**Hands on Exercise – 5 Implement one to many relationship between Employee and Department**

**CODE:**

**Department Class: -**

package com.cognizant.ex2handson3.model;  
import jakarta.persistence.\*;  
import java.util.Set;

@Entity  
@Table(name = "department")  
public class Department {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private int id;  
  
 @Column(name = "name")  
 private String name;  
  
 @OneToMany(mappedBy = "department",fetch = FetchType.*EAGER*)  
 private Set<Employee> employeeList;  
  
 public Department() {  
 }  
  
 public Department(int id, String name, Set<Employee> employeeList) {  
 this.id = id;  
 this.name = name;  
 this.employeeList = 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;  
 }  
}

**Department Service Class: -**

package com.cognizant.ex2handson3.service;  
import com.cognizant.ex2handson3.model.Department;  
import com.cognizant.ex2handson3.repository.DepartmentRepository;  
import jakarta.transaction.Transactional;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
@Service  
public class DepartmentService {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(DepartmentService.class);  
  
 @Autowired  
 private DepartmentRepository departmentRepository;  
  
 @Transactional  
 public Department get(int id) {  
 *LOGGER*.info("Start");  
 return departmentRepository.findById(id).orElse(null);  
 }  
  
 @Transactional  
 public void save(Department department) {  
 *LOGGER*.info("Start");  
 departmentRepository.save(department);  
 *LOGGER*.info("End");  
 }  
}

**Department Repository Interface: -**

package com.cognizant.ex2handson3.repository;  
import com.cognizant.ex2handson3.model.Department;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface DepartmentRepository extends JpaRepository<Department,Integer> {  
}

**Main class: -**

package com.cognizant.ex2handson3;  
import com.cognizant.ex2handson3.model.Department;  
import com.cognizant.ex2handson3.model.Employee;  
import com.cognizant.ex2handson3.service.DepartmentService;  
import com.cognizant.ex2handson3.service.EmployeeService;  
import com.cognizant.ex2handson3.service.SkillService;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.ApplicationContext;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import java.sql.Date;  
  
@SpringBootApplication  
public class Ex2handson3Application {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(Ex2handson3Application.class);  
  
 public static void main(String[] args) {  
 ApplicationContext context = SpringApplication.*run*(Ex2handson3Application.class, args);  
  
 EmployeeService employeeService = context.getBean(EmployeeService.class);  
 DepartmentService departmentService = context.getBean(DepartmentService.class);  
 SkillService skillService = context.getBean(SkillService.class);  
  
 *testGetDepartment*(departmentService);   
 }  
  
 private static void testGetDepartment(DepartmentService departmentService){  
 *LOGGER*.info("Start");  
 Department department=departmentService.get(1);  
 *LOGGER*.debug("Department: {}", department);  
 *LOGGER*.debug("Employees: {}", department.getEmployeeList());  
  
 System.*out*.println("Department: " + department.getName());  
 department.getEmployeeList().forEach(emp ->  
 System.*out*.println("Employee: " + emp.getName() + ", Salary: " + emp.getSalary())  
 );  
 *LOGGER*.info("End");  
 }  
}

**OUTPUT:**

