REPOSITORY PATTERN AND QUERY METHODS IN SPRING DATA JPA

Sandra Kumi

REVIEWER Mr. Thomas Darko

Repository Pattern and Query Methods in Spring Data JPA

Introduction to Repository Pattern

The Repository Pattern is a design pattern used to abstract the data access logic in an application. It provides a collection-like interface for accessing domain objects and managing their persistence. In Spring Data JPA, the repository pattern is implemented using repository interfaces that extend the JpaRepository interface, providing CRUD operations and query methods.

Repository Pattern and Query Methods in Spring Data JPA

Implementing Repository Interfaces

In Spring Data JPA, repository interfaces are used to define methods for interacting with the database.
These interfaces extend the JpaRepository interface, which provides methods for common CRUD operations
(Create, Read, Update, Delete).

Example of a basic repository interface:

```java import
org.springframework.data.jpa.repository.JpaRepository;

public interface EmployeeRepository extends JpaRepository<Employee, Long> {
// Additional custom query methods can be defined here
}
```

This repository interface allows you to perform CRUD operations on Employee entities without needing to write any SQL queries or implement any methods manually.

Repository Pattern and Query Methods in Spring Data JPA

Custom Query Methods Using Method Naming Conventions

Spring Data JPA provides the ability to define custom query methods using method naming conventions.
These methods are automatically implemented by Spring Data JPA based on the method names.

Examples of custom query methods:

```java import
org.springframework.data.jpa.repository.JpaRepository; import
java.util.List;

public interface DoctorRepository extends JpaRepository<Doctor, Long> {
    // Find doctors by specialty
    List<Doctor> findBySpeciality(String speciality);

    // Find doctors by last name
    List<Doctor> findByLastName(String lastName);

    // Count doctors by specialty     long
countBySpeciality(String speciality);
```

```
    // Check if a doctor with a specific last name exists
boolean existsByLastName(String lastName);



}
```

These methods allow you to perform complex queries without writing custom SQL or JPQL.

Spring Data JPA automatically interprets the method names and generates the appropriate queries.

Repository Pattern and Query Methods in Spring Data JPA

Using Repositories in Services and Controllers

Repositories are typically used in service classes and controllers to interact with the database. Service classes encapsulate business logic and use repositories to manage data persistence.

Example of a service class using a repository:

```java import
org.springframework.beans.factory.annotation.Autowired; import
org.springframework.stereotype.Service;

import java.util.List;

@Service
public class DoctorService {

    @Autowired     private DoctorRepository
doctorRepository;

    public Doctor createDoctor(Doctor doctor) {
return doctorRepository.save(doctor);
    }
```

```
    public List<Doctor> getAllDoctors() {


        return doctorRepository.findAll();
    }


    public void deleteDoctor(Long id) {
doctorRepository.deleteById(id);
    }
}
```

In this example, the DoctorService class uses the DoctorRepository to perform CRUD operations on Doctor entities.