**Hands on 1**

**Spring Data JPA - Quick Example**

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

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

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

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.util.List;

@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();

countries.forEach(country -> LOGGER.debug("Country: {}", country));

LOGGER.info("End");

}

}

**Country.java**

**package** com.cognizant.orm\_learn.model;

**import** jakarta.persistence.\*;

@Entity

@Table(name = "country")

**public** **class** Country {

@Id

@Column(name = "code")

**private** String code;

@Column(name = "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 + "]";

}

}

**CountryRepository.java**

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

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

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService.java**

package com.cognizant.orm\_learn.service;

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

import com.cognizant.orm\_learn.repository.CountryRepository;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**Application.properties**

# Logging

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=debug

logging.level.org.hibernate.type.descriptor.sql=trace

# DB Config

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

spring.datasource.username=root

spring.datasource.password=SYSTEM

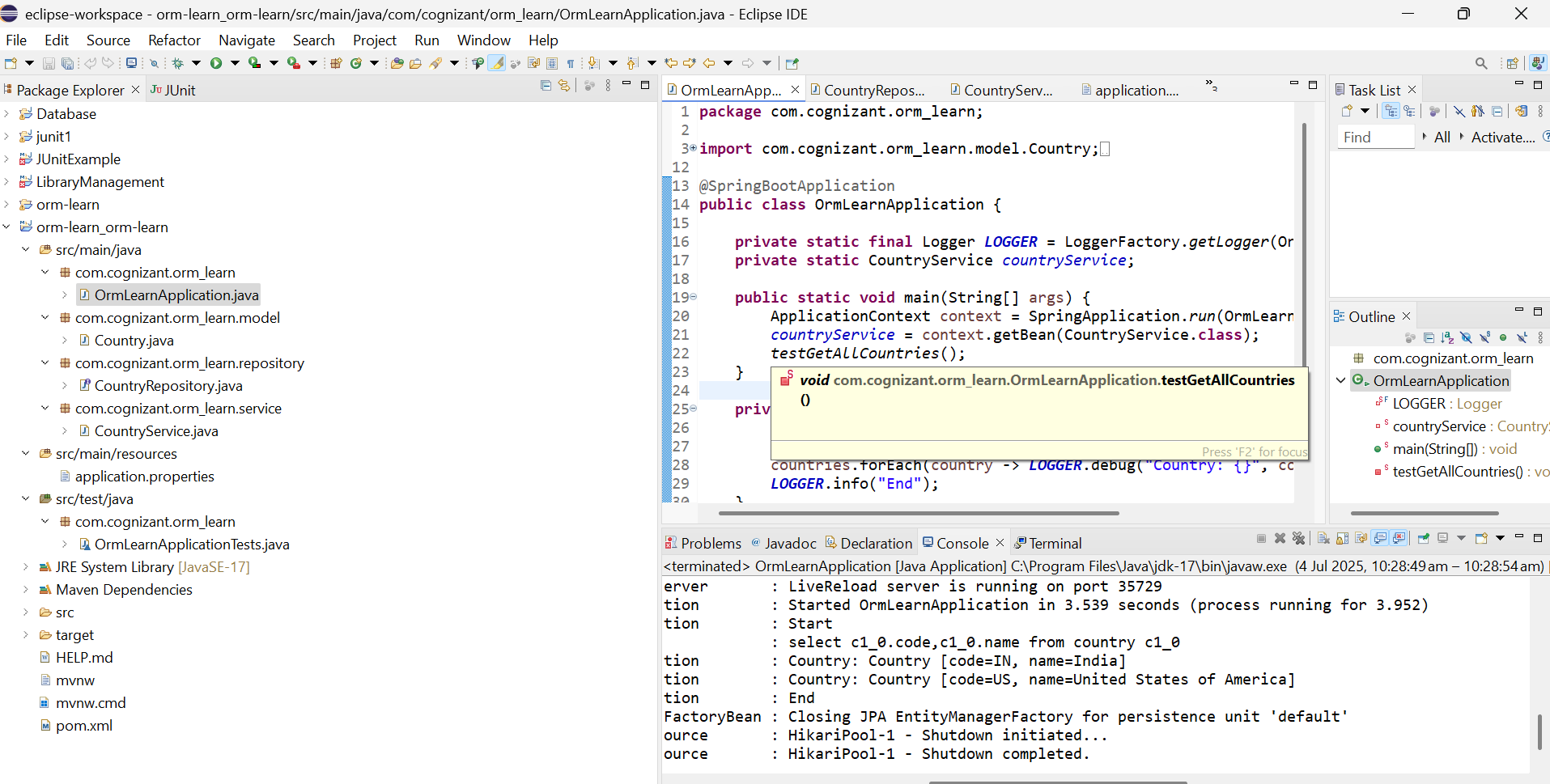
# Hibernate Dialect for Hibernate 6

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

# Schema validation (use validate if table already exists)

spring.jpa.hibernate.ddl-auto=validate

**Output:**

****

**Difference between JPA, Hibernate and Spring Data JPA**

| **Feature** | **JPA (Java Persistence API)** | **Hibernate** | **Spring Data JPA** |
| --- | --- | --- | --- |
| **Type** | Specification | Implementation of JPA | Abstraction layer on top of JPA |
| **Provides** | API/annotations only | ORM features + native SQL, HQL, caching | Repository interfaces and query methods |
| **Requires Implementation** | Yes – needs a provider like Hibernate | No – it's already a provider | No – uses JPA provider (like Hibernate) underneath |
| **Boilerplate Code** | Requires EntityManager, transactions manually | Slightly better but still requires code for common operations | Reduces boilerplate via JpaRepository, CrudRepository etc. |
| **Integration in Spring** | Manual configuration required | Can be configured easily | Auto-configured in Spring Boot |
| **Query Support** | JPQL | JPQL + HQL + Criteria API | JPQL + method names + @Query support |
| **Ease of Use** | Intermediate | Moderate | Very high |

**Bottom of Form**

**Implement services for managing Country**

**CountryService.java**

**package** com.cognizant.orm\_learn.service;

**import** com.cognizant.orm\_learn.model.Country;

**import** java.util.List;

**public** **interface** CountryService {

Country findCountryByCode(String code);

List<Country> getAllCountries();

}

**CountryServiceImpl.java**

**package** com.cognizant.orm\_learn.service;

**import** com.cognizant.orm\_learn.model.Country;

**import** com.cognizant.orm\_learn.repository.CountryRepository;

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

**import** org.springframework.stereotype.Service;

**import** java.util.List;

@Service

**public** **class** CountryServiceImpl **implements** CountryService {

@Autowired

**private** CountryRepository countryRepository;

@Override

**public** Country findCountryByCode(String code) {

**return** countryRepository.findById(code).orElse(**null**);

}

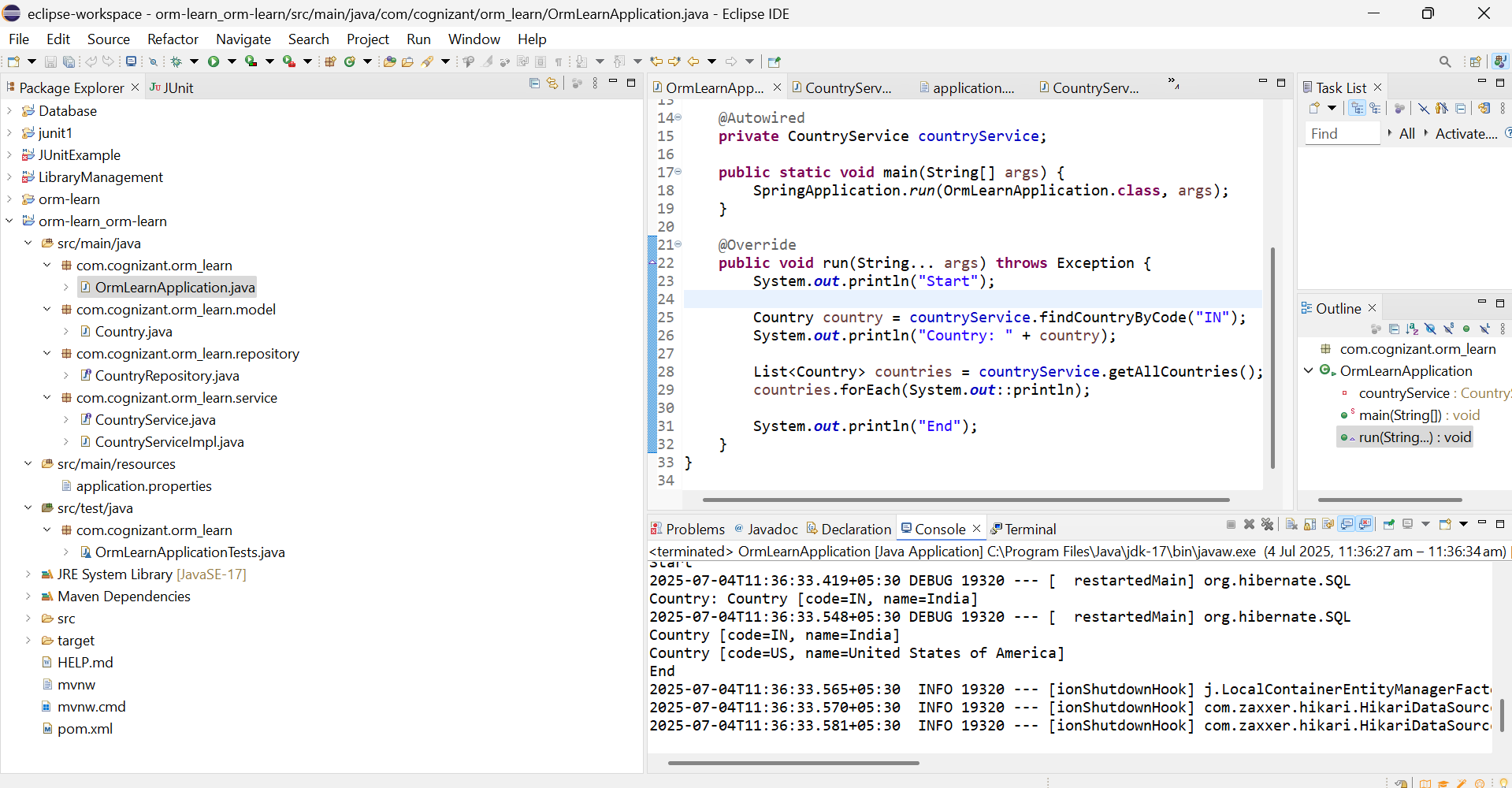
@Override

**public** List<Country> getAllCountries() {

**return** countryRepository.findAll();

}

}

**Output**

**Find a country based on country code**

**CountryServiceImpl.java**

**package** com.cognizant.orm\_learn.service;

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

**import** org.springframework.stereotype.Service;

**import** com.cognizant.orm\_learn.model.Country;

**import** com.cognizant.orm\_learn.repository.CountryRepository;

@Service

**public** **class** CountryServiceImpl **implements** CountryService {

@Autowired

**private** CountryRepository countryRepository;

@Override

**public** Country findCountryByCode(String code) {

**return** countryRepository.findByCode(code);

}

}

**CountryService.java**

**package** com.cognizant.orm\_learn.service;

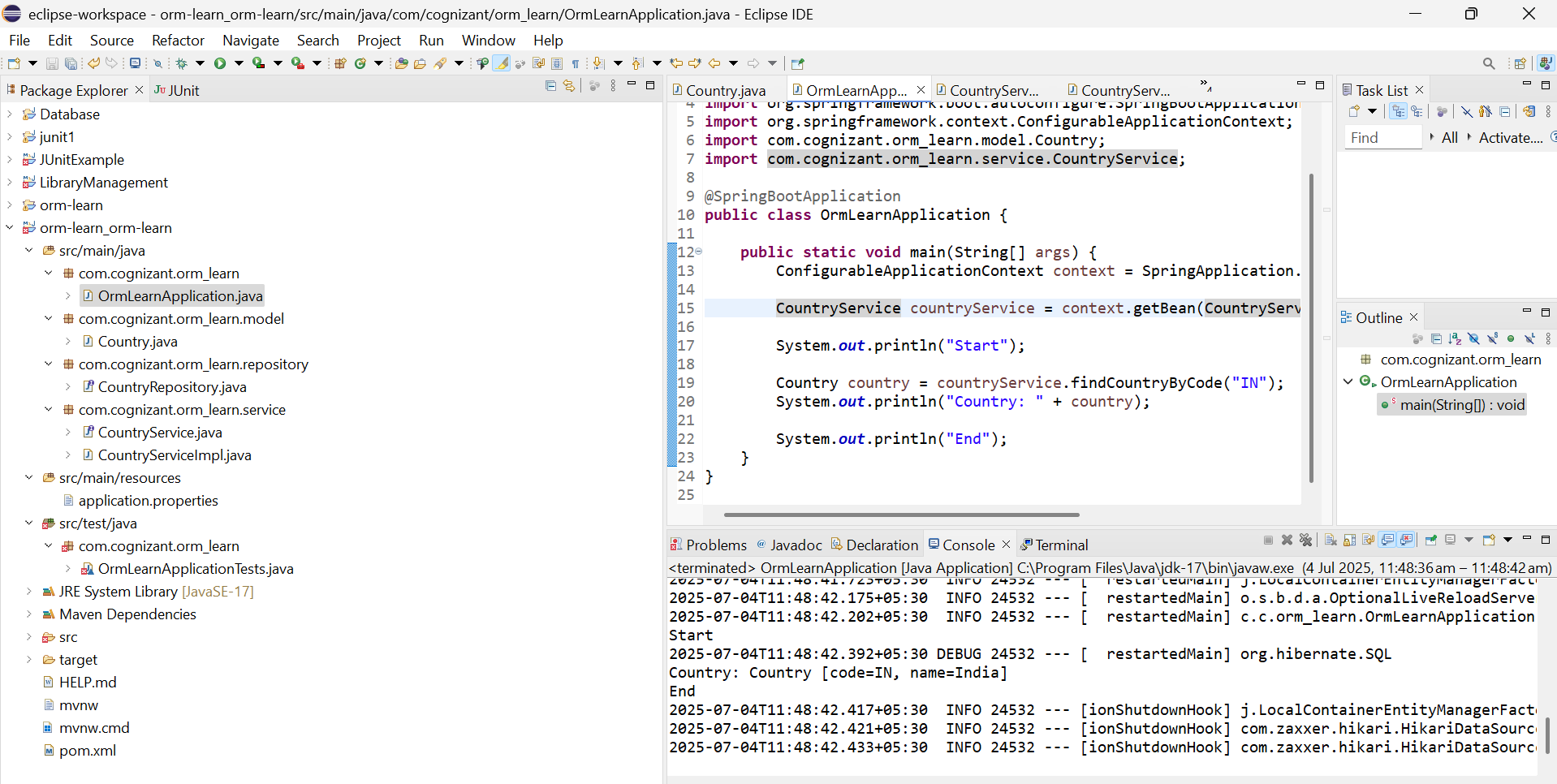
**import** com.cognizant.orm\_learn.model.Country;

**public** **interface** CountryService {

Country findCountryByCode(String code);

}

**Output:**

****