**Spring -data-jpa-handson**

**Country.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

*@Entity*

*@Table*(name = "country")

public class Country {

*@Id*

*@Column*(name = "co\_code")

private String code;

*@Column*(name = "co\_name")

private String name;

// Getters and Setters

// toString()

*@Override*

public String toString() {

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

}

}

**CountryRepository.java**

package com.cognizant.ormlearn.repository;

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

import org.springframework.stereotype.Repository;

import com.cognizant.ormlearn.model.Country;

*@Repository*

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService.java**

package com.cognizant.ormlearn.service;

import java.util.List;

import jakarta.transaction.Transactional;

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

import org.springframework.stereotype.Service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

*@Transactional*

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**OrmlearnApplication.java**

package com.cognizant.ormlearn;

import java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

*@SpringBootApplication*

public class OrmlearnApplication {

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

private static CountryService *countryService*;

public static void main(String[] args) {

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

*countryService* = context.getBean(CountryService.class);

*testGetAllCountries*();

}

private static void testGetAllCountries() {

***LOGGER***.info("Start");

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

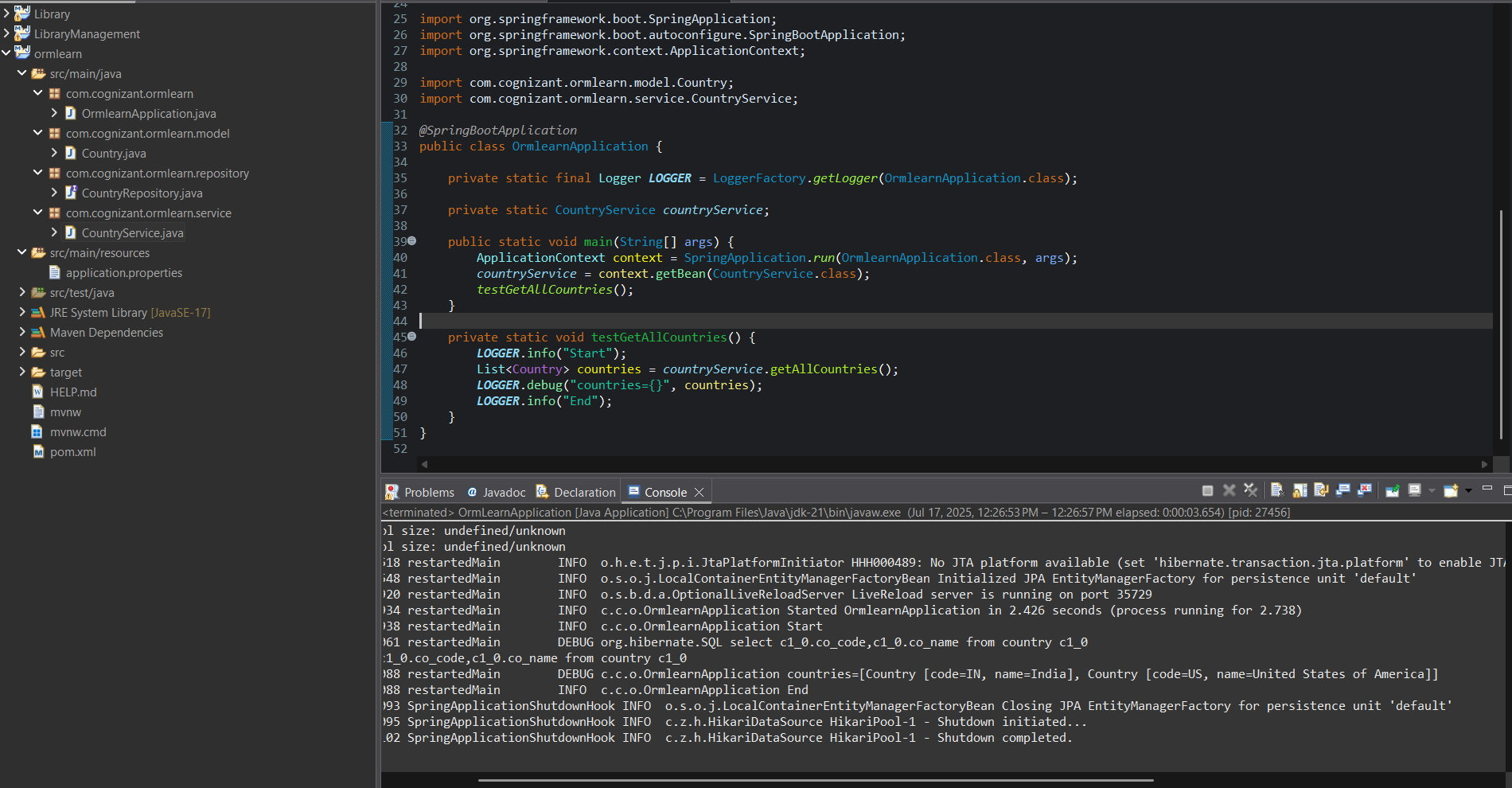
***LOGGER***.debug("countries={}", countries);

***LOGGER***.info("End");

}

}

**Output:**



**Exercise 2: Difference between JPA, Hibernate and Spring Data JPA**

**JPA (Java Persistence API)** is a specification provided by Java that defines a standard way to map Java objects to relational database tables. It acts as a set of rules or interfaces for object-relational mapping (ORM) but doesn't provide any actual implementation.

**Hibernate**, on the other hand, is a popular ORM framework and a concrete implementation of the JPA specification. It follows the rules defined by JPA and adds many additional features such as caching, lazy loading, and its own query language (HQL). While you can use Hibernate directly, it is commonly used as the JPA provider underneath frameworks like Spring.

**Spring Data JPA** is a part of the larger Spring ecosystem that simplifies the use of JPA by reducing boilerplate code and providing ready-made implementations of repository interfaces. It builds on top of JPA and Hibernate, allowing developers to perform common database operations with minimal code, such as using method names to automatically generate queries.