer: B
Transactions usually span multiple DAO calls, so managing them at the service layer maintains business consistency.

264. What is the effect of omitting @Service annotation on a service class?
A. Spring will not manage the class as a bean by default
B. The service will still work without any issues
C. Transactions wonâ€™t work on that service
D. Spring will automatically detect it as a bean anyway

Answer: A
Without @Service (or another stereotype), the class wonâ€™t be auto-detected as a Spring bean unless explicitly declared.

265. How do you inject a repository bean into a Spring service?
A. Using @Component
B. Using @Autowired or constructor injection
C. Using new keyword inside the service
D. Using @Transactional

Answer: B
Dependency injection via @Autowired or constructor injection is the standard way.

266. Which design principle does the Service layer typically promote?
A. Tight coupling
B. Separation of concerns and single responsibility
C. Monolithic design
D. Direct database calls in controllers

Answer: B
The Service layer promotes separation of concerns by encapsulating business logic.

267. If a service method calls another method within the same class annotated with @Transactional, what happens to the transactional behavior of the called method?
A. It will start a new transaction
B. Transactional behavior is ignored because of proxy limitations (self-invocation problem)
C. It will always rollback
D. It throws an exception

Answer: B
Due to Spring proxy-based AOP, self-invocation bypasses the proxy, so transactional annotations are ignored on internal method calls.

268. Whatâ€™s the recommended way to handle exceptions in the service layer?
A. Catch all exceptions and log, then swallow silently
B. Throw checked exceptions to the controller
C. Translate exceptions into meaningful business exceptions or use Springâ€™s DataAccessException hierarchy
D. Ignore exceptions and rely on controller advice

Answer: C
Services should translate exceptions or throw meaningful exceptions to the upper layers.

269. Which of the following is a good practice regarding transaction scope in the service layer?
A. Keep transactions as short as possible
B. Keep transactions open for the whole request lifecycle
C. Never use transactions in service layer
D. Use transactions only in controllers

Answer: A
Transactions should be kept as short as possible to reduce locking and improve concurrency.

270. Which of the following is true about Service layer testing?
A. Only unit tests are sufficient; integration tests are unnecessary
B. Service layer can be tested independently by mocking repository dependencies
C. Service layer tests require a live database
D. Service layer cannot be tested separately

Answer: B

Service layer is testable in isolation by mocking dependencies like repositories.

271. Why is the service layer the preferred place to define transaction boundaries in Spring?
A. Because repositories donâ€™t support transactions
B. To group multiple repository calls into a single business transaction
C. Controllers cannot be transactional
D. Transactions are only supported in the service package

Answer: B
The service layer defines business transactions that may span multiple DAO/repository calls.

272. What does Spring use to implement transactions declared with @Transactional?
A. XML configuration only
B. Aspect-Oriented Programming (AOP) proxies
C. Manual transaction management inside methods
D. Java Reflection APIs

Answer: B
Spring uses AOP proxies to intercept method calls and manage transactions.

273. If a Spring-managed service method annotated with @Transactional calls another method in the same class annotated with @Transactional(propagation = REQUIRES_NEW), what happens?
A. A new transaction starts for the inner method
B. The inner method executes without a transaction
C. The new transaction propagation is ignored because of self-invocation
D. The outer transaction is suspended and the inner method runs non-transactionally

Answer: C
Self-invocation bypasses proxies, so the REQUIRES_NEW is ignored, and the inner method runs in the outer transaction.

274. Which propagation behavior allows a method to run without a transaction and suspends any existing one?
A. REQUIRED
B. REQUIRES_NEW
C. SUPPORTS
D. NOT_SUPPORTED

Answer: D
NOT_SUPPORTED suspends the current transaction and runs non-transactionally.

275. What is the default rollback behavior of Springâ€™s @Transactional?
A. Rollback on all exceptions
B. Rollback only on unchecked exceptions (RuntimeException and Error)
C. No rollback by default
D. Rollback only on checked exceptions

Answer: B
By default, rollback happens only on unchecked exceptions.

276. If a service method catches an exception and does not rethrow it, what happens to the transaction?
A. It will rollback
B. It will commit because the exception is handled inside the transaction boundary
C. It throws a runtime exception
D. The transaction is suspended

Answer: B
Since the exception is caught and handled, Spring sees the transaction as successful and commits.

277. How can you force a transaction rollback for a checked exception?
A. Add @Transactional(noRollbackFor = Exception.class)
B. Set rollbackFor attribute in @Transactional with the exception class
C. Catch and ignore the exception
D. Wrap the checked exception in a runtime exception

Answer: B
Use rollbackFor to specify checked exceptions that should trigger rollback.

278. What is a recommended way to handle transactions in asynchronous service methods?
A. Annotate async methods with @Transactional as usual
B. Do not use @Transactional on async methods as transaction context wonâ€™t propagate
C. Use manual transaction management only
D. Use @Transactional(propagation = MANDATORY) on async methods

Answer: B
Transactions donâ€™t propagate to async threads by default, so @Transactional wonâ€™t work as expected.

279. Which of these statements about Springâ€™s transaction proxies is TRUE?
A. Transaction proxies work on final methods and classes by default
B. Transaction proxies are created using CGLIB by default for interfaces
C. Transaction proxies only intercept public methods called from outside the bean
D. Transaction proxies intercept private method calls

Answer: C
Spring proxies intercept only public methods called from outside the bean (proxy).

280. How can you make sure a transaction rollback happens even if a checked exception occurs and the exception is rethrown?
A. Use @Transactional(rollbackFor = Exception.class)
B. Catch the exception and throw a runtime exception
C. Use @Transactional(noRollbackFor = Exception.class)
D. Wrap the exception inside a DataAccessException

Answer: A
Specifying rollbackFor with checked exceptions ensures rollback when those exceptions are thrown.

281. Which JPA @Id generation strategy uses the databaseâ€™s identity column feature?
A. GenerationType.AUTO
B. GenerationType.IDENTITY
C. GenerationType.SEQUENCE
D. GenerationType.TABLE

Answer: B
IDENTITY uses the databaseâ€™s auto-increment or identity column.

282. What is the behavior of GenerationType.SEQUENCE?
A. It uses an auto-increment column
B. It uses a database sequence object to generate primary key values
C. It stores generated values in a separate table
D. It lets the persistence provider decide automatically

Answer: B
SEQUENCE relies on a database sequence to generate unique IDs.

283. Which GenerationType allows the JPA provider to pick an appropriate strategy based on the database dialect?
A. IDENTITY
B. SEQUENCE
C. TABLE
D. AUTO

Answer: D
AUTO lets the provider choose the best strategy based on the underlying DB.

284. What is a drawback of using GenerationType.TABLE?
A. It requires a special database sequence
B. It can cause performance bottlenecks due to table locking

C. It only works with Oracle databases
D. It cannot be used with composite keys

Answer: B
TABLE generation uses a dedicated table to generate IDs, which can cause contention and locking issues.

285. When using GenerationType.IDENTITY, what is a common limitation?
A. ID values are assigned after the insert operation, so the entity must be immediately inserted
B. It requires a separate sequence table
C. It cannot be used with MySQL
D. It automatically batches inserts for performance

Answer: A
Because IDENTITY uses DB auto-increment, the insert happens immediately to get the generated ID.

286. Which annotation is used to customize the sequence name for GenerationType.SEQUENCE?
A. @SequenceGenerator
B. @TableGenerator
C. @GeneratedValue
D. @IdGenerator

Answer: A
@SequenceGenerator defines sequence details like name and allocation size.

287. What does the allocationSize attribute in @SequenceGenerator control?
A. The length of the sequence name
B. How many sequence values to preallocate in memory for optimization
C. The maximum value the sequence can generate
D. The minimum value the sequence starts with

Answer: B
allocationSize defines how many sequence numbers are fetched at once to reduce DB calls.

288. How is GenerationType.TABLE typically configured?
A. Using @SequenceGenerator
B. Using @TableGenerator which specifies a separate table to hold ID values
C. Using database auto-increment columns
D. Using a UUID generator

Answer: B
@TableGenerator defines the table and columns used to generate unique IDs.

289. Which generation strategy should be avoided if your database does NOT support sequences?
A. TABLE
B. IDENTITY
C. SEQUENCE
D. AUTO

Answer: C
SEQUENCE wonâ€™t work if the database lacks support for sequences (e.g., older MySQL versions).

290. When using GenerationType.AUTO, what strategy does Hibernate typically choose on PostgreSQL databases?
A. IDENTITY
B. SEQUENCE
C. TABLE
D. AUTO_INCREMENT

Answer: B
Hibernate usually picks SEQUENCE on PostgreSQL because it supports sequences efficiently.

291. What does Spring Data do when you define a repository method like findByFirstName(String firstName)?
A. Throws an error because the query is not defined
B. Automatically derives a query based on the method name
C. Requires a manual @Query annotation to work
D. Returns all records regardless of the parameter

Answer: B
Spring Data derives the query from the method name automatically.

292. Which keyword is used in Spring Data query method names to combine conditions with logical OR?
A. And
B. Or
C. Between
D. IsNot

Answer: B
The keyword Or is used to combine conditions with logical OR.

293. Which of these method names will cause Spring Data to throw an exception at startup?
A. findByAgeGreaterThan(int age)
B. findByNameStartingWith(String prefix)
C. findByEmailLike(String email)
D. findByUnknownProperty(String value)

Answer: D
Spring Data cannot parse UnknownProperty if it doesnâ€™t match an entity field.

294. What is the effect of defining a method like findTop3ByOrderBySalaryDesc() in a repository?
A. Returns top 3 records sorted by salary ascending
B. Returns top 3 records sorted by salary descending
C. Returns all records with a limit of 3 but no sorting
D. Throws an exception because of invalid syntax

Answer: B
Returns the top 3 records sorted by salary descending.

295. How do you indicate a LIKE query with a method name in Spring Data?
A. Using Like keyword, e.g., findByNameLike(String name)
B. Using Contains keyword, e.g., findByNameContains(String name)
C. Both A and B are valid but behave slightly differently
D. You cannot perform LIKE queries without @Query

Answer: C
Both Like and Contains can be used; Like requires wildcards, Contains implies %value%.

296. What is the default behavior if a query method returns a List<T> but the query finds no results?
A. Throws an exception
B. Returns null
C. Returns an empty list
D. Returns a list with a null element

Answer: C
Returns an empty list if no matching entities found.

297. How can you define a query method to find entities created after a certain date?
A. findByCreatedDateAfter(Date date)
B. findAfterCreatedDate(Date date)
C. findCreatedDateAfter(Date date)
D. findCreatedAfter(Date date)

Answer: A
Correct syntax is findByCreatedDateAfter.

298. What will happen if a query method is defined as Optional<User> findByEmail(String email), but multiple users share the same email?
A. Returns the first user found
B. Throws an exception due to non-unique results

C. Returns null
D. Returns all matching users wrapped in Optional

Answer: B
Spring Data expects unique result for Optional<T>, else throws IncorrectResultSizeDataAccessException.

299. What keyword is used to ignore case in query method names?
A. IgnoreCase
B. CaseInsensitive
C. NoCase
D. Lowercase

Answer: A
Use IgnoreCase to make the query case-insensitive.

300. Can Spring Data query methods use multiple parameters for dynamic filtering?
A. Yes, parameters must match the property names in order
B. No, only one parameter allowed per method
C. Yes, but only if parameters are annotated with @Param
D. No, for multiple filters use JPQL queries only

Answer: A
Multiple parameters can be used, matched in method name order.

301.Which annotation is used to extract a path variable from the URL in a Spring controller method?

a) @RequestParam

b) @PathVariable

c) @RequestBody

d) @ResponseBody

302.What will be the default content type returned by a method in @RestController if no content type is specified?

a) text/html

b) application/json

c) application/xml

d) text/plain

303. **Which of the following annotations is NOT typically used to map HTTP methods in Spring?**

a) @GetMapping

b) @PostMapping

c) @DeleteMapping

d) @ViewMapping

304.In Spring MVC, which component resolves the view name returned by a @Controller method to the actual view?

a) DispatcherServlet

b) ViewResolver

c) HandlerMapping

d) HandlerInterceptor

305. To enable Cross-Origin Resource Sharing (CORS) on a Spring REST API method, which annotation can be used?

a) @CrossOrigin

b) @CorsAllowed

c) @RequestMapping

d) @EnableCORS

306.Which annotation would you use in a controller method parameter to capture HTTP query parameters?
a) @PathVariable
b) @RequestParam
c) @RequestBody
d) @ResponseBody

307. Which annotation tells Spring to treat the return value of a controller method as the response body instead of a view?
a) @RequestBody
b) @ResponseBody
c) @RequestParam
d) @PathVariable

308. What does the @RequestMapping annotation do?
a) Maps HTTP requests to handler methods of MVC and REST controllers
b) Specifies the database table for the entity
c) Configures bean dependencies
d) Validates input parameters

309.How do you specify that a controller method should only handle POST requests?
a) @RequestMapping(method = RequestMethod.POST)
b) @PostMapping
c) Both a and b
d) @GetMapping

310. What does the @ResponseStatus annotation do?
a) Maps URLs to methods
b) Sets the HTTP status code for a response
c) Binds request parameters
d) Injects dependencies

311. **Which protocol is typically used by RESTful web services?**
a) FTP
b) HTTP
c) SMTP
d) TCP

312. **What format is most commonly used to exchange data in REST web services?**

a) CSV

b) HTML

c) JSON

d) EXCEL

313. **Which XML-based language is used to describe a web service?**

a) SOAP

b) WSDL

c) REST

d) JSON

314. **Which HTTP status code typically means "Not Found" in REST?**

a) 200

b) 201

c) 404

d) 500

315. **Which annotation is commonly used in Spring Boot to create RESTful web services?**

a) @Controller

b) @RestController

c) @Service

d) @Component

316. What does the acronym WSDL stand for?

a) Web Service Data Language

b) Web Standard Description Language

c) Web Services Description Language

d) Web Service Deployment Language

317. **In REST, what does the term "stateless" mean?**

a) Server stores client session

b) Each request from client to server must contain all information needed to understand the request

c) Server keeps track of client state

d) Client never sends data

318. **What does the HTTP status code 201 indicate in REST?**

a) OK

b) Bad Request

c) Resource Created
d) Unauthorized

319. **What does the HTTP status code 204 mean?**
a) Resource created
b) Bad request
c) No content (successful response with no body)
d) Not found

320**.** What is the purpose of @RequestBody in Spring?
a) Bind query parameters
b) Inject model attributes
c) Bind the request JSON body to a Java object
d) Bind session variables

321. **1. What is a web service?**

a) A system that allows communication between two devices on the internet
b) A type of web page
c) A website for e-commerce
d) A database tool

322.**Which protocol is commonly used by web services to communicate?**

a) FTP
b) SMTP
c) HTTP
d) SNMP

323. **What does REST stand for?**

a) Reliable Enterprise State Transfer
b) Representational State Transfer
c) Remote Execution Service Technique
d) Remote State Transition

324. **In REST, which HTTP method is used to update a resource?**

a) GET
b) POST
c) PUT
d) DELETE

325. **In RESTful services, what does "stateless" mean?**

a) Server keeps track of all client sessions
b) Each request must contain all the information for processing
c) Server remembers client history
d) Server stores session info in cookies

326.**Which annotation in Spring Boot is used to handle REST APIs?**

a) @Controller
b) @RestController
c) @Component
d) @Service

327.**Which of the following is NOT a valid HTTP method in REST?**

a) CONNECT
b) PATCH
c) CREATE
d) DELETE

328. **What is WSDL used for in web services?**

a) Defines security settings
b) Describes the functionalities offered by a SOAP web service
c) Configures database connection
d) Enables REST communication

329. **Which of the following is a key feature of RESTful web services?**

a) Stateful
b) Platform-dependent
c) Stateless
d) Binary communication only

330. **13. Which annotation is used to map a specific URL to a method in Spring Boot?**

a) @RestController
b) @Autowired
c) @RequestMapping
d) @SpringBootApplication

333. **16. In REST, which HTTP method is idempotent?**

a) GET
b) POST
c) PUT
d) Both a and c

334. 18. Which Spring Boot class is used to handle HTTP responses with full control (status code, headers, body)?

a) ModelAndView
b) ResponseEntity
c) RestTemplate
d) HttpHeaders

335. **Which annotation in Spring Boot is used to define a global exception handler?**

a) @RestController
b) @ControllerAdvice
c) @ExceptionHandler
d) @ResponseStatus

336. **What is the role of the @ExceptionHandler annotation in Spring Boot?**

a) Marks a class as a controller
b) Declares a global variable
c) Handles specific exceptions in controller methods
d) Maps URLs to methods

337. **How can you return a custom HTTP status code when handling an exception?**

a) Using @ResponseStatus
b) Using @ExceptionHandler
c) By throwing a RuntimeException
d) With @RequestMapping

338.**Which of the following is a correct way to define a global exception handler class?**

a) Annotate with @RestController only
b) Annotate with @Component and @RequestMapping
c) Annotate with @ControllerAdvice and use @ExceptionHandler methods
d) No annotation is needed

339. **5. What does the @ResponseStatus(HttpStatus.NOT_FOUND) annotation do?**

a) Sends a redirect
b) Returns 200 OK
c) Sets HTTP status to 404
d) Returns JSON data

340. **Which class can be extended to create a centralized exception response in Spring Boot?**

a) Throwable
b) Exception
c) ResponseEntityExceptionHandler
d) HttpStatusCodeException

341.**What is typically returned by an exception handler method in REST APIs?**

a) HTML page
b) ResponseEntity with status and error body
c) JSP file
d) ModelAndView

342.**What is the purpose of BindingResult in Spring MVC?**

a) To store model attributes
b) To validate input manually
c) To access validation errors after binding
d) To display logs

343. **1. What is the purpose of @ControllerAdvice in Spring Boot?**

a) To create REST controllers
b) To provide centralized exception handling
c) To define database repositories
d) To mark classes as service

344.**Which annotation is used inside @ControllerAdvice to catch specific exceptions?**

a) @CatchException
b) @HandleError
c) @ExceptionHandler
d) @Catch

345. **You want to return 404 Not Found for a custom exception UserNotFoundException. Which is correct?**

a) Use @ResponseStatus(HttpStatus.BAD_REQUEST)
b) Use @ExceptionHandler(UserNotFoundException.class) only
c) Use @ResponseStatus(HttpStatus.NOT_FOUND) on the exception class
d) Use @RequestMapping inside the class

346. **5. Which of the following is true about ResponseEntity in exception handling?**

a) It cannot be used in exception handling
b) It only returns JSON strings
c) It can wrap both the response body and HTTP status
d) It only returns 200 OK

347.**What is the advantage of using ResponseEntityExceptionHandler class?**

a) It speeds up the API
b) It allows customizing default error handling logic
c) It improves logging
d) It avoids the need of using @RestController

348. **Which exception is automatically thrown for invalid request body data during validation?**

a) NullPointerException
b) ValidationException
c) IllegalArgumentException
d) MethodArgumentNotValidException

349.What does @ResponseStatus(HttpStatus.BAD_REQUEST) do?

a) Sends response with status 500
b) Sends response with status 404
c) Sends response with status 200
d) Sends response with status 400

350. In Spring Boot, where do you handle exceptions like HttpMessageNotReadableException?

a) In Entity classes
b) In @RestController methods
c) In a class extending ResponseEntityExceptionHandler
d) In application.properties

351. **What should you return from @ExceptionHandler to send a proper REST response?**

a) A JSP page
b) A JSON string only
c) ResponseEntity with error message and status
d) void

352. **What is the main purpose of generating consistent responses in REST APIs?**

a) To reduce code size
b) To support SOAP protocol
c) To ensure all responses follow a predictable format
d) To log errors

353.**What does a standard API response typically include?**

a) Only status code
b) Only data
c) Status, message, timestamp, and data
d) Controller class name

354. **Which data type is usually used to send structured response objects in Spring Boot?**

a) List<Object>
b) Map<String, Object>
c) Custom POJO class (ResponseDto)
d) int[]

355. **What does a good API response help with during debugging?**

a) Display more GUI options
b) Improve performance
c) Give meaningful messages and status codes
d) Reduce the size of the log file

356. **Which annotation can be used along with @ExceptionHandler to send a standard error response?**

a) @PathVariable
b) @ResponseStatus
c) @ResponseBody
d) @RequestBody

357. **What would a typical custom response class NOT include?**

a) data
b) message
c) timestamp
d) controller name

358. Which HTTP status code is best for returning a validation error?

a) 401 Unauthorized
b) 500 Internal Server Error
c) 400 Bad Request
d) 302 Found

359. **Which annotation helps format REST responses automatically as JSON?**

a) @JsonFormat
b) @RestController
c) @GetMapping
d) @Component

360. **What is the default port used by HTTP?**

a) 21
b) 443
c) 25
d) 80

361. Which HTTP method is used to retrieve data from a server?

a) POST

b) GET

c) PUT

d) DELETE

362. **What does the HTTP status code 404 mean?**

a) OK

b) Resource not found

c) Internal server error

d) Created

363. **Which HTTP method is used to update an existing resource?**

a) GET

b) POST

c) PUT

d) OPTIONS

364. **What is the function of HTTP headers?**

a) Encrypt the response

b) Provide metadata about the request or response

c) Create new HTML files

d) Start a database connection

365. **Which HTTP method is typically used to delete a resource on the server?**

a) GET

b) POST

c) PUT

d) DELETE

366. Which status code is returned for an unauthorized request?

a) 200

b) 500

c) 401

d) 403

367.**What does the HTTP status code 500 represent?**

a) OK

b) Created

c) Bad Request

d) Internal Server Error

368. What does the HTTP method OPTIONS do?

a) Submits form data
b) Retrieves headers that describe the communication options
c) Deletes a resource
d) Uploads a file

369. Which header is used to indicate the type of data being sent in an HTTP request?

a) Accept

b) Authorization

c) Content-Type

d) Host

370. **What is the purpose of the Accept header in an HTTP request?**

a) To send a cookie
b) To define response type the client expects
c) To send a file
d) To block requests

371. **Which HTTP method is not idempotent?**

a) GET
b) PUT
c) POST
d) DELETE

372. **15. Which of the following is NOT a valid HTTP method?**

a) TRACE
b) COPY
c) PATCH
d) CONNECT

373. **What does an HTTP response with status code 204 mean?**

a) OK
b) Created
c) No Content
d) Unauthorized

374. **Which part of the HTTP request contains parameters for a GET request?**

a) Request body
b) URL query string
c) Request header
d) Authorization header

375. **What is the full form of HTTP?**

a) HyperText Transmission Protocol
b) HyperText Transfer Protocol
c) HighText Transfer Protocol
d) Hyper Technical Transfer Protocol

376. **Which header is used to authorize a user in an HTTP request?**

a) Content-Type
b) Accept
c) Authorization
d) Location

377. **Which of the following status codes represents "Forbidden"?**

a) 401
b) 403
c) 404
d) 500

378. What does the Host header in an HTTP request represent?

a) Server's IP

b) Domain name of the server

c) Port number

d) Client IP

379. Which HTTP method is used to fetch only headers of a resource (not the body)?

a) GET

b) POST

c) HEAD

d) OPTIONS

380. **. In REST, which HTTP method is used for partial updates of a resource?**

a) POST
b) PUT
c) PATCH
d) DELETE

381. **Which status code is returned when a resource has been permanently moved?**

a) 301
b) 302
c) 404
d) 410

382. **Which HTTP header is used to specify allowed HTTP methods for CORS or OPTIONS?**

a) Allow
b) Accept
c) Content-Length
d) Server

383. **Which of the following is NOT a valid component of an HTTP request?**

a) Request line
b) Request body
c) HTTP trailer
d) Header fields

384. **Which method has size limitations on the amount of data sent?**
a) POST
b) GET
c) PUT
d) DELETE

Answer: b) GET
(Because data is sent via URL query string, which has length limits)

385. **When a REST request hits a Spring Boot application, what is the first component to receive it?**
a) Service Layer
b) Controller
c) Repository
d) DispatcherServlet

386. What role does the DispatcherServlet play in Spring MVC?
a) It processes business logic
b) It routes incoming HTTP requests to appropriate handlers/controllers
c) It connects to the database
d) It handles exception logging

387. **Which annotation is used to mark a class as a REST controller in Spring?**
a) @Controller
b) @RestController
c) @Service
d) @Repository

388. **In the request flow, after the controller processes the request, which layer is typically called next?**
a) Repository layer
b) Service layer
c) Configuration layer
d) Filter chain

389. **What is the primary responsibility of the service layer in Spring REST architecture?**
a) Data persistence
b) Business logic processing
c) Handling HTTP requests
d) Mapping JSON to Java objects

390. **Which Spring annotation is commonly used to mark the service layer class?**
a) @Controller
b) @Service
c) @Repository
d) @Componen

391. **How does the service layer communicate with the database in Spring?**
a) Direct SQL queries
b) Via the Repository layer
c) Using JMS
d) Through HTTP requests

392. **What does the Repository layer usually extend in Spring Data JPA?**

a) JpaRepository

b) CrudRepository

c) Both a and b

d) None of the above

393. **Which annotation is used to handle exceptions globally in Spring REST?**

a) @ControllerAdvice

b) @ExceptionHandler

c) @RestControllerAdvice

d) All of the above

394. What happens if an exception is thrown inside a controller method and there is no exception handler?

a) The application crashes

b) A default error response with HTTP 500 is returned

c) The request is ignored silently

d) The server restarts

395. **How is JSON serialization/deserialization handled in Spring REST by default?**

a) Using JAXB

b) Using Jackson library

c) Using Gson

d) Manually by developer

396. **. In the Spring REST flow, what component serializes the Java object to JSON before sending the response?**

a) Service Layer

b) HttpMessageConverter

c) Repository Layer

d) DispatcherServlet

397. **. Which component in Spring MVC can be used to intercept requests before they reach the controller?**

a) Filter

b) Interceptor

c) Both a and b

d) Service

398. **What is the difference between a Filter and an Interceptor in Spring?**

a) Filters are Servlet API components; Interceptors are Spring-specific

b) Filters execute after Interceptors

c) Interceptors cannot modify requests

d) Filters run only for REST services

399. **Which annotation is used to validate request data in Spring REST?**

a) @Valid

b) @Validated

c) @Check

d) Both a and b

400. **How does Spring REST handle content negotiation?**

a) By checking the URL path only

b) By reading the Accept header in HTTP request

c) By looking at the User-Agent header

d) It does not support content negotiation

401. **Which component is responsible for resolving views in Spring MVC?**

a) DispatcherServlet

b) ViewResolver

c) Controller

d) Service

402. **In Spring REST, what is the function of @CrossOrigin annotation?**

a) To handle HTTP errors

b) To enable Cross-Origin Resource Sharing (CORS)

c) To map request paths

d) To validate input

403. **What is content negotiation in RESTful web services?**

a) Selecting the best data format for the client from available options

b) Negotiating the price of API usage

c) Authenticating the client

d) Encrypting the data

404. **Which HTTP header indicates the format of the data being sent by the client to the server?**

a) Content-Type

b) Accept

c) Host
d) Referer

405. **In Spring REST, which component handles content negotiation?**
a) DispatcherServlet
b) HttpMessageConverter
c) ContentNegotiationManager
d) ViewResolver

406. **What is the default content type returned by Spring REST if no Accept header is specified?**
a) text/html
b) application/json
c) application/xml
d) text/plain

407. **Which method of content negotiation uses the URL path extension like /user.json or /user.xml?**
a) Header-based negotiation
b) Parameter-based negotiation
c) Path extension-based negotiation
d) Cookie-based negotiation

408. **How can content negotiation be configured in Spring Boot?**
a) Using application.properties or application.yml
b) Using Java Config with ContentNegotiationConfigurer
c) Both a and b
d) It cannot be configured

409. **Which of the following is NOT a supported media type for content negotiation in REST?**
a) application/json
b) application/xml
c) image/jpeg
d) application.exe

410. If a client requests a media type not supported by the server in the Accept header, what is the typical HTTP response code?
a) 200 OK
b) 406 Not Acceptable
c) 404 Not Found
d) 500 Internal Server Error

411. **Which interface in Spring handles the conversion between Java objects and HTTP messages for content negotiation?**
a) HttpServletRequest
b) HttpMessageConverter
c) MessageConverterAdapter
d) RequestBodyConverter

412. **Which annotation can be used to customize the content negotiation strategy globally in Spring Boot?**
a) @EnableWebMvc
b) @EnableContentNegotiation
c) No specific annotation; use configuration classes
d) @ContentNegotiation

413.True or False: Content negotiation can also consider query parameters to determine response format.
a) True
b) False

414. **Which of the following is NOT a common way to version a RESTful API?**
a) URI Versioning
b) Query Parameter Versioning
c) Header Versioning
d) Token Versioning

415. **What is a typical format of URI versioning?**
a) /v1/products
b) /products?version=1
c) /products with X-API-VERSION
d) /products.html

416. **What HTTP header is commonly used for header-based API versioning?**
a) Content-Type
b) Accept
c) X-API-VERSION
d) Authorization

417. **Which versioning approach can result in URL pollution?**
a) URI versioning
b) Header versioning

c) Content Negotiation
d) Semantic versioning

418. **In Spring Boot, which annotation allows handling multiple versions of a REST endpoint?**
a) @Version
b) @RequestMapping with params/headers
c) @RestController
d) @ModelAttribute

419. **Which versioning strategy uses Accept: application/vnd.company.app-v1+json?**
a) Header versioning
b**) Media type versioning**
c) URL versioning
d) Parameter versioning

420. **What is one major drawback of media type versioning?**
a) Breaks RESTful principles
b) Cannot be cached
c) Requires special client configuration
d) Not supported in Spring Boot

421. **Which versioning strategy keeps URLs clean but may be harder to test?**
a) Query parameter
b) URI versioning
c) Header versioning
d) JSON versioning

422. **What is the goal of API versioning?**
a) Increase response time
b) Avoid breaking existing clients
c) Reduce database queries
d) Prevent caching

423. **. Which tool in Spring helps differentiate method handlers for different versions?**
a) DispatcherServlet
b) RequestMappingHandlerMapping
c) HandlerInterceptor
d) HttpMessageConverter

424. **Which JavaScript function is commonly used to call REST APIs in React?**
a) JSON.parse
b) fetch()

c) createAPI()
d) connect()

425. **What is the purpose of useEffect() when consuming REST services in React?**
a) Manage state
b) Fetch data after component mounts
c) Initialize forms
d) Handle CSS

426. **Which library is commonly used as an alternative to fetch for API calls in React?**
a) Redux
b) Axios
c) Lodash
d) Express

427. **In Axios, what does the .then() block handle?**
a) Input form binding
b) Successful response
c) Error response
d) Component rendering

428. **How do you send JSON data in a POST request using Axios?**
a) .post(url, headers)
b) .post(url, body)
c) .post(url, {data: json})
d) .post(url, json)

429.Where should the REST API base URL ideally be stored in a React app?
a) index.html
b) localStorage
c) .env file
d) App.css

430. **Which testing framework is commonly used with Spring Boot?**
a) Mocha
b) JUnit
c) Jasmine
d) Karma

431. **What annotation is used to create a test class for Spring Data repository testing?**
a) @RepositoryTest
b) @DataJpaTest

c) @SpringBootTest
d) @JpaRepositoryTest

432. **What is the default database used in Spring Boot tests if none is specified?**
a) MySQL
b) PostgreSQL
c) H2
d) Oracle

433. **What does @DataJpaTest automatically configure?**
a) Full application context
b) Controllers and services
c) JPA repositories and in-memory DB
d) External APIs

434. **Which method is used to save data in a repository test?**
a) add()
b) push()
c) save()
d) commit()

435. **How do you retrieve all records in a Spring Data JPA repository?**
a) find()
b) listAll()
c) getAll()
d) findAll()

436. **How do you test a custom query in a repository?**
a) By mocking it
b) By calling it with @DataJpaTest
c) Through Postman
d) Using @WebMvcTest

437. **What is the advantage of using H2 in tests?**
a) Real production data
b) Persistence
c) Fast in-memory execution
d) UI-based results

438. **Which annotation rolls back the transaction after each test?**
a) @TestTransaction
b) @Rollback

c) @BeforeEach
d) @AfterAll

439. **How can you verify the record count in a repository test?**
a) assertEquals(repo.size())
b) assertThat(repo.count())
c) repo.length
d) check(repo)

440. **What annotation is used to mock a bean in Spring Boot test?**
a) @InjectMock
b) @MockBean
c) @MockService
d) @Stub

441. **Which annotation is used for testing only web layer?**
a) @SpringBootTest
b) @WebMvcTest
c) @RestControllerTest
d) @ControllerCheck

442. **In controller tests, what is used to simulate HTTP requests?**
a) TestTemplate
b) RestTemplate
c) MockMvc
d) HttpRunner

443. **What does MockMvc.perform() do?**
a) Runs the server
b) Sends a real API request
c) Simulates an HTTP request to the controller
d) Starts a full app context

444. **How do you simulate a GET request in MockMvc?**
a) get(url)
b) MockMvc.get(url)
c) MockMvcRequestBuilders.get(url)
d) mock(url)

445. **. How do you test the HTTP status code in MockMvc result?**
a) result.getCode()
b) status().isOk()

c) checkStatus()
d) assertCode()

446. **What method is used to mock service calls in controller tests?**
a) when(service.method()).thenReturn(value)
b) mock(service).get()
c) expect(service.method()).return()
d) spy(service)

447. **What is injected into test class to perform controller unit tests?**
a) TestRestController
b) WebClient
c) MockMvc
d) @MockMvcTest

448. **. Which library supports mocking in Java unit tests?**
a) JPA
b) Mockito
c) JDBC
d) JAX-Rs

449. **What assertion library is commonly used with JUnit and Mockito?**
a) Assert.js
b) Chai
c) AssertJ
d) Mocha

450. **What does integration testing validate?**
a) UI only
b) Each isolated method
c**) Interaction between multiple layers**
d) None of the above

451. **. What annotation loads full Spring context for integration testing?**
a) @ContextTest
b) @SpringBootTest
c) @WebMvcTest
d) @DataTest

452. **What is the difference between unit and integration test?**
a) Unit tests are faster
b) Integration tests test full stack

c) Unit tests test isolated units

d) All of the above

453. **Which database is preferred in integration testing for isolation?**

a) Production DB

b) H2 (in-memory)

c) Local MySQL

d) Remote PostgreSQL

454. **What annotation is used to run a test method in Spring context?**

a) @SpringContext

b) @RunWith(SpringRunner.class)

c) @RunTest

d) @ContextConfig

455. **What is the main advantage of integration testing in Spring Boot?**

a) Fast execution

b) Simulates real-world behavior

c) Avoids database usage

d) UI-based testing

456. **Can we use MockMvc for integration tests?**

a) No

b) Only for unit

c) Yes, but limited

d) Yes, especially with @SpringBootTest

457. Which test checks if REST endpoint returns correct status and body together?

a) Controller unit test

b) Repository test

c) Integration test

d) Postman test

458. **. In Spring Boot, which tool is often used to validate JSON in integration tests?**

a) jsonPath

b) XPath

c) cssSelector

d) jQuery

459. **Which annotation runs Spring Boot integration test with actual HTTP server?**

a) @WebMvcTest

b) @MockBean

c) @SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
d) @DataJpaTest

460. **Which of the following best represents query parameter versioning?**
a) /v1/products
b) /products?version=1
c) /products/version/1
d) /products/v=1

461. **What is the risk of URI versioning in REST?**
a) Breaks stateless nature
b) Increases response time
c) Pollutes endpoint URLs
d) Difficult to implement

462. **Which versioning approach allows the best support for caching?**
a) URI versioning
b) Header versioning
c) Media type versioning
d) Query parameter versioning

463. **Which of the following is NOT an HTTP standard header?**
a) Accept
b) Content-Type
c) X-API-VERSION
d) Authorization

464. **In media type versioning, where is version info stored?**
a) Path variable
b) JSON payload
c) Accept header
d) URL

465. **What is axios.get(url) equivalent to in fetch()?**
a) fetch(url, {method: 'GET'})
b) fetch.get(url)
c) get(url)
d) fetch(url)

466. **What should you use in React to show a loading state during an API call?**
a) useEffect()
b) useRef()

c) A loading state via useState()
d) useCallback()

467.**Which test type is best for @WebMvcTest?**
a) Repository logic
b) Service logic
c) Controller logic
d) Full integration

468. **How do you simulate HTTP GET call in MockMvc?**
a) performGet()
b) MockMvc.get()
c) MockMvc.perform(get("/api"))
d) execute("/api")

469. **What does when(...).thenReturn(...) do in a test?**
a) Declares new method
b) Initializes real service
c) Mocks service call
d) Logs output

470. **What does @DataJpaTest include by default?**
a) MockMvc setup
b) Controller context
c) In-memory DB with repositories
d) Full application context

471. **Which dependency is typically added to pom.xml for building REST services in Spring Boot?**
a) spring-boot-starter-web
b) spring-boot-starter-data-jpa
c) spring-boot-starter-security
d) spring-boot-starter-thymeleaf

472. **What happens if @RequestBody is missing and the method expects a body?**
a) Returns null
b) Throws HttpMessageNotReadableException
c) Ignores the payload
d) Compiles but returns empty object

473. **What does @CrossOrigin annotation do?**
a) Restricts HTTP methods

b) Enables CORS support

c) Secures REST APIs

d) Logs the request origin

474. **What will @RestController automatically include that @Controller does not?**

a) View rendering

b) JSON/XML response conversion

c) Security filter

d) Request mapping logs

478. **Which class in Spring handles converting Java objects to JSON for REST responses?**

a) ObjectMapper

b) RestHandler

c) ResponseEntityBuilder

d) WebMvcConfigurer

479. **Which annotation is used to define custom exception classes?**

a) @RestError

b) @CustomException

c) None

d) @ExceptionHandler

480. **What is returned by a method annotated with @ExceptionHandler?**

a) Error logs

b) ResponseEntity or object with error details

c) HTML view

d) Null object

481. **What does @RequestMapping allow you to define?**

a) URL path

b) HTTP method

c) Consumes/produces

d) All of the above

482. In RESTful services, which status code represents a successful DELETE operation?

a) 201 Created

b**) 204 No Content**

c) 404 Not Found

d) 200 OK

483. **When is a HttpMessageNotReadableException typically thrown?**

a) When path variable is missing
b) When JSON request is invalid
c) When service is not found
d) When database is down

484. **Which class is used for integration testing with random port?**

a) RestTemplate
b) MockRestService
c) TestRestTemplate
d) WebTestClient

485. **What is the role of @AutoConfigureMockMvc?**

a) Enables JPA in testing
b) Injects MockMvc without starting the server
c) Sets up DevTools
d) Loads production profile

486. **Which lifecycle event occurs before each integration test method?**

a) @PostConstruct
b) @BeforeTestMethod
c) @BeforeEach
d) @BeforeTestExecution

487. **Which dependency is required for TestRestTemplate to work in integration tests?**

a) spring-boot-starter-web
b) spring-boot-starter-test
c) spring-boot-devtools
d) spring-test-utils

488. **In integration tests, what is the purpose of @Sql annotation?**

a) Define query timeout
b) Auto-generate schema
c) Run SQL scripts before/after test
d) Connect to live database