# 2021_25_IV_Rest API G2 CA2_Slot 1

**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 attribute is used with the @ManyToOne annotation to specify the target entity class?

- mappedBy
- cascade
- targetEntity
- joinColumn

Q2. Which annotation is used to specify a named query in JPA?

- @NamedQuery
- @Named
- @NamedPersonQuery
- @NamedNativeQuery

Q3. What is the purpose of the EntityManager in JPA?

- To manage entity instances during runtime
- To define the structure of the database schema
- To map entity classes to database tables
- To generate SQL queries automatically

Q4. What is the main advantage of a bidirectional relationship in JPA?

It makes it easier to query the database

It provides a more efficient way to store data in the database

It allows for more flexible and powerful object-oriented programming

None of the above

Q5. How is a One-to-Many relationship typically mapped in the database?

With a foreign key in the table of the owning entity that references the table of the other entity

With a foreign key in the table of the other entity that references the table of the owning entity

With a join table that has foreign keys to both entities' tables

With a composite key that combines the primary key of both entities

Q6. Which of the following is a valid JPQL function?

LENGTH

MAX

SUM

BOTH LENGTH AND SUM

Q7. Which JPA annotation is used to create a one to one relationship between two entities?

OneToOne

oneToone

One-To-One

1-To-1

Q8. Which attribute is used with the @OneToMany annotation to specify the name of the foreign key column in the child entity's table?

mappedBy

cascade

- targetEntity
- joinColumn

Q9. Which of the following is not a valid JPQL query?

- SELECT e FROM Employee e
- SELECT e.name FROM Employee e
- SELECT name FROM Employee
- SELECT COUNT(e) FROM Employee e

## Section 2 - Coding

**Section Summary**

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

**Additional Instructions:**

None

Q1. **GET Bike Details with JPQL Query**

**Overview:**
Get the bike details using JPQL Query 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 “Bike.java”
Your table name should be **bike.**

Create 6 variables in Bike.java

- 1. bikeId - int
- 2. regNumber- string
- 3. ownerName- string
- 4. year- int
- 5. bikeName- string
- 6. modelName - string

as well as create getters and setters and constructors for the corresponding variables.

- Inside the Repository, create a Java file named "BikeRepo.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)**
GET - "/year/{year}" --> List of Bike object
GET - "/year/{year}/bikeName/{bikeName}" --> List of Bike object

**Question 2 (16 Marks)**
POST - "/" --> true/false
GET - “/bikeName/{bikeName}” --> Bike object

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

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost/bike?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 targetEntity

**Solution**

No Solution

Q2 @NamedQuery

**Solution**

No Solution

Q3 To manage entity instances during runtime

**Solution**

No Solution

Q4 It allows for more flexible and powerful object-oriented programming

**Solution**

No Solution

Q5 With a foreign key in the table of the owning entity that references the table of the other entity

**Solution**

No Solution

Q6 BOTH LENGTH AND SUM

**Solution**

No Solution

Q7 OneToOne

**Solution**

No Solution

Q8 joinColumn

**Solution**

No Solution

Q9

SELECT name FROM Employee

**Solution**

No Solution

## Section 2 - Coding

Q1

Solution cannot be displayed