**Exercise 1: Employee Management System - Overview and Setup**

Model  
Employee

package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
@Entity  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String designation;  
 private double salary;  
  
 @ManyToOne  
 @JoinColumn(name = "department\_id")  
 private Department department;  
}

Department  
package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
import java.util.List;  
  
@Entity  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Department {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 @OneToMany(mappedBy = "department")  
 private List<Employee> employees;  
}

Repository  
  
EmployeeRepository

package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
import java.util.List;  
  
@Entity  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Department {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 @OneToMany(mappedBy = "department")  
 private List<Employee> employees;  
}

DepartmentRepository

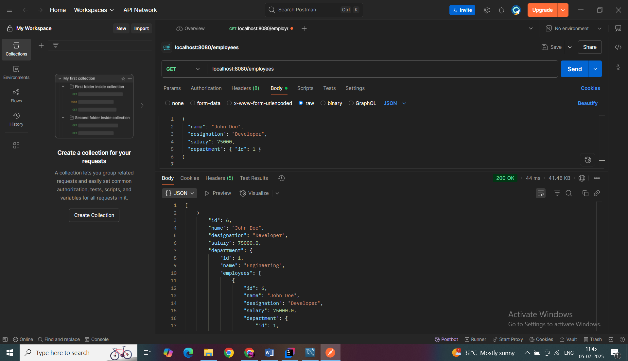
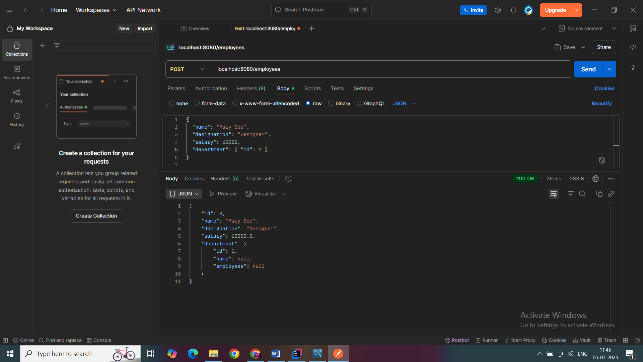
package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Department;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface DepartmentRepository extends JpaRepository<Department, Long> {  
}

Service

EmployeeService

package com.example.Employee.Management.service;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.repository.EmployeeRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 public List<Employee> getAllEmployees() {  
 return employeeRepository.findAll();  
 }  
  
 public Employee saveEmployee(Employee employee) {  
 return employeeRepository.save(employee);  
 }  
}

Controller  
EmployeeController  
  
package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.service.EmployeeService;  
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 EmployeeService employeeService;  
  
 @GetMapping  
 public List<Employee> getEmployees() {  
 return employeeService.getAllEmployees();  
 }  
  
 @PostMapping  
 public Employee createEmployee(@RequestBody Employee employee) {  
 return employeeService.saveEmployee(employee);  
 }  
}

OUTPUT:  
  

**Exercise 2: Employee Management System - Creating Entities**

**ENTITIES  
Department**

package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
@Entity  
@Table(name = "employee")  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String email;  
  
 @ManyToOne  
 @JoinColumn(name = "department\_id") // foreign key column in 'employee' table  
 private Department department;  
}

EMPLOYEE  
  
package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
import java.util.List;  
  
@Entity  
@Table(name = "department")  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Department {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 @OneToMany(mappedBy = "department", cascade = CascadeType.*ALL*, fetch = FetchType.*LAZY*)  
 @ToString.Exclude  
 private List<Employee> employees;  
}

**Exercise 3: Employee Management System - Creating Repositories  
  
Repository  
  
Employee**

package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
}

Department

package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Department;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface DepartmentRepository extends JpaRepository<Department, Long> {  
}

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

Controller  
  
EmployeeController  
  
package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.repository.EmployeeRepository;  
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 employeeRepository;  
  
 @PostMapping  
 public Employee createEmployee(@RequestBody Employee employee) {  
 return employeeRepository.save(employee);  
 }  
  
 @GetMapping  
 public List<Employee> getAllEmployees() {  
 return employeeRepository.findAll();  
 }  
  
 @GetMapping("/{id}")  
 public Employee getEmployeeById(@PathVariable Long id) {  
 return employeeRepository.findById(id).orElse(null);  
 }  
}

DepartmentController  
  
package com.example.Employee.Management.controller;  
  
  
  
import com.example.Employee.Management.model.Department;  
import com.example.Employee.Management.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  
 private DepartmentRepository departmentRepository;  
  
 @PostMapping  
 public Department createDepartment(@RequestBody Department department) {  
 return departmentRepository.save(department);  
 }  
  
 @GetMapping  
 public List<Department> getAllDepartments() {  
 return departmentRepository.findAll();  
 }  
  
 @GetMapping("/{id}")  
 public Department getDepartmentById(@PathVariable Long id) {  
 return departmentRepository.findById(id).orElse(null);  
 }  
}

**Exercise 5: Employee Management System - Defining Query Methods**

1.Custom Query Methods

package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
import java.util.List;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
  
 // 1. Find employees by name  
 List<Employee> findByName(String name);  
  
 // 2. Find employees by department id  
 List<Employee> findByDepartmentId(Long departmentId);  
  
 // 3. Find employees whose salary is greater than a certain value  
 List<Employee> findBySalaryGreaterThan(Double salary);  
}

2. @Query Annotation-Based Methods  
  
package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.data.jpa.repository.Query;  
import org.springframework.data.repository.query.Param;  
  
import java.util.List;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
  
 // JPQL query: Find by designation  
 @Query("SELECT e FROM Employee e WHERE e.designation = :designation")  
 List<Employee> getEmployeesByDesignation(@Param("designation") String designation);  
  
 // Native SQL: Find top N employees by salary  
 @Query(value = "SELECT \* FROM employee ORDER BY salary DESC LIMIT :limit", nativeQuery = true)  
 List<Employee> getTopPaidEmployees(@Param("limit") int limit);  
}

3.Named Queries (@NamedQuery)  
Employee  
package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
@Entity  
@Getter @Setter @ToString  
@NamedQueries({  
 @NamedQuery(name = "Employee.findByEmail",  
 query = "SELECT e FROM Employee e WHERE e.email = :email"),  
 @NamedQuery(name = "Employee.findByDepartmentAndSalary",  
 query = "SELECT e FROM Employee e WHERE e.department.id = :deptId AND e.salary > :minSalary")  
})  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String email;  
 private String designation;  
 private Double salary;  
  
 @ManyToOne  
 @JoinColumn(name = "department\_id")  
 private Department department;  
}

EmployeeRepository

import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.data.jpa.repository.Query;  
import org.springframework.data.repository.query.Param;  
  
import java.util.List;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
 @Query(name = "Employee.findByEmail")  
 Employee findByEmail(@Param("email") String email);  
  
 @Query(name = "Employee.findByDepartmentAndSalary")  
 List<Employee> findByDepartmentAndSalary(@Param("deptId") Long deptId, @Param("minSalary") Double minSalary);  
  
}

**Exercise 6: Employee Management System - Implementing Pagination and Sorting**

Entities  
Employee  
  
package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
@Entity  
@Table(name = "employee")  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 private String email;  
  
 private String designation;  
  
 private Double salary;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "department\_id") // foreign key column  
 @ToString.Exclude // to avoid recursive printing  
 private Department department;  
}

Department  
  
  
package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
import java.util.List;  
  
@Entity  
@Table(name = "department")  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Department {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 // One department can have many employees  
 @OneToMany(mappedBy = "department", cascade = CascadeType.*ALL*)  
 @ToString.Exclude // to prevent recursive output  
 private List<Employee> employees;  
}

Repository  
  
EmployeeRepository

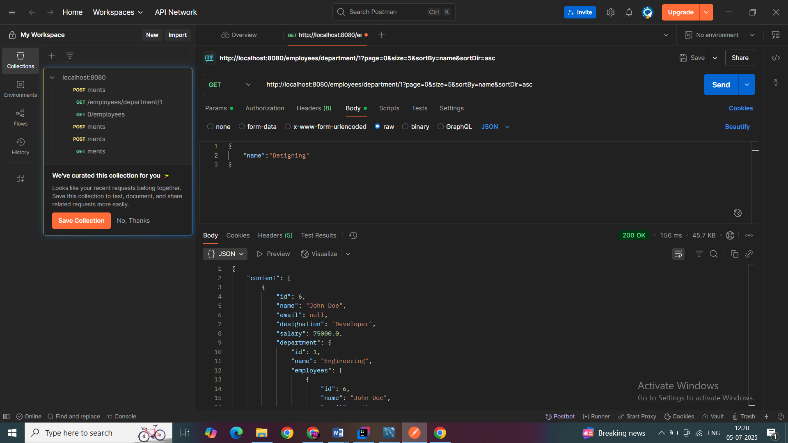
package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
  
 Page<Employee> findAll(Pageable pageable);  
  
 Page<Employee> findByDepartmentId(Long departmentId, Pageable pageable);  
}

EmployeeService

import org.springframework.data.domain.Pageable;  
import org.springframework.stereotype.Service;  
  
@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 public Page<Employee> getAllEmployees(Pageable pageable) {  
 return employeeRepository.findAll(pageable);  
 }  
  
 public Page<Employee> getEmployeesByDepartment(Long deptId, Pageable pageable) {  
 return employeeRepository.findByDepartmentId(deptId, pageable);  
 }  
}

EmployeeController  
  
  
  
package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.service.EmployeeService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.PageRequest;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.domain.Sort;  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/employees")  
public class EmployeeController {  
  
 @Autowired  
 private EmployeeService employeeService;  
  
 // Basic Pagination & Sorting  
 @GetMapping  
 public Page<Employee> getAllEmployees(  
 @RequestParam(defaultValue = "0") int page, // page number  
 @RequestParam(defaultValue = "5") int size, // page size  
 @RequestParam(defaultValue = "id") String sortBy, // field name  
 @RequestParam(defaultValue = "asc") String sortDir // asc/desc  
 ) {  
 Sort sort = sortDir.equalsIgnoreCase("asc") ? Sort.*by*(sortBy).ascending()  
 : Sort.*by*(sortBy).descending();  
 Pageable pageable = PageRequest.*of*(page, size, sort);  
 return employeeService.getAllEmployees(pageable);  
 }  
  
 // Filter by department with pagination & sorting  
 @GetMapping("/department/{deptId}")  
 public Page<Employee> getEmployeesByDepartment(  
 @PathVariable Long deptId,  
 @RequestParam(defaultValue = "0") int page,  
 @RequestParam(defaultValue = "5") int size,  
 @RequestParam(defaultValue = "id") String sortBy,  
 @RequestParam(defaultValue = "asc") String sortDir  
 ) {  
 Sort sort = sortDir.equalsIgnoreCase("asc") ? Sort.*by*(sortBy).ascending()  
 : Sort.*by*(sortBy).descending();  
 Pageable pageable = PageRequest.*of*(page, size, sort);  
 return employeeService.getEmployeesByDepartment(deptId, pageable);  
 }  
}

OUTPUT:



**Exercise 7: Employee Management System - Enabling Entity Auditing**

**Model  
  
Auditable**package com.example.Employee.Management.model;  
  
import jakarta.persistence.Column;  
import jakarta.persistence.EntityListeners;  
import jakarta.persistence.MappedSuperclass;  
import lombok.Getter;  
import lombok.Setter;  
import org.springframework.data.annotation.\*;  
import org.springframework.data.jpa.domain.support.AuditingEntityListener;  
  
import java.time.LocalDateTime;  
  
@Getter  
@Setter  
@MappedSuperclass  
@EntityListeners(AuditingEntityListener.class)  
public abstract class Auditable {  
  
 @CreatedDate  
 @Column(updatable = false)  
 private LocalDateTime createdDate;  
  
 @LastModifiedDate  
 private LocalDateTime lastModifiedDate;  
  
 @CreatedBy  
 @Column(updatable = false)  
 private String createdBy;  
  
 @LastModifiedBy  
 private String modifiedBy;  
}

**Employee**package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import lombok.\*;  
  
@Entity  
@Table(name = "employee")  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Employee extends Auditable{  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 private String email;  
  
 private String designation;  
  
 private Double salary;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "department\_id") // foreign key column  
 @ToString.Exclude // to avoid recursive printing  
 private Department department;  
}

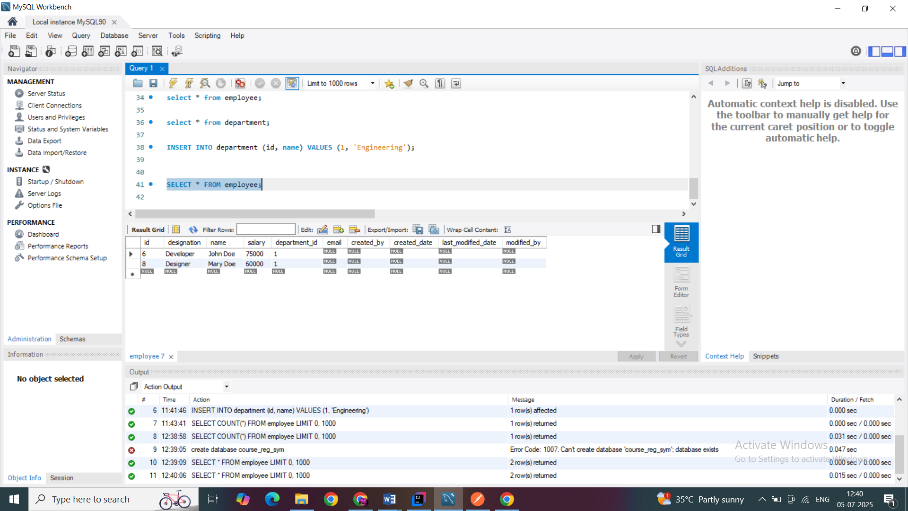
**Department**@Entity  
@Table(name = "department")  
@Getter  
@Setter  
@NoArgsConstructor  
@AllArgsConstructor  
@ToString  
public class Department extends Auditable {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 // One department can have many employees  
 @OneToMany(mappedBy = "department", cascade = CascadeType.*ALL*)  
 @ToString.Exclude // to prevent recursive output  
 private List<Employee> employees;  
}

**Config**package com.example.Employee.Management.config;  
  
import org.springframework.context.annotation.Configuration;  
import org.springframework.data.jpa.repository.config.EnableJpaAuditing;  
  
@Configuration  
@EnableJpaAuditing  
public class AuditConfig {  
}

**EmployeeRepository**package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
  
 Page<Employee> findAll(Pageable pageable);  
  
 Page<Employee> findByDepartmentId(Long departmentId, Pageable pageable);  
}

**EmployeeService**package com.example.Employee.Management.service;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.repository.EmployeeRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.stereotype.Service;  
  
@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 public Page<Employee> getAllEmployees(Pageable pageable) {  
 return employeeRepository.findAll(pageable);  
 }  
  
 public Page<Employee> getEmployeesByDepartment(Long deptId, Pageable pageable) {  
 return employeeRepository.findByDepartmentId(deptId, pageable);  
 }  
}

**EmployeeController**package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.service.EmployeeService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.PageRequest;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.domain.Sort;  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/employees")  
public class EmployeeController {  
  
 @Autowired  
 private EmployeeService employeeService;  
  
 // Basic Pagination & Sorting  
 @GetMapping  
 public Page<Employee> getAllEmployees(  
 @RequestParam(defaultValue = "0") int page, // page number  
 @RequestParam(defaultValue = "5") int size, // page size  
 @RequestParam(defaultValue = "id") String sortBy, // field name  
 @RequestParam(defaultValue = "asc") String sortDir // asc/desc  
 ) {  
 Sort sort = sortDir.equalsIgnoreCase("asc") ? Sort.*by*(sortBy).ascending()  
 : Sort.*by*(sortBy).descending();  
 Pageable pageable = PageRequest.*of*(page, size, sort);  
 return employeeService.getAllEmployees(pageable);  
 }  
  
 // Filter by department with pagination & sorting  
 @GetMapping("/department/{deptId}")  
 public Page<Employee> getEmployeesByDepartment(  
 @PathVariable Long deptId,  
 @RequestParam(defaultValue = "0") int page,  
 @RequestParam(defaultValue = "5") int size,  
 @RequestParam(defaultValue = "id") String sortBy,  
 @RequestParam(defaultValue = "asc") String sortDir  
 ) {  
 Sort sort = sortDir.equalsIgnoreCase("asc") ? Sort.*by*(sortBy).ascending()  
 : Sort.*by*(sortBy).descending();  
 Pageable pageable = PageRequest.*of*(page, size, sort);  
 return employeeService.getEmployeesByDepartment(deptId, pageable);  
 }  
}

**OUTPUT:  
  
**

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

**Projection**

package com.example.Employee.Management.projection;  
  
interface DepartmentInfo {  
 String getName();  
}

package com.example.Employee.Management.projection;  
  
public interface EmployeeInfo {  
 String getName();  
  
 String getDesignation();  
  
 DepartmentInfo getDepartment();  
}

**DTO**package com.example.Employee.Management.DTO;  
  
public class EmployeeDTO {  
 private String name;  
 private String departmentName;  
  
 public EmployeeDTO(String name, String departmentName) {  
 this.name = name;  
 this.departmentName = departmentName;  
 }  
  
 // Getters  
 public String getName() {  
 return name;  
 }  
  
 public String getDepartmentName() {  
 return departmentName;  
 }  
}

**Repository**package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.DTO.EmployeeDTO;  
import com.example.Employee.Management.projection.EmployeeInfo;  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.data.jpa.repository.Query;  
  
import java.util.List;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
 List<EmployeeInfo> findAllBy(); // returns projection  
 @Query("SELECT new com.example.Employee.Management.dto.EmployeeDTO(e.name, e.department.name) FROM Employee e")  
 List<EmployeeDTO> fetchEmployeeDTOs();  
}

**Controller**package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.projection.EmployeeInfo;  
import com.example.Employee.Management.repository.EmployeeRepository;  
import com.example.Employee.Management.service.EmployeeService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.PageRequest;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.domain.Sort;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/employees")  
public class EmployeeController {  
  
 @Autowired  
 private EmployeeService employeeService;  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
 // Basic Pagination & Sorting  
 @GetMapping  
 public Page<Employee> getAllEmployees(  
 @RequestParam(defaultValue = "0") int page, // page number  
 @RequestParam(defaultValue = "5") int size, // page size  
 @RequestParam(defaultValue = "id") String sortBy, // field name  
 @RequestParam(defaultValue = "asc") String sortDir // asc/desc  
 ) {  
 Sort sort = sortDir.equalsIgnoreCase("asc") ? Sort.*by*(sortBy).ascending()  
 : Sort.*by*(sortBy).descending();  
 Pageable pageable = PageRequest.*of*(page, size, sort);  
 return employeeService.getAllEmployees(pageable);  
 }  
 @GetMapping("/projection/interface")  
 public List<EmployeeInfo> getEmployeeInterfaceProjection() {  
 return employeeRepository.findAllBy();  
 }  
  
 // Filter by department with pagination & sorting  
 @GetMapping("/department/{deptId}")  
 public Page<Employee> getEmployeesByDepartment(  
 @PathVariable Long deptId,  
 @RequestParam(defaultValue = "0") int page,  
 @RequestParam(defaultValue = "5") int size,  
 @RequestParam(defaultValue = "id") String sortBy,  
 @RequestParam(defaultValue = "asc") String sortDir  
 ) {  
 Sort sort = sortDir.equalsIgnoreCase("asc") ? Sort.*by*(sortBy).ascending()  
 : Sort.*by*(sortBy).descending();  
 Pageable pageable = PageRequest.*of*(page, size, sort);  
 return employeeService.getEmployeesByDepartment(deptId, pageable);  
 }  
}

**Exercise 9: Employee Management System - Customizing Data Source Configuration**

**Entity  
  
Audit**package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import java.time.LocalDateTime;  
  
@Entity  
public class Audit {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String action;  
  
 private LocalDateTime timestamp;  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getAction() {  
 return action;  
 }  
  
 public void setAction(String action) {  
 this.action = action;  
 }  
  
 public LocalDateTime getTimestamp() {  
 return timestamp;  
 }  
  
 public void setTimestamp(LocalDateTime timestamp) {  
 this.timestamp = timestamp;  
 }  
}

**Employee**package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String email;  
  
 @ManyToOne  
 @JoinColumn(name = "department\_id")  
 private Department department;  
  
 // Getters and setters  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 public Department getDepartment() {  
 return department;  
 }  
  
 public void setDepartment(Department department) {  
 this.department = department;  
 }  
}

**Department**package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import java.util.List;  
  
@Entity  
public class Department {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 @OneToMany(mappedBy = "department")  
 private List<Employee> employees;  
  
 // Getters and setters  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public List<Employee> getEmployees() {  
 return employees;  
 }  
  
 public void setEmployees(List<Employee> employees) {  
 this.employees = employees;  
 }  
}

**Repository**

**AuditRepository**

package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Audit;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface AuditRepository extends JpaRepository<Audit, Long> {  
}

**EmployeeRepository**

package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
}

**Service  
  
AuditService**@Service  
public class AuditService {  
  
 @Autowired  
 private AuditRepository auditLogRepository;  
  
 public void logAction(String action) {  
 Audit log = new Audit();  
 log.setAction(action);  
 log.setTimestamp(LocalDateTime.*now*());  
 auditLogRepository.save(log);  
 }  
}

**EmployeeService**package com.example.Employee.Management.service;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.repository.EmployeeRepository;  
import com.example.Employee.Management.service.AuditService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 @Autowired  
 private AuditService auditService;  
  
 public Employee saveEmployee(Employee employee) {  
 Employee saved = employeeRepository.save(employee);  
 auditService.logAction("Created employee: " + saved.getName());  
 return saved;  
 }  
  
 public List<Employee> getAllEmployees() {  
 return employeeRepository.findAll();  
 }  
}

**EmployeeController**package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.service.EmployeeService;  
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 EmployeeService employeeService;  
  
 @PostMapping  
 public Employee createEmployee(@RequestBody Employee employee) {  
 return employeeService.saveEmployee(employee);  
 }  
  
 @GetMapping  
 public List<Employee> getAllEmployees() {  
 return employeeService.getAllEmployees();  
 }  
}

**Exercise 10: Employee Management System - Hibernate-Specific Features**

**Model  
Employee**package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import org.hibernate.annotations.CacheConcurrencyStrategy;  
import org.hibernate.annotations.DynamicInsert;  
import org.hibernate.annotations.DynamicUpdate;  
  
@Entity  
@DynamicInsert // only include non-null fields in insert statement  
@DynamicUpdate // only include modified fields in update statement  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String email;  
  
 @ManyToOne  
 @JoinColumn(name = "department\_id")  
 private Department department;  
  
 // Getters and setters  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }

**Department**package com.example.Employee.Management.model;  
  
import jakarta.persistence.\*;  
import org.hibernate.annotations.DynamicInsert;  
import org.hibernate.annotations.DynamicUpdate;  
  
import java.util.List;  
  
@Entity  
@DynamicInsert // only include non-null fields in insert statement  
@DynamicUpdate  
public class Department {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 @OneToMany(mappedBy = "department")  
 private List<Employee> employees;  
  
 // Getters and setters  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public List<Employee> getEmployees() {  
 return employees;  
 }  
  
 public void setEmployees(List<Employee> employees) {  
 this.employees = employees;  
 }  
  
 public Department(Long id) {  
 this.id = id;  
 }  
}

**Service  
  
EmployeeService**

package com.example.Employee.Management.service;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.repository.EmployeeRepository;  
import com.example.Employee.Management.service.AuditService;  
import jakarta.transaction.Transactional;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 @Autowired  
 private AuditService auditService;  
  
 public Employee saveEmployee(Employee employee) {  
 Employee saved = employeeRepository.save(employee);  
 auditService.logAction("Created employee: " + saved.getName());  
 return saved;  
 }  
  
 public List<Employee> getAllEmployees() {  
 return employeeRepository.findAll();  
 }  
  
 @Transactional  
 public void saveEmployeesInBatch(List<Employee> employees) {  
 for (int i = 0; i < employees.size(); i++) {  
 employeeRepository.save(employees.get(i));  
 // flush and clear periodically to avoid memory issues  
 if (i % 20 == 0) {  
 employeeRepository.flush(); // add `extends JpaRepository<Employee, Long>, JpaSpecificationExecutor<Employee>` to enable this  
 }  
 }  
 }  
  
}

**Repository**package com.example.Employee.Management.repository;  
  
import com.example.Employee.Management.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Long> {  
}

**Controller**package com.example.Employee.Management.controller;  
  
import com.example.Employee.Management.model.Employee;  
import com.example.Employee.Management.service.EmployeeService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/employees")  
public class EmployeeController {  
  
 @Autowired  
 private EmployeeService employeeService;  
  
 @PostMapping  
 public Employee createEmployee(@RequestBody Employee employee) {  
 return employeeService.saveEmployee(employee);  
 }  
  
 @GetMapping  
 public List<Employee> getAllEmployees() {  
 return employeeService.getAllEmployees();  
 }  
 @PostMapping("/batch")  
 public ResponseEntity<String> saveEmployeesBatch(@RequestBody List<Employee> employees) {  
 employeeService.saveEmployeesInBatch(employees);  
 return ResponseEntity.*ok*("Batch inserted successfully");  
 }  
  
}