**Spring Core and Maven**

**Exercise 1: Configuring a Basic Spring Application**

**LibraryManagement Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.Cognizant</groupId>

<artifactId>LibraryManagement</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>LibraryManagement</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>6.2.7</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**applicationContext.xml:**

<beans xmlns=*"http://www.springframework.org/schema/beans"*

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

xsi:schemaLocation=*"*

*http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"*>

<bean id=*"bookService"* class=*"com.library.service.BookService"*>

</bean>

<bean id=*"bookRepository"* class=*"com.library.repository.BookRepository"*>

</bean>

</beans>

**BookService.java:**

package com.library.service;

public class BookService {

public BookService() {

System.*out*.println("BookService object created");

}

}

**BookRepository.java:**

package com.library.repository;

public class BookRepository {

public BookRepository() {

System.*out*.println("BookRepository object Created");

}

}

**App.java**

package com.Cognizant.LibraryManagement;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

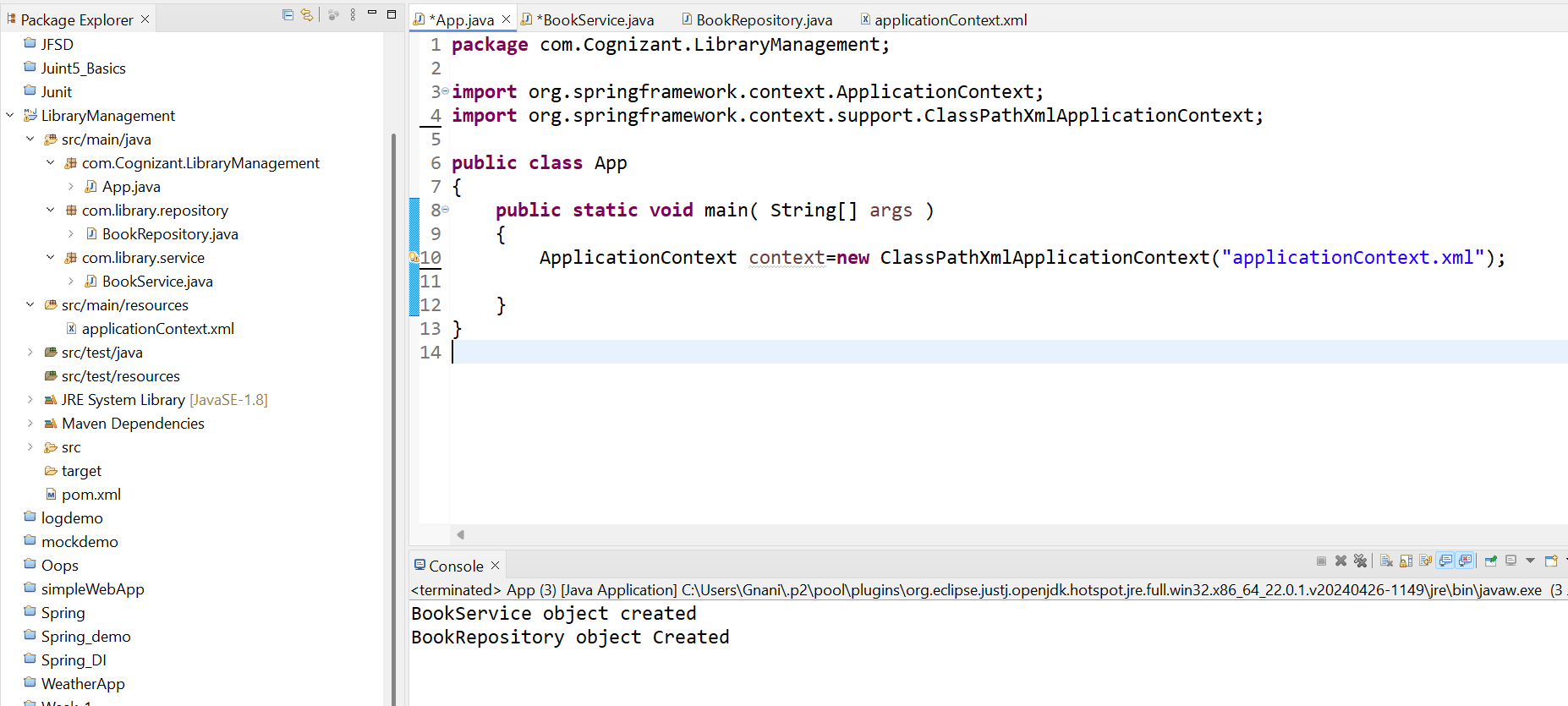
public class App {

public static void main( String[] args ) {

ApplicationContext context=new ClassPathXmlApplicationContext("applicationContext.xml");

}}

**Output:**

****

**Exercise 2: Implementing Dependency Injection**

**applicationContext.xml:**

<beans xmlns=*"http://www.springframework.org/schema/beans"*

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

xsi:schemaLocation=*"*

*http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"*>

<bean id=*"bookService"* class=*"com.library.service.BookService"*>

<property name=*"bookRepository"* ref=*"bookRepository"*/>

</bean>

<bean id=*"bookRepository"* class=*"com.library.repository.BookRepository"*>

</bean>

</beans>

**BookRepository.java:**

package com.library.repository;

public class BookRepository {

public BookRepository() {

System.*out*.println("BookRepository object Created");

}

public void repository() {

System.*out*.println("Book Repository Method ");

}

}

**BookService.java:**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

public BookService() {

System.*out*.println("BookService object created");

}

private BookRepository bookRepository;

public void service() {

bookRepository.repository();

System.*out*.println("Book service Method");

}

public BookRepository getBookRepository() {

return bookRepository;

}

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

}

**App.java:**

package com.Cognizant.LibraryManagement;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class App {

public static void main( String[] args ){

ApplicationContext context=new ClassPathXmlApplicationContext("applicationContext.xml");

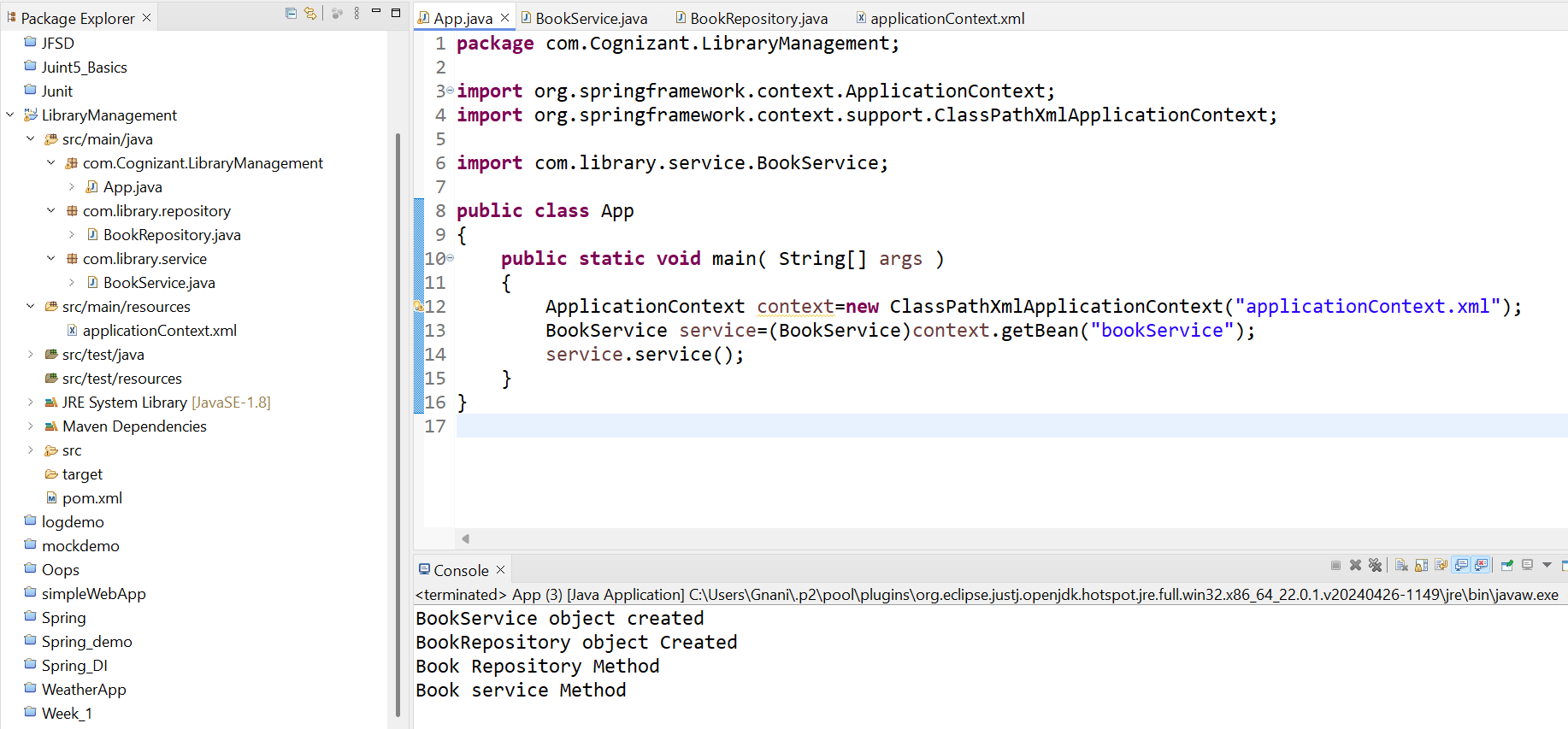
BookService service=(BookService)context.getBean("bookService");

service.service();

}

}

**Output:**



**Exercise 4: Creating and Configuring a Maven Project**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

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

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.library</groupId>

<artifactId>LibraryManagement</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>jar</packaging>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.33</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.33</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.33</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**App.java:**

package com.Cognizant.LibraryManagement;

public class App {

public static void main( String[] args ){

System.*out*.println("Library Management");

}

}

**Spring Data JPA - Quick Example**

**Country.java:**

package com.cognizant.ormlearn.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 = "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 + "]";

}

}

**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();

}

}

**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> {

}

**applications.properties:**

spring.application.name=orm-learn

# Logging

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=trace

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

# PostgreSQL DB Config

spring.datasource.url=jdbc:postgresql://localhost:5433/ormlearn

spring.datasource.username=postgres

spring.datasource.password=Postgresql

spring.datasource.driver-class-name=org.postgresql.Driver

# Hibernate

spring.jpa.hibernate.ddl-auto=validate

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

**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();

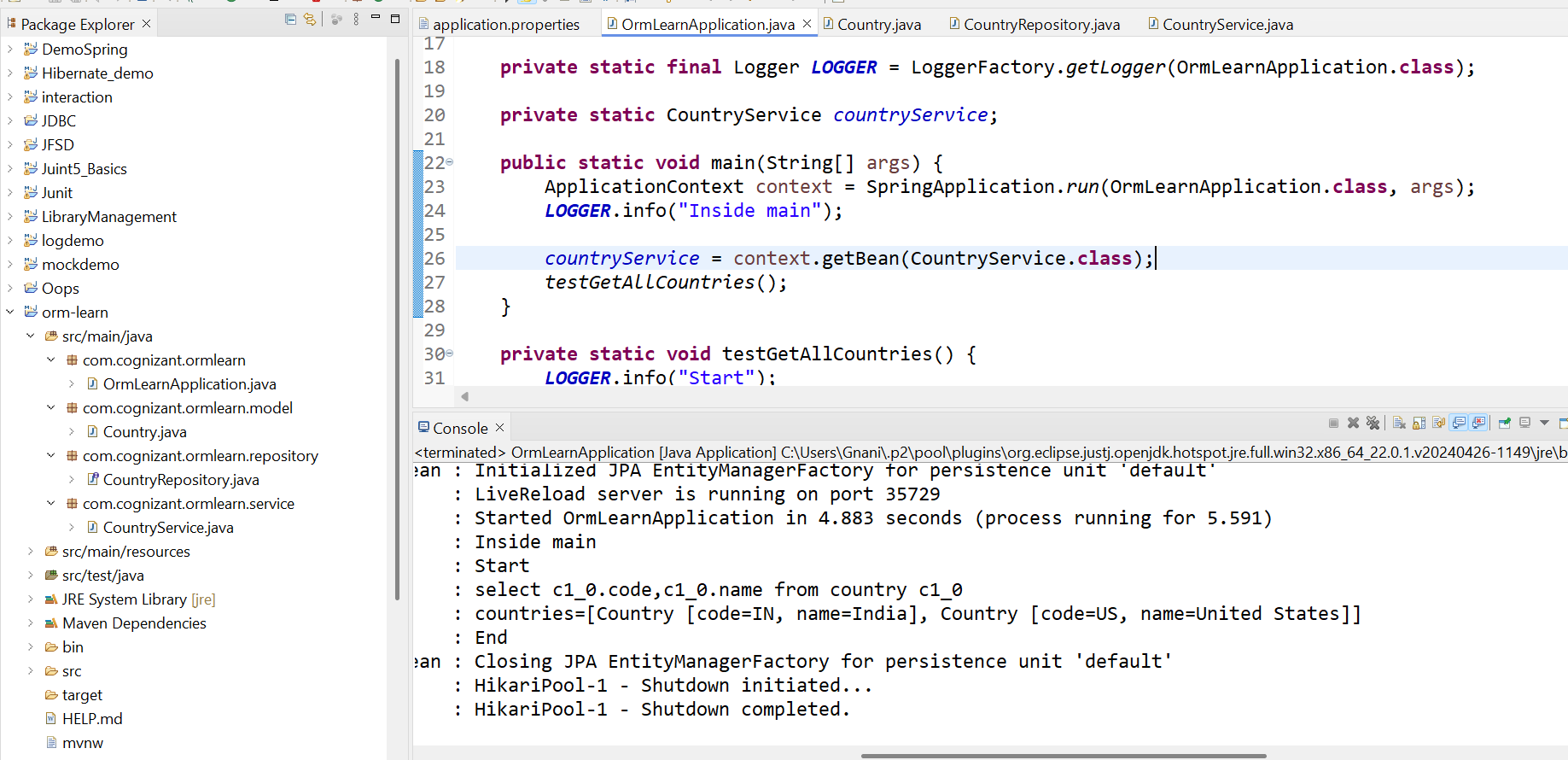
*LOGGER*.debug("countries={}", countries);

*LOGGER*.info("End");

}

}

**Output:**

****

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

**Employee.java:**

package com.cognizant.hibernate\_employee\_demo;

import jakarta.persistence.\*;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.*IDENTITY*)

private int id;

private String name;

private double salary;

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 double getSalary() { return salary; }

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

@Override

public String toString() {

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

}

}

**HibernateUtil.java:**

package com.cognizant.hibernate\_employee\_demo;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

public class HibernateUtil {

private static final SessionFactory *sessionFactory* = *buildSessionFactory*();

private static SessionFactory buildSessionFactory() {

try {

return new Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

} catch (Throwable ex) {

throw new ExceptionInInitializerError("Initial SessionFactory creation failed: " + ex);

}

}

public static SessionFactory getSessionFactory() {

return *sessionFactory*;

}

}

**hibernate.cfg.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>org.postgresql.Driver</property>

<property name=*"hibernate.connection.url"*>jdbc:postgresql://localhost:5433/ormlearn</property>

<property name=*"hibernate.connection.username"*>postgres</property>

<property name=*"hibernate.connection.password"*>Postgresql</property>

<property name=*"hibernate.dialect"*>org.hibernate.dialect.PostgreSQLDialect</property>

<property name=*"hibernate.hbm2ddl.auto"*>validate</property>

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

<mapping class=*"com.cognizant.hibernate\_employee\_demo.Employee"*/>

</session-factory>

</hibernate-configuration>

**App.java:**

package com.cognizant.hibernate\_employee\_demo;

import org.hibernate.Session;

import org.hibernate.Transaction;

public class App {

public static void main(String[] args) {

*addEmployee*("Classic Hibernate", 45000.0);

}

public static void addEmployee(String name, double salary) {

Session session = HibernateUtil.*getSessionFactory*().openSession();

Transaction tx = null;

try {

tx = session.beginTransaction();

Employee emp = new Employee();

emp.setName(name);

emp.setSalary(salary);

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

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

tx.commit();

} catch (Exception e) {

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

e.printStackTrace();

} finally {

session.close();

}

}

}

**Output:**

