Step 1: Implementing CRUD Operations using JpaRepository

We'll use the EmployeeRepository and DepartmentRepository interfaces to implement CRUD operations for employees and departments.

EmployeeService.java:

package com.example.employeemanagementsystem.service;

import com.example.employeemanagementsystem.entity.Employee;

import com.example.employeemanagementsystem.repository.EmployeeRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class EmployeeService {

private final EmployeeRepository employeeRepository;

@Autowired

public EmployeeService(EmployeeRepository employeeRepository) {

this.employeeRepository = employeeRepository;

}

public List<Employee> getAllEmployees() {

return employeeRepository.findAll();

}

public Employee getEmployeeById(Long id) {

return employeeRepository.findById(id).orElseThrow();

}

public Employee createEmployee(Employee employee) {

return employeeRepository.save(employee);

}

public Employee updateEmployee(Employee employee) {

return employeeRepository.save(employee);

}

public void deleteEmployee(Long id) {

employeeRepository.deleteById(id);

}

}

DepartmentService.java:

package com.example.employeemanagementsystem.service;

import com.example.employeemanagementsystem.entity.Department;

import com.example.employeemanagementsystem.repository.DepartmentRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class DepartmentService {

private final DepartmentRepository departmentRepository;

@Autowired

public DepartmentService(DepartmentRepository departmentRepository) {

this.departmentRepository = departmentRepository;

}

public List<Department> getAllDepartments() {

return departmentRepository.findAll();

}

public Department getDepartmentById(Long id) {

return departmentRepository.findById(id).orElseThrow();

}

public Department createDepartment(Department department) {

return departmentRepository.save(department);

}

public Department updateDepartment(Department department) {

return departmentRepository.save(department);

}

public void deleteDepartment(Long id) {

departmentRepository.deleteById(id);

}

}

Step 2: Implementing RESTful Endpoints using EmployeeController and DepartmentController

We'll create RESTful endpoints for CRUD operations using EmployeeController and DepartmentController.

EmployeeController.java:

package com.example.employeemanagementsystem.controller;

import com.example.employeemanagementsystem.entity.Employee;

import com.example.employeemanagementsystem.service.EmployeeService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/api/employees")

public class EmployeeController {

private final EmployeeService employeeService;

@Autowired

public EmployeeController(EmployeeService employeeService) {

this.employeeService = employeeService;

}

@GetMapping

public ResponseEntity<List<Employee>> getAllEmployees() {

return new ResponseEntity<>(employeeService.getAllEmployees(), HttpStatus.OK);

}

@GetMapping("/{id}")

public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {

return new ResponseEntity<>(employeeService.getEmployeeById(id), HttpStatus.OK);

}

@PostMapping

public ResponseEntity<Employee> createEmployee(@RequestBody Employee employee) {

return new ResponseEntity<>(employeeService.createEmployee(employee), HttpStatus.CREATED);

}

@PutMapping("/{id}")

public ResponseEntity<Employee> updateEmployee(@PathVariable Long id, @RequestBody Employee employee) {

return new ResponseEntity<>(employeeService.updateEmployee(employee), HttpStatus.OK);

}

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteEmployee(@PathVariable Long id) {

employeeService.deleteEmployee(id);

return new ResponseEntity<>(HttpStatus.NO\_CONTENT);

}

}

DepartmentController.java:

package com.example.employeemanagementsystem.controller;

import com.example.employeemanagementsystem.entity.Department;

import com.example.employeemanagementsystem.service.DepartmentService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/api/departments")

public class DepartmentController {

private final DepartmentService departmentService;

@Autowired

public DepartmentController(DepartmentService departmentService) {

this.departmentService = departmentService;

}

@GetMapping

public ResponseEntity<List<Department>> getAllDepartments() {

return new ResponseEntity<>(departmentService.getAllDepartments(), HttpStatus.OK);

}

@GetMapping("/{id}")

public ResponseEntity<Department> getDepartmentById(@PathVariable Long id) {

return new ResponseEntity<>(departmentService.getDepartmentById(id), HttpStatus.OK);

}

@PostMapping

public ResponseEntity<Department> createDepartment(@RequestBody Department department) {

return new ResponseEntity<>(departmentService.createDepartment(department), HttpStatus.CREATED);

}

@PutMapping("/{