2021_25_IV_Rest API G2 CA2_Slot 2

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 JPQL query?

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

Q2. Which method is used to execute a named query in JPA?

- executeQuery()
- getResultList()
- getSingleResult()
- All of the above

Q3. Which of the following is true about JPQL?

- It supports SQL-like syntax
- It is only used for selecting data
- It can be used with any database management system
- It doesn't support joins

Q4. What is a One-to-Many relationship in JPA?

A relationship between two entities where one entity can have multiple instances of the other entity

A relationship between two entities where one entity can have only one instance of the other entity

A relationship between three entities where one entity can have multiple instances of the other two entities

A relationship between two entities where both entities can have multiple instances of each other

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. In a One-to-Many relationship, which entity is typically the owning entity?

The entity with the @ManyToOne annotation

The entity with the @OneToMany annotation

Both entities are considered owning entities

None of the entities are considered owning entities

Q7. How do you define a bidirectional relationship in JPA?

By using the @ManyToOne annotation in one entity and the @OneToMany annotation in the other entity.

By using the @OneToOne annotation in both entities

By using the @ManyToMany annotation in both entities.

None of the above

Q8. Which of the following is a valid JPQL function?

LENGTH

MAX

SUM

BOTH LENGTH AND SUM

Q9. Which JPA annotation is used to create a one to one relationship between two entities?

OneToOne

oneToone

One-To-One

1-To-1

Section 2 - Coding

Section Summary

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

Additional Instructions:

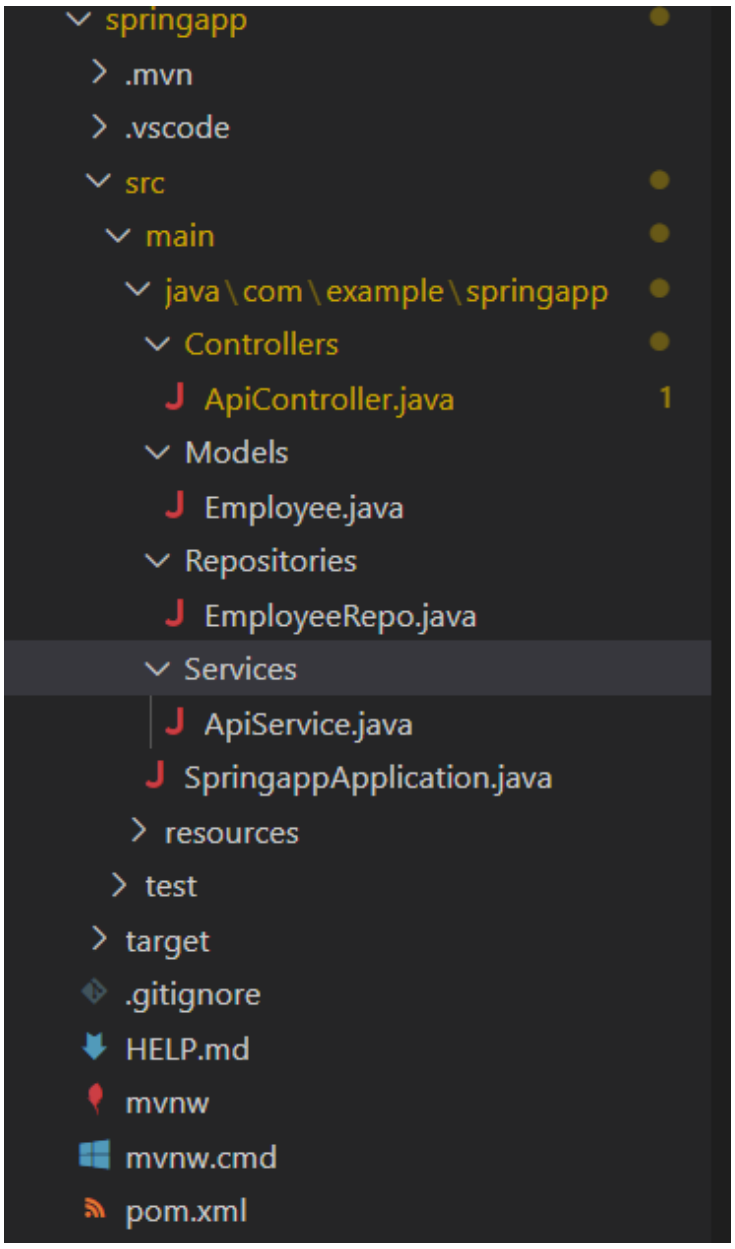
None

Q1. **GET Employee Details with Sorting and Pagination**

Overview:
To perform Sorting & Pagination operations of the employee's details in API.

Functional Requirements:

- Build an application for getting employee details.
- Have to create a folder named "Controllers" inside the springapp in src.
- Inside the Controllers create a java file named "ApiController.java".
- Next have to create the same inside the springapp named as "Models".
- Inside Models, you should create the java file named "Employee.java".
- And create Repositories folder and inside that create "EmployeeRepo.java"
  Create 6 variables
- 1. employeeId- int
- 2. firstName- string
- 3. lastName- string
- 4. email- string
- 5. mobile- long
- 6. address- string
  as well as create getters and setters and constructors for the corresponding variables.
  Finally, create the "Service" folder, and inside that create the "ApiService.java" file.
  The project structure looks like this image



Core Platform
OpenJDK 11

API:

Question 1(16 Marks):
POST - "/" --> true/false
GET - "/{field} --> List of Employee object

Question 2(16 Marks):
GET - "/{offset}/{pagesize}" --> Page content
GET - "/{offset}/{pagesize}/{field}" --> Page content

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:
To start/run the application '**mvn spring-boot:run**'

Answer Key & Solution

Section 1 - MCQ

Q1 SELECT name FROM Employee

Solution

No Solution

Q2 All of the above

Solution

No Solution

Q3 It supports SQL-like syntax

Solution

No Solution

Q4 A relationship between two entities where one entity can have multiple instances of the other entity

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 The entity with the @ManyToOne annotation

Solution

No Solution

Q7 By using the @ManyToOne annotation in one entity and the @OneToMany annotation in the other entity.

Solution

No Solution

Q8 BOTH LENGTH AND SUM

Solution

No Solution

Q9 OneToOne

Solution

No Solution

Section 2 - Coding

Q1 Solution cannot be displayed