**Exercise 4: Employee Management System - Implementing CRUD Operations**

**Business Scenario:**

Implement CRUD operations for managing employees and departments.

**Instructions:**

1. **Basic CRUD Operations:**

Use JpaRepository methods to create, read, update, and delete employees and departments.

Implement RESTful endpoints for these operations using EmployeeController and DepartmentController.

package com.company.controller;

import com.company.entity.Employee;

import com.company.repository.EmployeeRepository;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

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

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

import java.util.List;

@RestController

@RequestMapping("/employees")

public class EmployeeController {

@Autowired

private EmployeeRepository employeeRepo;

public Employee createEmployee(@RequestBody Employee employee) {

return employeeRepo.save(employee);

}

@GetMapping

public List<Employee> getAllEmployees() {

return employeeRepo.findAll();

}

@GetMapping("/{id}")

public Employee getEmployeeById(@PathVariable Long id) {

return employeeRepo.findById(id).orElse(null);

}

@PutMapping("/{id}")

public Employee updateEmployee(@PathVariable Long id, @RequestBody Employee updateEmployee) {

return employeeRepo.findById(id).map(emp -> {

emp.setName(updateEmployee.getName());

emp.setEmail(updateEmployee.getEmail());

emp.setDepartment(updateEmployee.getDepartment());

return employeeRepo.save(emp);

}).orElse(null);

}

@DeleteMapping("/{id}")

public String deleteEmployee(@PathVariable Long id) {

employeeRepo.deleteById(id);

return id + " deleted";

}

@GetMapping("/department/{name}")

public List<Employee> getByDeptName(@PathVariable String name) {

return employeeRepo.findEmployeesByDepartmentName(name);

}

@GetMapping("/paginated")

public Page<Employee> getPaginatedEmployees(

@RequestParam(defaultValue = "0") int page,

@RequestParam(defaultValue = "5") int size,

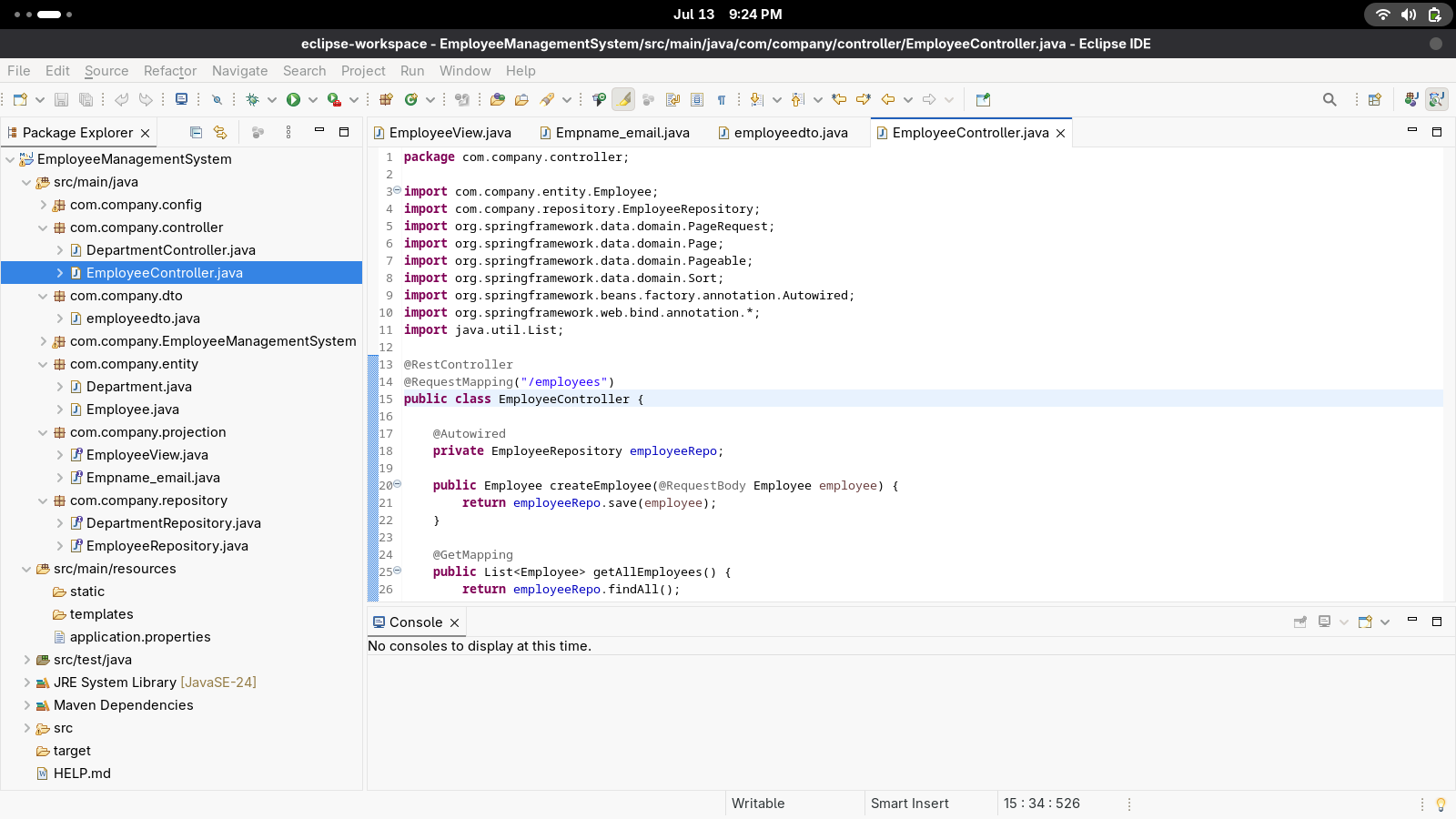
@RequestParam(defaultValue = "name")String sortBy){

Pageable pageable = PageRequest.of(page, size, Sort.by(sortBy));

return employeeRepo.findAll(pageable);

}

}



package com.company.controller;

import com.company.entity.Department;

import com.company.repository.DepartmentRepository;

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

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

import java.util.List;

@RestController

@RequestMapping("/departments")

public class DepartmentController {

@Autowired

public DepartmentRepository departmentRepo;

@PostMapping

public Department createDepartment(@RequestBody Department department) {

return departmentRepo.save(department);

}

@GetMapping

public List<Department> getDepartment() {

return departmentRepo.findAll();

}

@GetMapping("/{id}")

public Department getDepartmentById(@PathVariable Long id) {

return departmentRepo.findById(id).orElse(null);

}

@PutMapping("/{id}")

public Department updateDepartment(@PathVariable Long id, @RequestBody Department department) {

return departmentRepo.findById(id).map(dep->{

dep.setName(department.getName());

return departmentRepo.save(dep);

}).orElse(null);

}

@DeleteMapping("/{id}")

public String deleteDepartment(@PathVariable Long id) {

departmentRepo.deleteById(id);

return id + " deleted";

}

}

