2021_25_IV_Rest API_CA2_G2_Slot3

Test Summary

- No. of Sections: 2
- No. of Questions: 10
- Total Duration: 90 min

Section 1 - MCQ

Section Summary

- No. of Questions: 9
- Duration: 15 min

Additional Instructions:
None

Q1. Which of the following is NOT a valid data type in JSON?

- Date
- String
- Number
- Boolean
- Array

Q2. In a RESTful API architecture, which layer is responsible for handling HTTP requests, routing, and returning HTTP responses?

- Controller
- Service layer
- View
- Model

Q3. What is the purpose of the @Value annotation in Spring Framework?

- To inject values from application.properties files
- To define a bean in the Spring context
- To specify a default value for a method parameter
- To define a constant value in a class

Q4. What is the main purpose of the @JsonProperty annotation in Jackson library?

To specify the name of a JSON property during serialization and deserialization

To define a new JSON property in a JSON object

To specify the order of JSON properties in a JSON object

To indicate that a property should be ignored during serialization and deserialization

Q5. Which of the following statements about the @PathVariable annotation in Spring is NOT true?

The @PathVariable annotation is used to bind a part of the URI path to a method parameter.

The @PathVariable annotation is used to extract query parameters from the URL.

The @PathVariable annotation can be used with any HTTP method.

The @PathVariable annotation can be used to handle dynamic parts of the URL.

Q6. Which of the following is true about the @RequestParam annotation in Spring?

The @RequestParam annotation is required to be used in every controller method that expects query parameters in the URL.

The @RequestParam annotation is optional and can be used selectively based on the specific requirements of the controller method.

The @RequestParam annotation is only used with GET HTTP method and cannot be used with other HTTP methods.

The @RequestParam annotation can only bind primitive data types as method parameters.

Q7. In JPA, how can you use the AND operator to combine multiple conditions in a query?

Using the & symbol

Using the AND keyword

Using the && symbol

Using the AND() method

Q8. Which of the following JPA query is equivalent to the SQL query "SELECT * FROM Employee WHERE age > 25 AND department = 'HR'"?

SELECT e FROM Employee e WHERE e.age > 25 AND e.department = 'HR'

SELECT e FROM Employee e WHERE e.age > 25 && e.department = 'HR'

SELECT e FROM Employee e WHERE e.age > 25 OR department = 'HR'

SELECT e FROM Employee e WHERE e.age > 25 && department = 'HR'

Q9. What is the typical request body for a DELETE API?

Empty or null

JSON or XML representation of the resource to be deleted

Query parameters with the resource ID to be deleted

Form data with the resource ID to be deleted

Section 2 - Project

Section Summary
- No. of Questions: 1
- Duration: 75 min

Additional Instructions:
None

Q1. **GET Song Details with CRUD Operation.**

Overview:

Get the song details using CRUD operations.

Functional Requirements:
Create 4 folders inside the **WORKSPACE/springapp/src/main/java/com/example/springapp**
1. Controller
2. Model
3. Repository
4. Service

Inside the controller, create a Java file named "ApiController.java"
Inside the model, create a Java file named "Song.java"

Create 5 variables
1. songId - int
2. musicDirectorName- string
3. yearOfRelease - int
4. songName- string
5. movieName - string

as well as create getters and setters and constructors for the corresponding variables.
- Inside the Repository, create a Java file named "SongRepo.java"
- Inside the Service, create a Java file named "ApiService.java"
The project structure looks like this image

Core Platform
OpenJDK 11

API:

Question 1 (16 marks)

POST - "/" --> true/false **(8 Marks)**
GET - "/{id}" --> Song object **(4 Marks)**
GET - "/" --> List of Song object **(4 Marks)**

Question 2 (16 marks)

PUT - “/{id}” --> Song object **(8 Marks)**
DELETE - “/{id}” --> true/false **(8 Marks)**

Note:
Copy and paste it into the **application.properties** file

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost/song?createDatabaseIfNotExist=true
spring.datasource.username=root
spring.datasource.password=examly
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.show-sql= true
spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
```

API endpoint:
8080

Platform Guidelines:
To run the command use **Terminal** in the platform.

Spring Boot:
Navigate to the springapp directory => **cd springapp**
To start/run the application '**mvn spring-boot:run**'
Click on the Run Test Case button to pass all the test cases

Answer Key & Solution

Section 1 - MCQ

Q1 Date

Solution

No Solution

Q2 Controller

Solution

No Solution

Q3 To inject values from application.properties files

Solution

No Solution

Q4 To specify the name of a JSON property during serialization and deserialization

Solution

No Solution

Q5 The @PathVariable annotation is used to extract query parameters from the URL.

Solution

No Solution

Q6 The @RequestParam annotation is optional and can be used selectively based on the specific requirements of the controller method.

Solution

No Solution

Q7 Using the AND keyword

Solution

No Solution

Q8 SELECT e FROM Employee e WHERE e.age > 25 AND e.department = 'HR'

Solution

No Solution

Q9 Empty or null

Solution

No Solution

Section 2 - Project

Q1 Solution cannot be displayed