**Exercise 8: Employee Management System - Creating Projections**

**Business Scenario:**

Create projections to fetch specific data subsets from the employee and department entities.

**Instructions:**

1. **Projections:**

Define interface-based and class-based projections.

Use @Value and constructor expressions to control the fetched data.

package com.company.repository;

import com.company.entity.Employee;

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

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import com.company.projection.EmployeeView;

import com.company.dto.employeeDTO;

import java.util.List;

public interface EmployeeRepository extends JpaRepository <Employee, Long> {

List<Employee> findByName(String name);

List<Employee> findByDepartment(String name);

@Query(name = "Employee.findByEmail")

List<Employee> findEmployeesByEmail(@Param("email") String email);

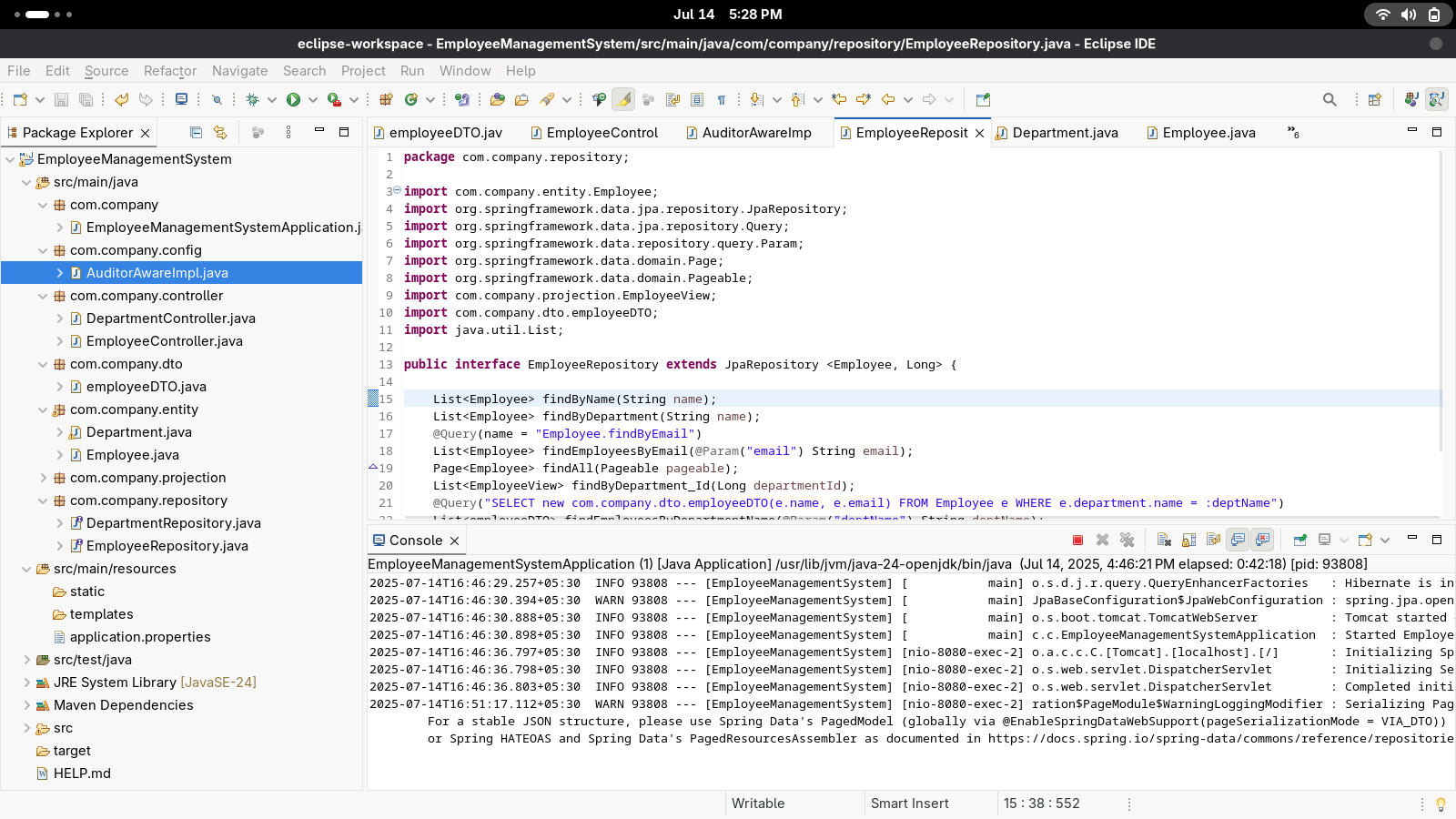
Page<Employee> findAll(Pageable pageable);

List<EmployeeView> findByDepartment\_Id(Long departmentId);

@Query("SELECT new com.company.dto.employeeDTO(e.name, e.email) FROM Employee e WHERE e.department.name = :deptName")

List<employeeDTO> findEmployeesByDepartmentName(@Param("deptName") String deptName);

}



package com.company.dto;

public class employeeDTO {

private String name;

private String email;

public employeeDTO(String name, String email) {

this.name = name;

this.email = email;

}

public String getName() {

return name;

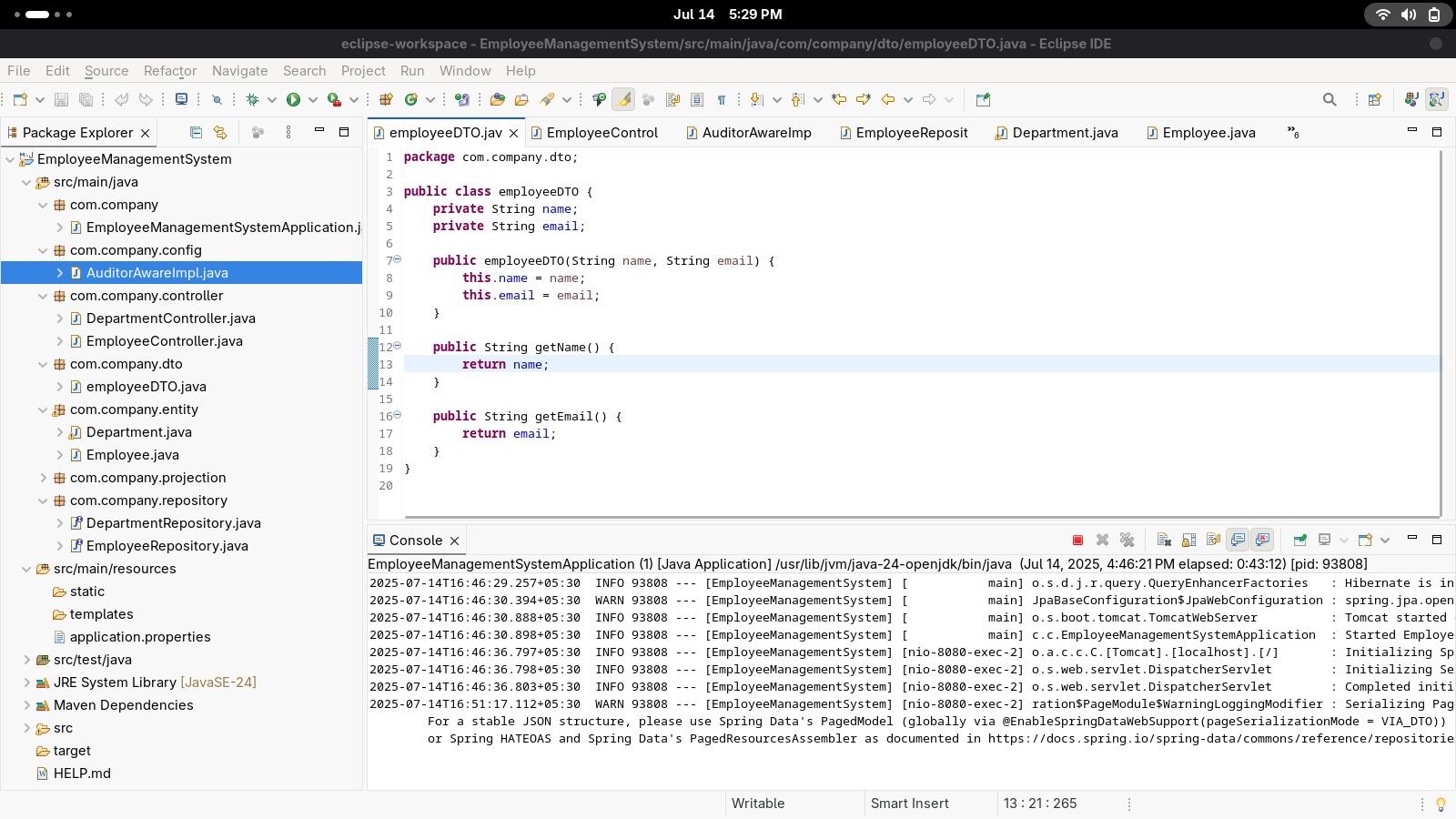
}

public String getEmail() {

return email;

}

}



package com.company.controller;

import com.company.entity.Employee;

import com.company.repository.EmployeeRepository;

import com.company.dto.employeeDTO;

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;

@PostMapping

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("/by-department/{name}")

public List<employeeDTO> getEmployeesByDept(@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);

}

}