**Hands On 6: Implement many to many relationship between Employee and Skill**

**Employee.java**

package com.cognizant.employee.model;  
  
import jakarta.persistence.\*;  
import java.util.Date;  
import java.util.Set;  
  
@Entity  
@Table(name = "employee")  
public class Employee {  
  
 @Id  
 @Column(name = "em\_id")  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private int id;  
  
 @Column(name = "em\_name")  
 private String name;  
  
 @Column(name = "em\_salary")  
 private double salary;  
  
 @Column(name = "em\_permanent")  
 private boolean permanent;  
  
 @Column(name = "em\_date\_of\_birth")  
 @Temporal(TemporalType.*DATE*)  
 private Date dateOfBirth;  
  
 @ManyToOne  
 @JoinColumn(name = "em\_dp\_id")  
 private Department department;  
  
 @ManyToMany(fetch = FetchType.*EAGER*)  
 @JoinTable(  
 name = "employee\_skill",  
 joinColumns = @JoinColumn(name = "es\_em\_id"),  
 inverseJoinColumns = @JoinColumn(name = "es\_sk\_id")  
 )  
 private Set<Skill> skillList;  
  
 *// --- Getters and Setters ---* public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public double getSalary() {  
 return salary;  
 }  
  
 public void setSalary(double salary) {  
 this.salary = salary;  
 }  
  
 public boolean isPermanent() {  
 return permanent;  
 }  
  
 public void setPermanent(boolean permanent) {  
 this.permanent = permanent;  
 }  
  
 public Date getDateOfBirth() {  
 return dateOfBirth;  
 }  
  
 public void setDateOfBirth(Date dateOfBirth) {  
 this.dateOfBirth = dateOfBirth;  
 }  
  
 public Department getDepartment() {  
 return department;  
 }  
  
 public void setDepartment(Department department) {  
 this.department = department;  
 }  
  
 public Set<Skill> getSkillList() {  
 return skillList;  
 }  
  
 public void setSkillList(Set<Skill> skillList) {  
 this.skillList = skillList;  
 }  
  
 *// --- toString() ---* @Override  
 public String toString() {  
 return "Employee{" +  
 "\nid=" + id +  
 ", \nname='" + name + '\'' +  
 ", \nsalary=" + salary +  
 ", \npermanent=" + permanent +  
 ", \ndateOfBirth=" + dateOfBirth +  
 ", \ndepartment=" + department +  
 ", \nskills=" + skillList +  
 "\n}";  
 }  
}

**Department.java**

package com.cognizant.employee.model;  
  
import jakarta.persistence.\*;  
import java.util.Set;  
  
@Entity  
@Table(name = "department")  
public class Department {  
  
 @Id  
 @Column(name = "dp\_id")  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private int id;  
  
 @Column(name = "dp\_name")  
 private String name;  
  
 @OneToMany(mappedBy = "department", fetch = FetchType.*EAGER*)  
 private Set<Employee> employeeList;  
  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
  
 public String getName() { return name; }  
 public void setName(String name) { this.name = name; }  
  
 public Set<Employee> getEmployeeList() { return employeeList; }  
 public void setEmployeeList(Set<Employee> employeeList) { this.employeeList = employeeList; }  
  
 @Override  
 public String toString() {  
 return "Department{" +  
 "\nid=" + id +  
 ", \nname='" + name + '\'' +  
 '}';  
 }  
}

**DepartmentRepository.java**

package com.cognizant.employee.repository;  
  
import com.cognizant.employee.model.Department;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface DepartmentRepository extends JpaRepository<Department, Integer> {  
}

**SkillService.java**

package com.cognizant.employee.service;  
  
import java.util.List;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import com.cognizant.employee.model.Skill;  
import com.cognizant.employee.repository.SkillRepository;  
  
@Service  
public class SkillService {  
  
 @Autowired  
 private SkillRepository skillRepository;  
  
 public List<Skill> getAllSkills() {  
 return skillRepository.findAll();  
 }  
  
 public Skill get(int id) {  
 return skillRepository.findById(id).orElse(null);  
 }  
}

**EmployeeApplication.java**

package com.cognizant.employee;  
  
import com.cognizant.employee.model.Employee;  
import com.cognizant.employee.model.Skill;  
import com.cognizant.employee.service.EmployeeService;  
import com.cognizant.employee.service.DepartmentService;  
  
import com.cognizant.employee.service.SkillService;  
import jakarta.transaction.Transactional;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.boot.CommandLineRunner;  
import org.springframework.context.annotation.Bean;  
  
import java.util.Set;  
  
@SpringBootApplication  
public class EmployeeApplication {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(EmployeeApplication.class);  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(EmployeeApplication.class, args);  
 }  
  
 @Bean  
 public CommandLineRunner run(EmployeeService employeeService, DepartmentService departmentService, SkillService skillService) {  
 return args -> {testAddSkillToEmployee(employeeService, skillService); };  
 }  
  
 @Transactional  
 private void testAddSkillToEmployee(EmployeeService employeeService, SkillService skillService) {  
 *LOGGER*.info("START testAddSkillToEmployee");  
  
 int employeeId = 1; int skillId = 2; Employee employee = employeeService.get(employeeId);  
 Skill skill = skillService.get(skillId);  
  
 *LOGGER*.info("Employee Before Update:");  
 *LOGGER*.info(" - ID : {}", employee.getId());  
 *LOGGER*.info(" - Name : {}", employee.getName());  
  
 *LOGGER*.info("Existing Skills:");  
 for (Skill s : employee.getSkillList()) {  
 *LOGGER*.info(" • [{}] {}", s.getId(), s.getName());  
 }  
  
 Set<Skill> skillList = employee.getSkillList();  
 skillList.add(skill);  
 employee.setSkillList(skillList);  
 employeeService.save(employee);  
  
 *LOGGER*.info("Added Skill: [{}] {}", skill.getId(), skill.getName());  
  
 *LOGGER*.info("Employee After Update:");  
 for (Skill s : employee.getSkillList()) {  
 *LOGGER*.info(" • [{}] {}", s.getId(), s.getName());  
 }  
  
 *LOGGER*.info("END testAddSkillToEmployee");  
 }  
  
}

**Output**

