**Hands on 1: Spring Data JPA - Quick Example**

package com.cognizant.orm\_lean.Repository;

import com.cognizant.orm\_lean.model.Country;

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

import org.springframework.stereotype.Repository;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

package com.cognizant.orm\_lean.model;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "country")

public class Country {

    @Id

    @Column(name = "co\_code")

    private String code;

    @Column(name = "co\_name")

    private String name;

    public String getCode() {

        return code;

    }

    public void setCode(String code) {

        this.code = code;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    @Override

    public String toString() {

        return "Country -->> [code=" + code + ", name=" + name + "]";

    }

}

package com.cognizant.orm\_lean.service;

import com.cognizant.orm\_lean.model.Country;

import com.cognizant.orm\_lean.Repository.CountryRepository;

import jakarta.transaction.Transactional;

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

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

  @Autowired

    private CountryRepository countryRepository;

    @Transactional

    public List<Country> getAllCountries() {

        return countryRepository.findAll();

    }

}

package com.cognizant.orm\_lean;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import com.cognizant.orm\_lean.service.CountryService;

import com.cognizant.orm\_lean.model.Country;

import java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class OrmLeanApplication {

    private static final Logger LOGGER = LoggerFactory.getLogger(OrmLeanApplication.class);

   private static CountryService countryService;

public static void main(String[] args) {

ApplicationContext context =  SpringApplication.run(OrmLeanApplication.class, args);

countryService = context.getBean(CountryService.class);

System.out.println("\n\n");

LOGGER.info("inside main");

testGetAllCountries();

}

  private static void testGetAllCountries() {

  System.out.println("\n\n");

        LOGGER.info("Start");

        List<Country> countries = countryService.getAllCountries();

        LOGGER.debug("countries={}", countries);

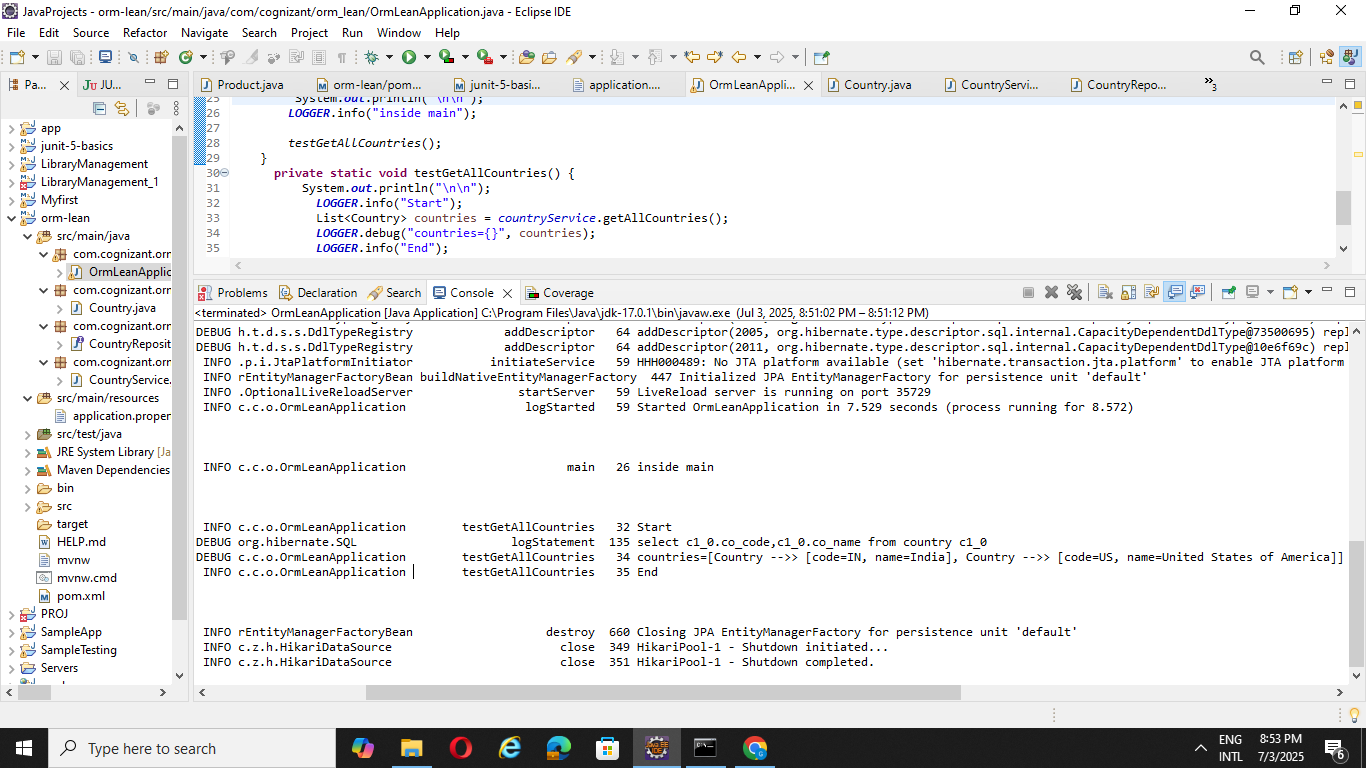
        LOGGER.info("End");

        System.out.println("\n\n");

    }

}

**OUTPUT:**



**Hands on 4 : Difference between JPA, Hibernate and Spring Data JPA**

->Implementation Spring data JPA:

package com.example.repository;

import com.example.model.Employee;

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

import org.springframework.stereotype.Repository;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

package com.example.model;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class Employee {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Integer id;

    private String name;

    // Getters and setters

    public Integer getId() {

        return id;

    }

    public void setId(Integer id) {

        this.id = id;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

}

package com.example.service;

import com.example.model.Employee;

import com.example.repository.EmployeeRepository;

import jakarta.transaction.Transactional;

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

import org.springframework.stereotype.Service;

@Service

public class EmployeeService {

    @Autowired

    private EmployeeRepository employeeRepository;

    @Transactional

    public void addEmployee(Employee employee) {

        employeeRepository.save(employee);

    }

}

package com.example;

import com.example.model.Employee;

import com.example.service.EmployeeService;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class MainApp {

    private static EmployeeService employeeService;

    public static void main(String[] args) {

        ApplicationContext context = SpringApplication.run(MainApp.class, args);

        employeeService = context.getBean(EmployeeService.class);

        Employee emp = new Employee();

        emp.setName("Arun");

        employeeService.addEmployee(emp);

        Employee emp1 = new Employee();

        emp1.setId(1);

        emp1.setName("Deva");

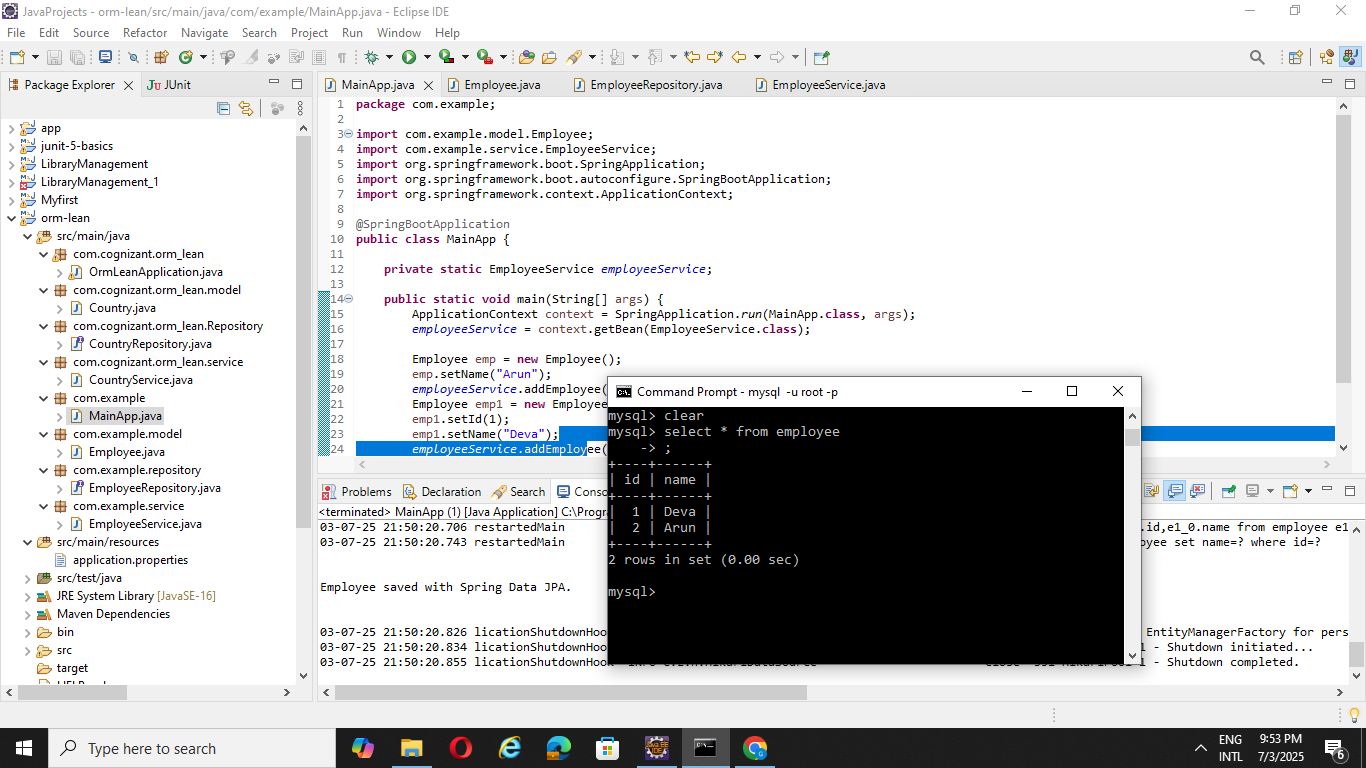
        employeeService.addEmployee(emp1);

        System.out.println("\n\nEmployee saved with Spring Data JPA.\n\n");

    }

}

**OUTPUT:**



🡪Impementaton Using Hibernate

package com.example.dao;

import com.example.model.Employee;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import org.hibernate.cfg.Configuration;

import org.hibernate.service.ServiceRegistry;

import org.hibernate.SessionFactory;

public class EmployeeDAO {

    private static SessionFactory factory;

    static {

        try {

            Configuration configuration = new Configuration().configure();

            ServiceRegistry serviceRegistry =

                new StandardServiceRegistryBuilder()

                    .applySettings(configuration.getProperties()).build();

            factory = configuration.buildSessionFactory(serviceRegistry);

        } catch (Throwable ex) {

            System.err.println("SessionFactory creation failed: " + ex);

            throw new ExceptionInInitializerError(ex);

        }

    }

    public Integer addEmployee(Employee employee) {

        Transaction tx = null;

        Integer employeeID = null;

        try (Session session = factory.openSession()) {

            tx = session.beginTransaction();

            employeeID = (Integer) session.save(employee);

            tx.commit();

        } catch (Exception e) {

            if (tx != null) tx.rollback();

            e.printStackTrace();

        }

        return employeeID;

    }

}

package com.example.model;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class Employee {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Integer id;

    private String name;

    private String department;

    // getters and setters

    public Integer getId() {

        return id;

    }

    public void setId(Integer id) {

        this.id = id;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public String getDepartment() {

        return department;

    }

    public void setDepartment(String department) {

        this.department = department;

    }

    @Override

    public String toString() {

        return "Employee [id=" + id + ", name=" + name + ", department=" + department + "]";

    }

}

package com.example.hibernate\_standalone;

import com.example.dao.EmployeeDAO;

import com.example.model.Employee;

public class MainApplication

{

    public static void main( String[] args )

    {

     EmployeeDAO dao = new EmployeeDAO();

        Employee emp = new Employee();

        emp.setName("Alice");

        emp.setDepartment("Finance");

        Integer id = dao.addEmployee(emp);

        System.out.println(“Employee created with ID: " + id);

    }

}