WEEK-3

**Spring Data JPA with Hibernate**

**Exercise-1: Spring Data JPA - Quick Example:**

**Application.properties:**

spring.application.name=orm-learn

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=trace

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

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n

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=1234

spring.jpa.hibernate.ddl-auto=validate

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

**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*(scanBasePackages = "com.cognizant.ormlearn")

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);

***LOGGER***.info("Inside main");

*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");

} }

**Country.java:**

package com.cognizant.ormlearn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Table;

import jakarta.persistence.Id;

import jakarta.persistence.Column;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

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();

}

}

**SQL Table creation:**

create schema ormlearn;

use ormlearn;

create table country (

code varchar(2) primary key,

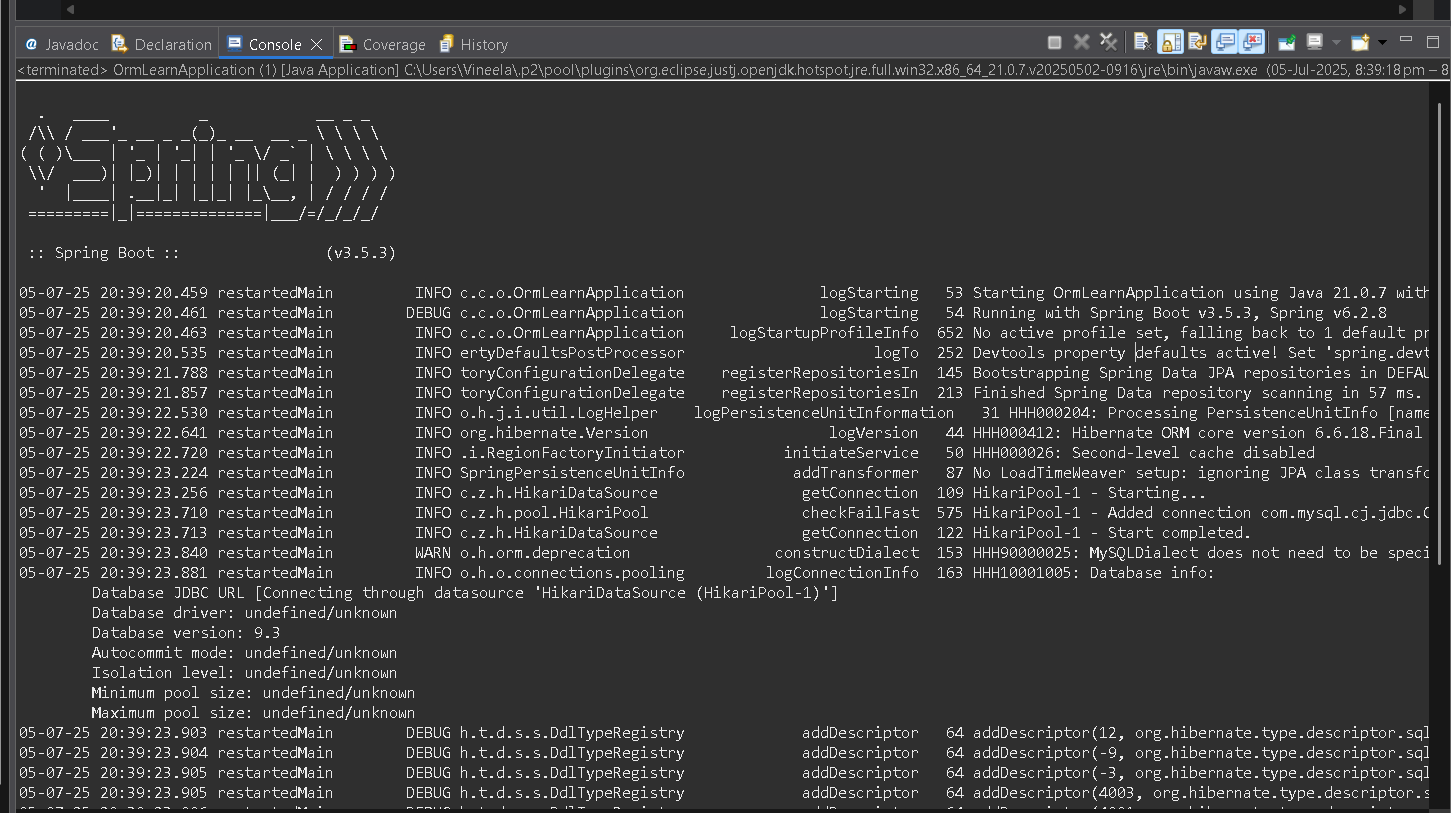
name varchar(50)

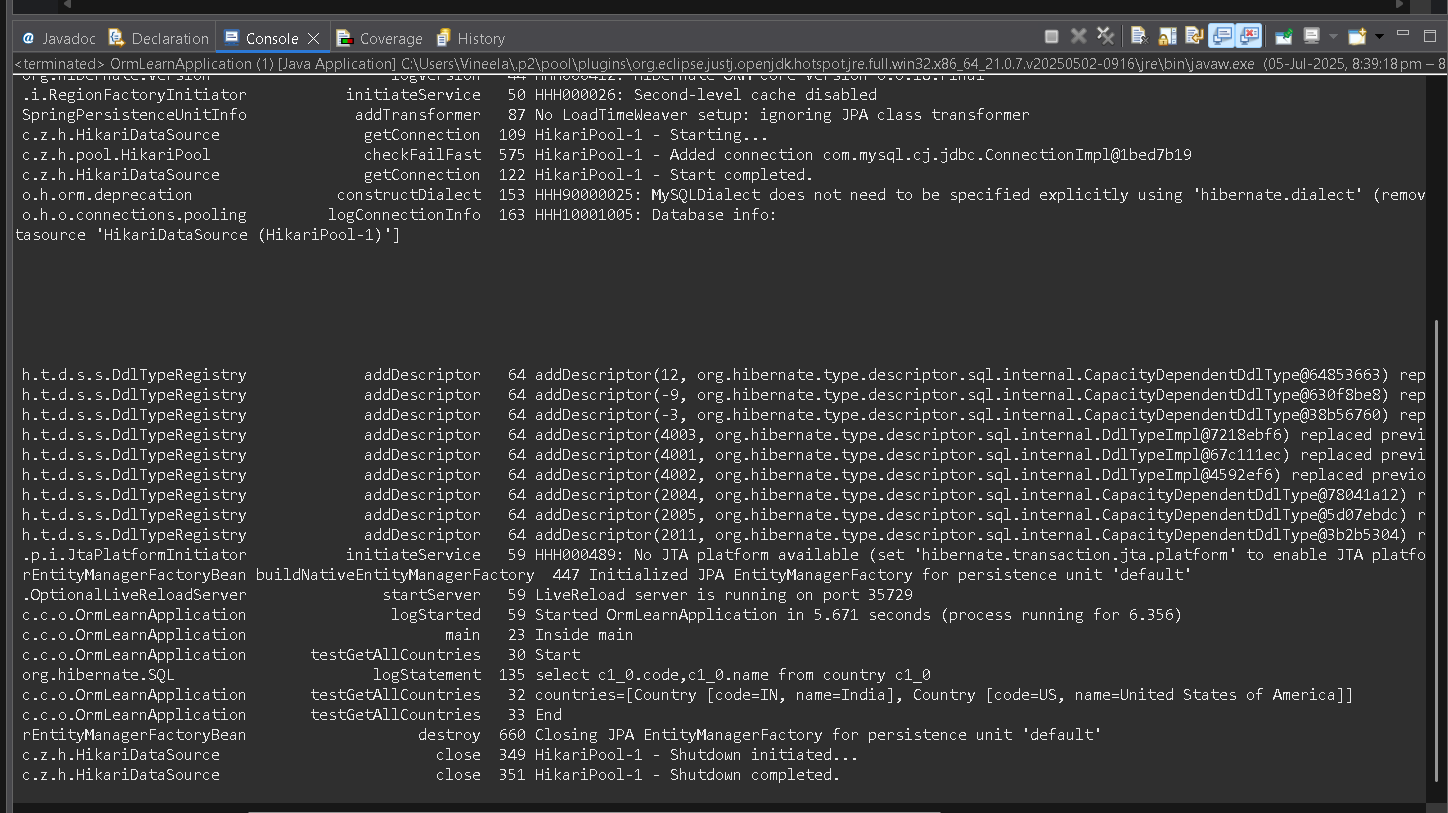
);

insert into country values ('IN', 'India');

insert into country values ('US', 'United States of America');

**OUTPUT:**

****

****

**Exercise-4: Difference between JPA, Hibernate and Spring Data JPA**

**1.Hibernate Demo:**

**Employee.hbm.xml:**

<hibernate-mapping>

<class name="com.example.hibernate.Employee" table="employee">

<id name="id" column="id">

<generator class="native"/>

</id>

<property name="firstName" column="first\_name"/>

<property name="lastName" column="last\_name"/>

<property name="salary" column="salary"/>

</class>

</hibernate-mapping>

**2.hibernate.cfg.xml:**

<hibernate-configuration>

<session-factory>

<property name="hibernate.connection.driver\_class">com.mysql.cj.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/empdb</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">1234</property>

<property name="hibernate.dialect">org.hibernate.dialect.MySQL5Dialect</property>

<property name="show\_sql">true</property>

<mapping resource="Employee.hbm.xml"/>

</session-factory>

</hibernate-configuration>

**App.java:**

package com.example.hibernate;

public class App {

public static void main(String[] args) {

EmployeeDAO dao = new EmployeeDAO();

Employee e1 = new Employee();

e1.setFirstName("Mankidi");

e1.setLastName("Vineela");

e1.setSalary(45000.0f);

Integer id = dao.addEmployee(e1);

System.***out***.println("Inserted Employee with ID: " + id);

}

}

**Employee.java:**

package com.example.hibernate;

public class Employee {

private int id;

private String firstName;

private String lastName;

private float salary;

public int getId() { return id; }

public void setId(int id) { this.id = id; }

public String getFirstName() { return firstName; }

public void setFirstName(String firstName) { this.firstName = firstName; }

public String getLastName() { return lastName; }

public void setLastName(String lastName) { this.lastName = lastName; }

public float getSalary() { return salary; }

public void setSalary(float salary) { this.salary = salary; }

*@Override*

public String toString() {

return "Employee [id=" + id + ", firstName=" + firstName +

", lastName=" + lastName + ", salary=" + salary + "]";

}

}

**EmployeeDAO.java:**

package com.example.hibernate;

import org.hibernate.\*;

import org.hibernate.cfg.Configuration;

public class EmployeeDAO {

private static SessionFactory *factory* = new Configuration().configure().buildSessionFactory();

public Integer addEmployee(Employee emp) {

Session session = *factory*.openSession();

Transaction tx = null;

Integer id = null;

try {

tx = session.beginTransaction();

id = (Integer) session.save(emp);

tx.commit();

} catch (HibernateException e) {

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

e.printStackTrace();

} finally {

session.close();

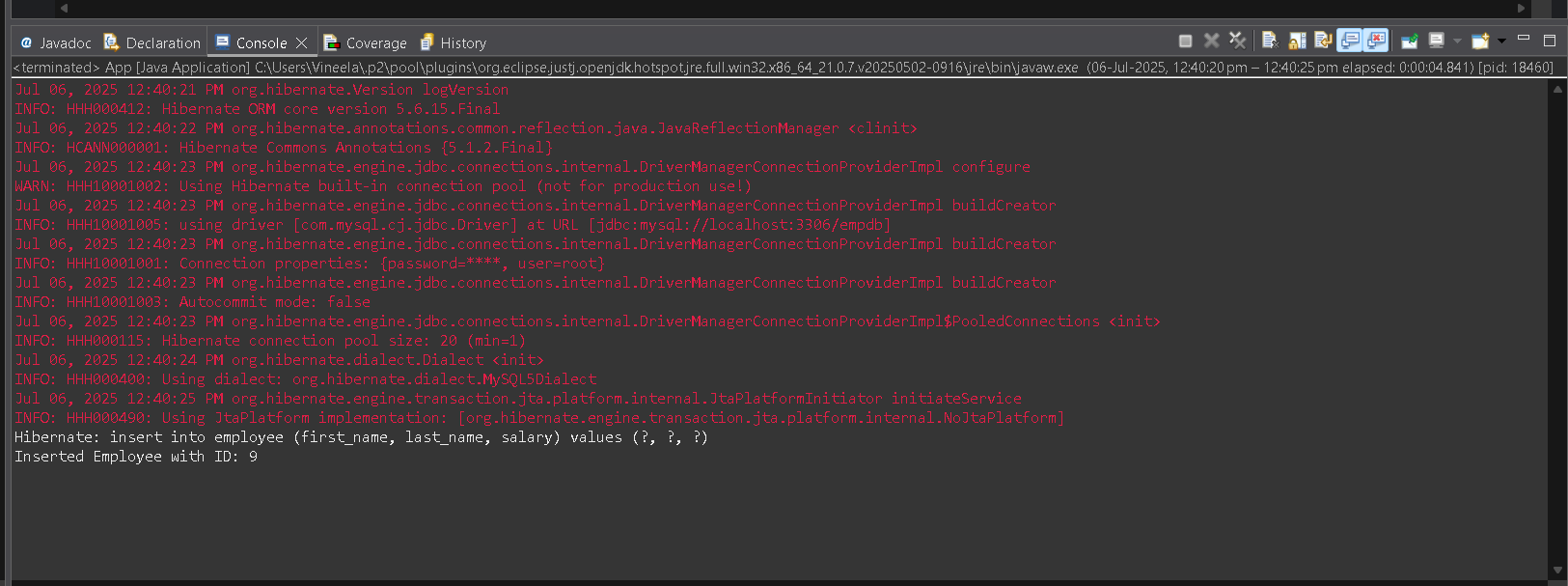
}

return id;

}

}

**OUTPUT:**



**2.jpa-standard-demo:**

**Persistence.xml:**

<?xml version="1.0" encoding="UTF-8"?>

<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence

http://xmlns.jcp.org/xml/ns/persistence/persistence\_2\_2.xsd"

version="2.2">

<persistence-unit name="employeePU" transaction-type="RESOURCE\_LOCAL">

<provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>

<class>com.example.jpa.Employee</class>

<properties>

<property name="javax.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>

<property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/empdb"/>

<property name="javax.persistence.jdbc.user" value="root"/>

<property name="javax.persistence.jdbc.password" value="1234"/>

<property name="hibernate.dialect" value="org.hibernate.dialect.MySQL5Dialect"/>

<property name="hibernate.hbm2ddl.auto" value="update"/>

<property name="hibernate.show\_sql" value="true"/>

</properties>

</persistence-unit>

</persistence>

**App.java:**

package com.example.jpa;

public class App {

public static void main(String[] args) {

Employee emp = new Employee();

emp.setFirstName("Sree");

emp.setLastName("Ram");

emp.setSalary(50000);

EmployeeService service = new EmployeeService();

service.saveEmployee(emp);

service.close();

}

}

**Employee.java:**

package com.example.jpa;

import javax.persistence.\*;

*@Entity*

*@Table*(name = "employee")

public class Employee {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private int id;

private String firstName;

private String lastName;

private double salary;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

}

**EmployeeService.java:**

package com.example.jpa;

import javax.persistence.\*;

public class EmployeeService {

private EntityManagerFactory emf;

private EntityManager em;

public EmployeeService() {

emf = Persistence.*createEntityManagerFactory*("employeePU");

em = emf.createEntityManager();

}

public void saveEmployee(Employee emp) {

EntityTransaction tx = em.getTransaction();

try {

tx.begin();

em.persist(emp);

tx.commit();

System.***out***.println("Employee saved: " + emp.getId());

} catch (Exception e) {

if (tx.isActive()) tx.rollback();

e.printStackTrace();

}

}

public void close() {

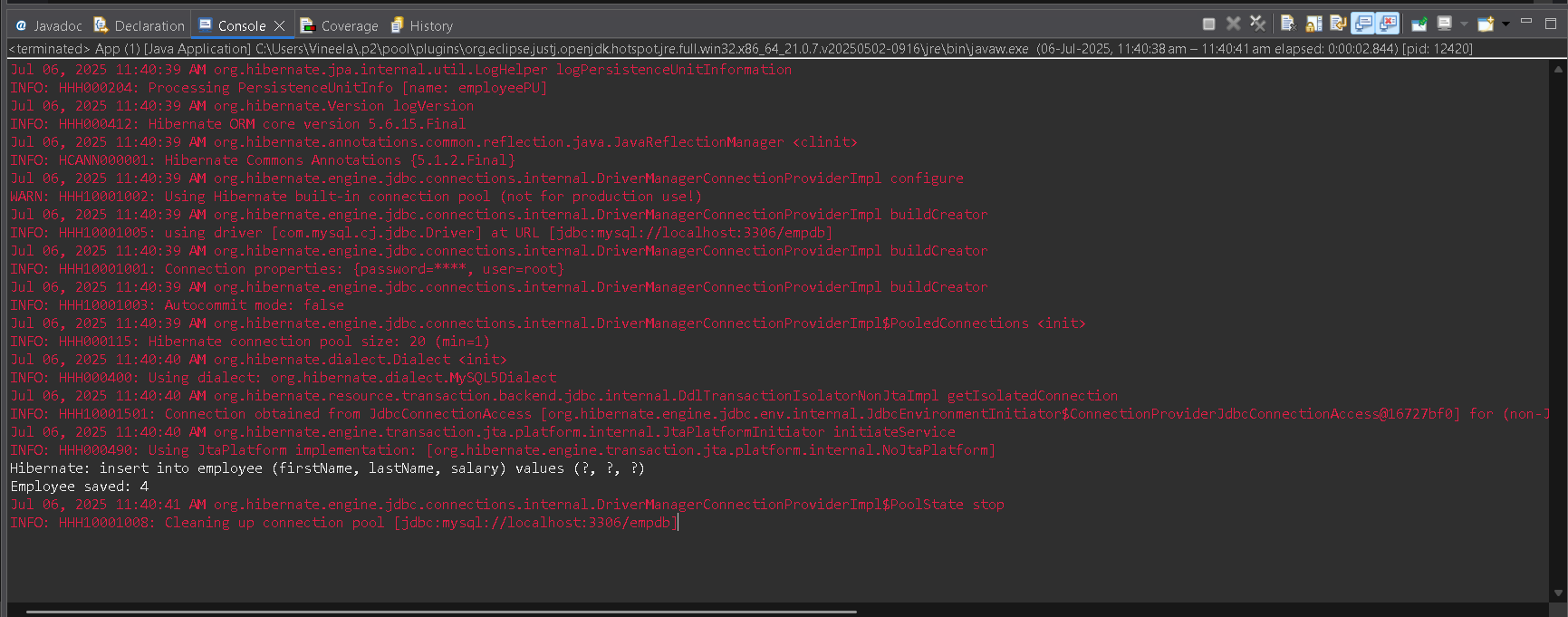
em.close();

emf.close();

}

}

**OUTPUT:**



**3.Spring-data-jpa-demo:**

**Application.properties:**

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

spring.datasource.username=root

spring.datasource.password=1234

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

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

**SpringDataJpaApplication:**

package com.example.springdata;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringDataJpaApplication {

public static void main(String[] args) {

SpringApplication.*run*(SpringDataJpaApplication.class, args);

}

}

**Employee.java:**

package com.example.springdata.entity;

import javax.persistence.\*;

*@Entity*

public class Employee {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private int id;

private String firstName;

private String lastName;

private double salary;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

}

**EmployeeRepository.java:**

package com.example.springdata.repository;

import com.example.springdata.entity.Employee;

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

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**AppRunner.java:**

package com.example.springdata.runner;

import com.example.springdata.entity.Employee;

import com.example.springdata.repository.EmployeeRepository;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

*@Component*

public class AppRunner implements CommandLineRunner {

*@Autowired*

private EmployeeRepository employeeRepository;

*@Override*

public void run(String... args) {

Employee emp = new Employee();

emp.setFirstName("Ravi");

emp.setLastName("Kumar");

emp.setSalary(50000);

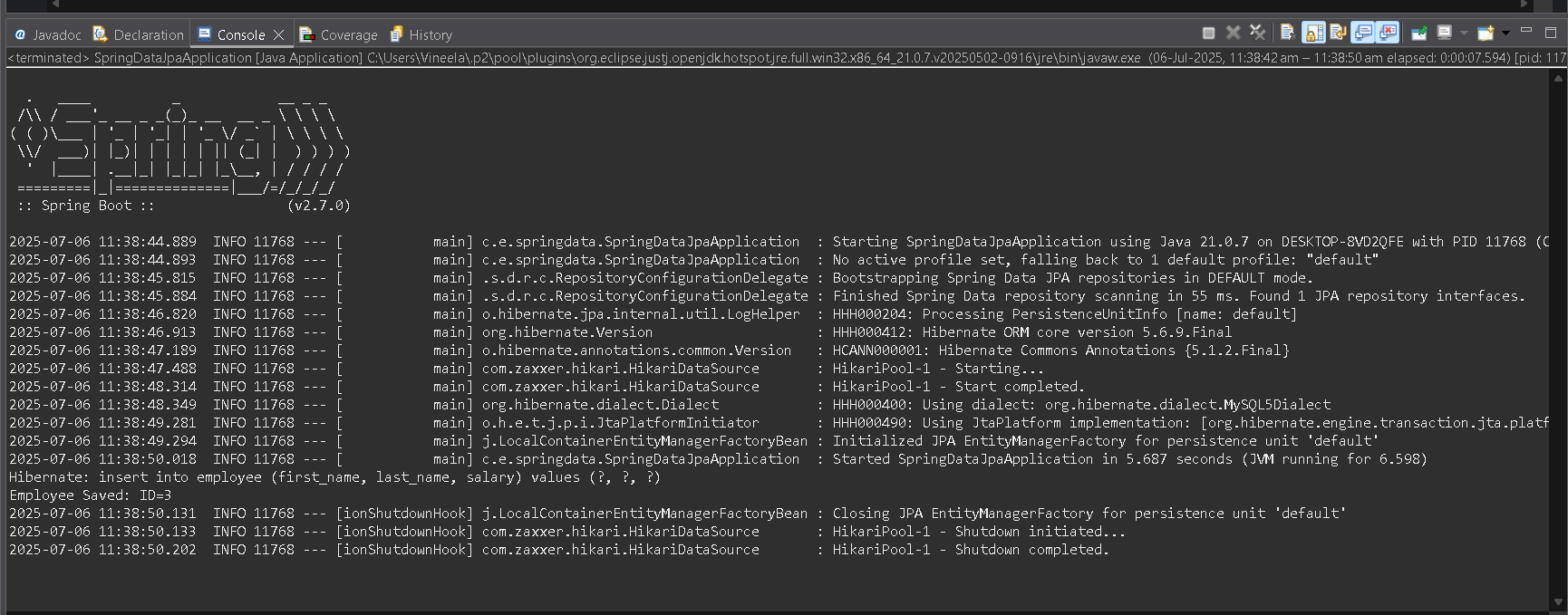
employeeRepository.save(emp);

System.***out***.println("Employee Saved: ID=" + emp.getId());

}

}

**OUTPUT:**



**Exercise-5: Implement services for managing Country**

**Application.properties:**

spring.application.name=orm-learn

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=trace

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

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger**{25}** %25M %4L %m%n

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=1234

spring.jpa.hibernate.ddl-auto=validate

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

spring.jpa.hibernate.ddl-auto=validate

**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*(scanBasePackages = "com.cognizant.ormlearn")

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);

***LOGGER***.info("Inside main");

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

*testGetAllCountries*();

*testFindCountryByCode*();

*testAddCountry*();

*testUpdateCountry*();

*testDeleteCountry*();

*testFindCountriesByPartialName*();

}

private static void testGetAllCountries() {

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

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

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

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

}

private static void testFindCountryByCode() {

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

Country country = *countryService*.findCountryByCode("IN");

***LOGGER***.debug("Country: {}", country);

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

}

private static void testAddCountry() {

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

Country newCountry = new Country();

newCountry.setCode("ZZ");

newCountry.setName("Zootopia");

*countryService*.addCountry(newCountry);

***LOGGER***.debug("Added Country: {}", *countryService*.findCountryByCode("ZZ"));

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

}

private static void testUpdateCountry() {

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

Country country = *countryService*.findCountryByCode("ZZ");

country.setName("Zootopia Updated");

*countryService*.updateCountry(country);

***LOGGER***.debug("Updated Country: {}", *countryService*.findCountryByCode("ZZ"));

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

}

private static void testDeleteCountry() {

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

*countryService*.deleteCountry("ZZ");

Country deleted = *countryService*.findCountryByCode("ZZ");

***LOGGER***.debug("Deleted Country: {}", deleted);

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

}

private static void testFindCountriesByPartialName() {

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

List<Country> result = *countryService*.findCountriesByPartialName("land");

***LOGGER***.debug("Matching Countries: {}", result);

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

}

}

**Country.java:**

package com.cognizant.ormlearn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Table;

import jakarta.persistence.Id;

import jakarta.persistence.Column;

*@*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.ormlearn.repository;

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

import org.springframework.stereotype.Repository;

import com.cognizant.ormlearn.model.Country;

import java.util.List;

*@Repository*

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContainingIgnoreCase(String name);

}

**CountryService.java:**

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.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();

}

public Country findCountryByCode(String code) {

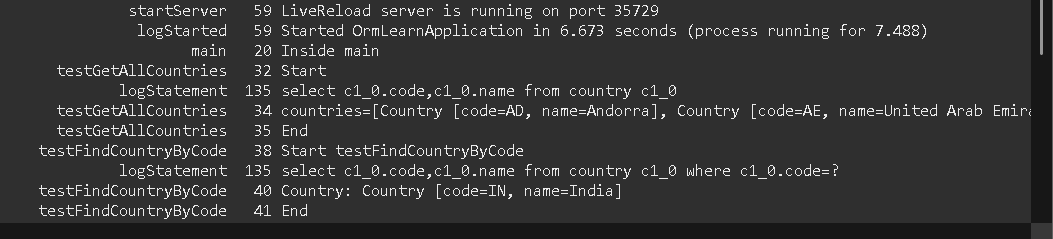
return countryRepository.findById(code)

.orElseThrow(() -> new RuntimeException("Country not found with code: " + code));

}

}

**OUTPUT:**



**Exercise-6: Find a country based on country code**

**OrmLearnApplication.java:**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ConfigurableApplicationContext;

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) throws CountryNotFoundException {

ConfigurableApplicationContext context = SpringApplication.*run*(OrmLearnApplication.class, args);

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

***LOGGER***.info("Inside main");

*testGetAllCountries*();

*testFindCountryByCode*();

}

private static void testGetAllCountries() {

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

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

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

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

}

private static void testFindCountryByCode() throws CountryNotFoundException {

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

Country country = *countryService*.findCountryByCode("IN");

***LOGGER***.debug("Country: {}", country);

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

}

}

CountryNotFoundException.java:

package com.cognizant.ormlearn.service.exception;

public class CountryNotFoundException extends Exception {

public CountryNotFoundException(String message) {

super(message);

}

}

**CountryRepository.java:**

package com.cognizant.ormlearn.repository;

import com.cognizant.ormlearn.model.Country;

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

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService.java:**

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.Optional;

import java.util.List;

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

*@Transactional*

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(countryCode);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found with code: " + countryCode);

}

return result.get();

}

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**Country.java:**

package com.cognizant.ormlearn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Table;

import jakarta.persistence.Id;

import jakarta.persistence.Column;

*@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*

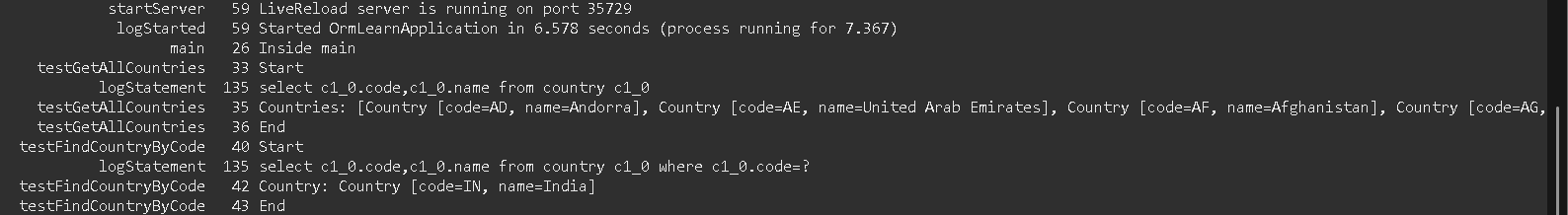
public String toString() {

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

}

}

OUTPUT:



**Exercise-7:** **Add a new country**

**OrmLearnApplication.java:**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class OrmLearnApplication {

private static CountryService countryService;

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

public static void main(String[] args) {

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

LOGGER.info("Inside main");

countryService = context.getBean(CountryService.class);

testGetAllCountries();

testFindCountryByCode();

testAddCountry(); // <-- add this if testing adding new country

}

private static void testGetAllCountries() {

LOGGER.info("Start");

for (Country country : countryService.getAllCountries()) {

LOGGER.debug("Country: {}", country);

}

LOGGER.info("End");

}

private static void testFindCountryByCode() {

LOGGER.info("Start");

try {

Country country = countryService.findCountryByCode("IN");

LOGGER.debug("Country: {}", country);

} catch (CountryNotFoundException e) {

LOGGER.error("Exception: {}", e.getMessage());

}

LOGGER.info("End");

}

private static void testAddCountry() {

LOGGER.info("Start");

Country country = new Country();

country.setCode("XY");

country.setName("Xyland");

countryService.addCountry(country);

try {

Country addedCountry = countryService.findCountryByCode("XY");

LOGGER.debug("Added Country: {}", addedCountry);

} catch (CountryNotFoundException e) {

LOGGER.error("Country not found: {}", e.getMessage());

}

LOGGER.info("End");

}

}

**CountryService.java:**

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(countryCode);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found with code: " + countryCode);

}

return result.get();

}

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

@Transactional

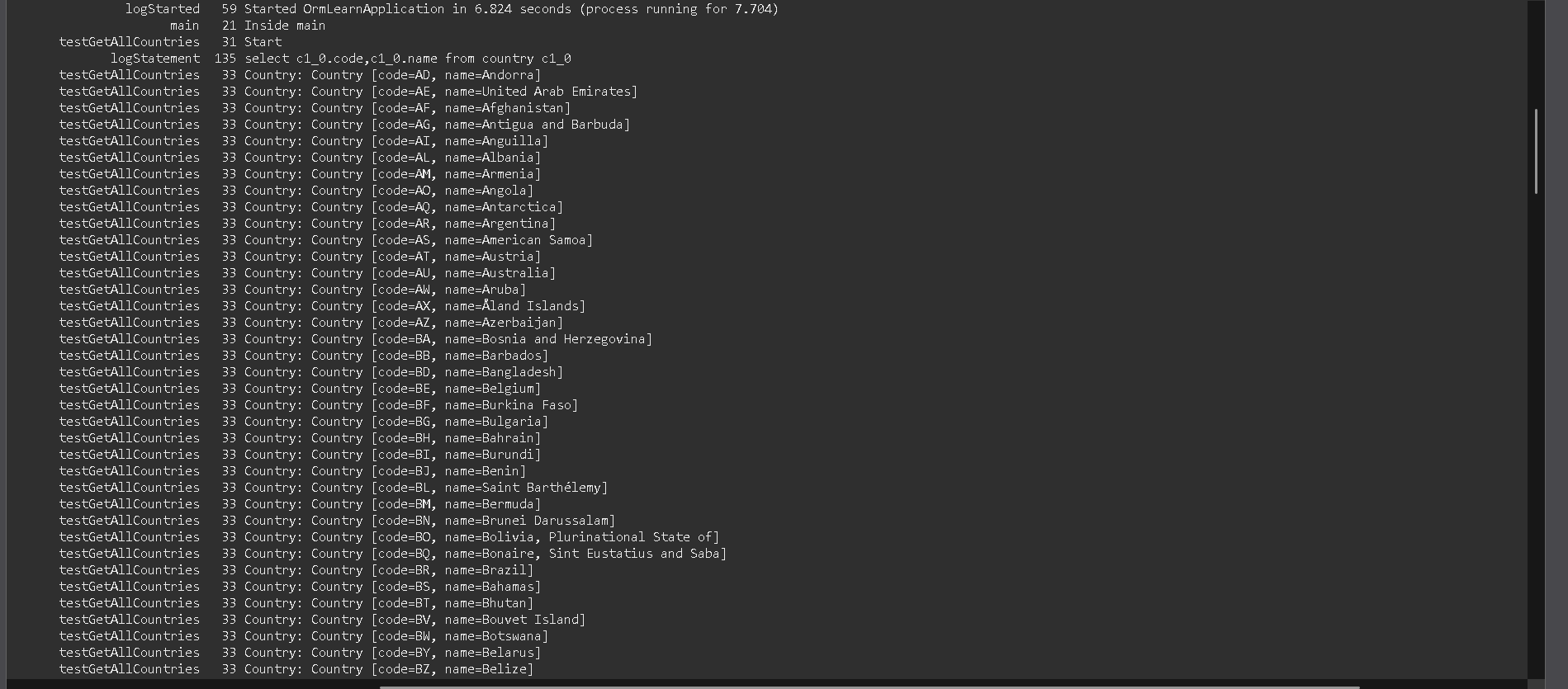
public void addCountry(Country country) {

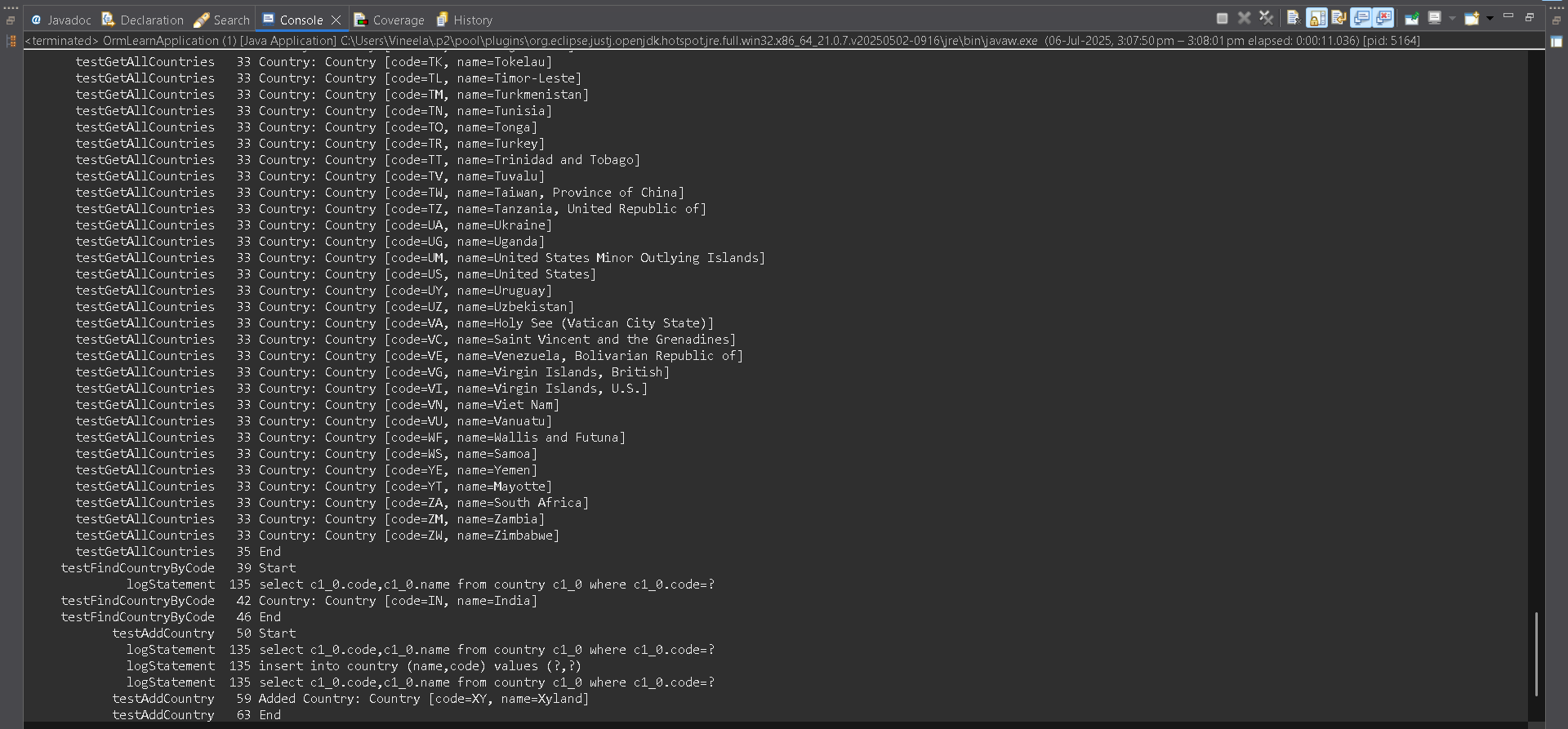
countryRepository.save(country);

}

}

OUTPUT:





**Exercise-8:Update a country based on Code**

**CountryService.java:**

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.Optional;

import java.util.List;

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

*@Transactional*

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(countryCode);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found with code: " + countryCode);

}

return result.get();

}

*@Transactional*

public void addCountry(Country country) {

countryRepository.save(country);

}

*@Transactional*

public void updateCountry(String code, String name) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(code);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found with code: " + code);

}

Country country = result.get();

country.setName(name); // update the name

countryRepository.save(country); // persist the change

}

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**OrmLearnApplication.java:**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

*@SpringBootApplication*

public class OrmLearnApplication {

private static CountryService *countryService*;

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

public static void main(String[] args) {

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

***LOGGER***.info("Inside main");

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

*testGetAllCountries*();

*testFindCountryByCode*();

*testAddCountry*();

*testUpdateCountry*();

}

private static void testGetAllCountries() {

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

for (Country country : *countryService*.getAllCountries()) {

***LOGGER***.debug("Country: {}", country);

}

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

}

private static void testFindCountryByCode() {

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

try {

Country country = *countryService*.findCountryByCode("IN");

***LOGGER***.debug("Country: {}", country);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Exception: {}", e.getMessage());

}

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

}

private static void testAddCountry() {

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

Country country = new Country();

country.setCode("XY");

country.setName("Xyland");

*countryService*.addCountry(country);

try {

Country addedCountry = *countryService*.findCountryByCode("XY");

***LOGGER***.debug("Added Country: {}", addedCountry);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Country not found: {}", e.getMessage());

}

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

}

private static void testUpdateCountry() {

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

try {

*countryService*.updateCountry("XY", "Xylandia"); // Update country code XY to new name

Country updatedCountry = *countryService*.findCountryByCode("XY");

***LOGGER***.debug("Updated Country: {}", updatedCountry);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Country not found: {}", e.getMessage());

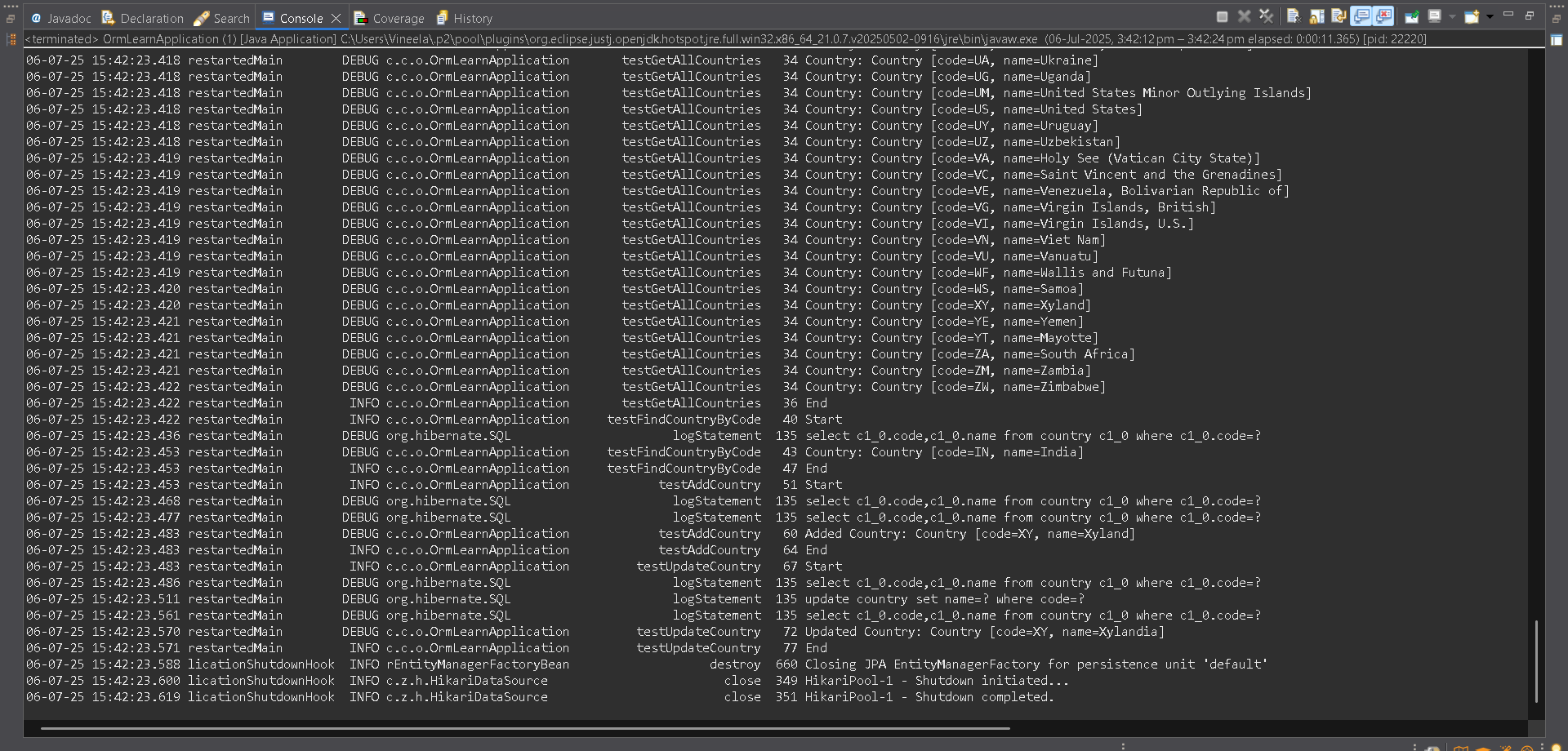
}

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

}

}

**OUTPUT:**



**Exercise-9:Delete a country based on code**

**CountryService.java:**

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.Optional;

import java.util.List;

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

*@Transactional*

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(countryCode);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found with code: " + countryCode);

}

return result.get();

}

*@Transactional*

public void addCountry(Country country) {

countryRepository.save(country);

}

*@Transactional*

public void deleteCountry(String code) throws CountryNotFoundException {

if (!countryRepository.existsById(code)) {

throw new CountryNotFoundException("Country not found with code: " + code);

}

countryRepository.deleteById(code);

}

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**OrmLearnApplication.java:**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

import com.cognizant.ormlearn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

*@SpringBootApplication*

public class OrmLearnApplication {

private static CountryService *countryService*;

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

public static void main(String[] args) {

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

***LOGGER***.info("Inside main");

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

*testFindCountryByCode*();

*testAddCountry*();

*testDeleteCountry*();

}

private static void testFindCountryByCode() {

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

try {

Country country = *countryService*.findCountryByCode("IN");

***LOGGER***.debug("Country: {}", country);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Exception: {}", e.getMessage());

}

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

}

private static void testAddCountry() {

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

Country country = new Country();

country.setCode("XY");

country.setName("Xyland");

*countryService*.addCountry(country);

try {

Country addedCountry = *countryService*.findCountryByCode("XY");

***LOGGER***.debug("Added Country: {}", addedCountry);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Country not found: {}", e.getMessage());

}

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

}

private static void testDeleteCountry() {

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

try {

*countryService*.deleteCountry("XY");

***LOGGER***.debug("Deleted country with code: XY");

*countryService*.findCountryByCode("XY");

} catch (CountryNotFoundException e) {

***LOGGER***.error("Country not found after delete (expected): {}", e.getMessage());

}

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

}

}

**OUTPUT:**

