WEEK-3

SPRING CORE AND MAVEN

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

**CODE:**

applicationContent.xml

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

<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"*>

<!-- Basic Beans (no injection) -->

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

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

</beans>

BookRepository.java

**package** com.library.repository;

**public** **class** BookRepository {

**public** **void** saveBook(String bookName) {

System.***out***.println("Book '" + bookName + "' saved to the repository.");

}

}

BookService.java

**package** com.library.service;

**import** com.library.repository.BookRepository;

**public** **class** BookService {

**private** BookRepository bookRepository;

**public** **void** addBook(String bookName) {

**if** (bookRepository != **null**) {

System.***out***.println("Adding book: " + bookName);

bookRepository.saveBook(bookName);

} **else** {

System.***out***.println("Repository not set.");

}

}

**public** **void** setBookRepository(BookRepository bookRepository) {

**this**.bookRepository = bookRepository;

}

}

MainApp.java

**package** org.library.app;

**import** com.library.repository.BookRepository;

**import** com.library.service.BookService;

**public** **class** MainApp {

**public** **static** **void** main(String[] args) {

BookRepository repo = **new** BookRepository();

BookService service = **new** BookService();

service.setBookRepository(repo); // manual injection

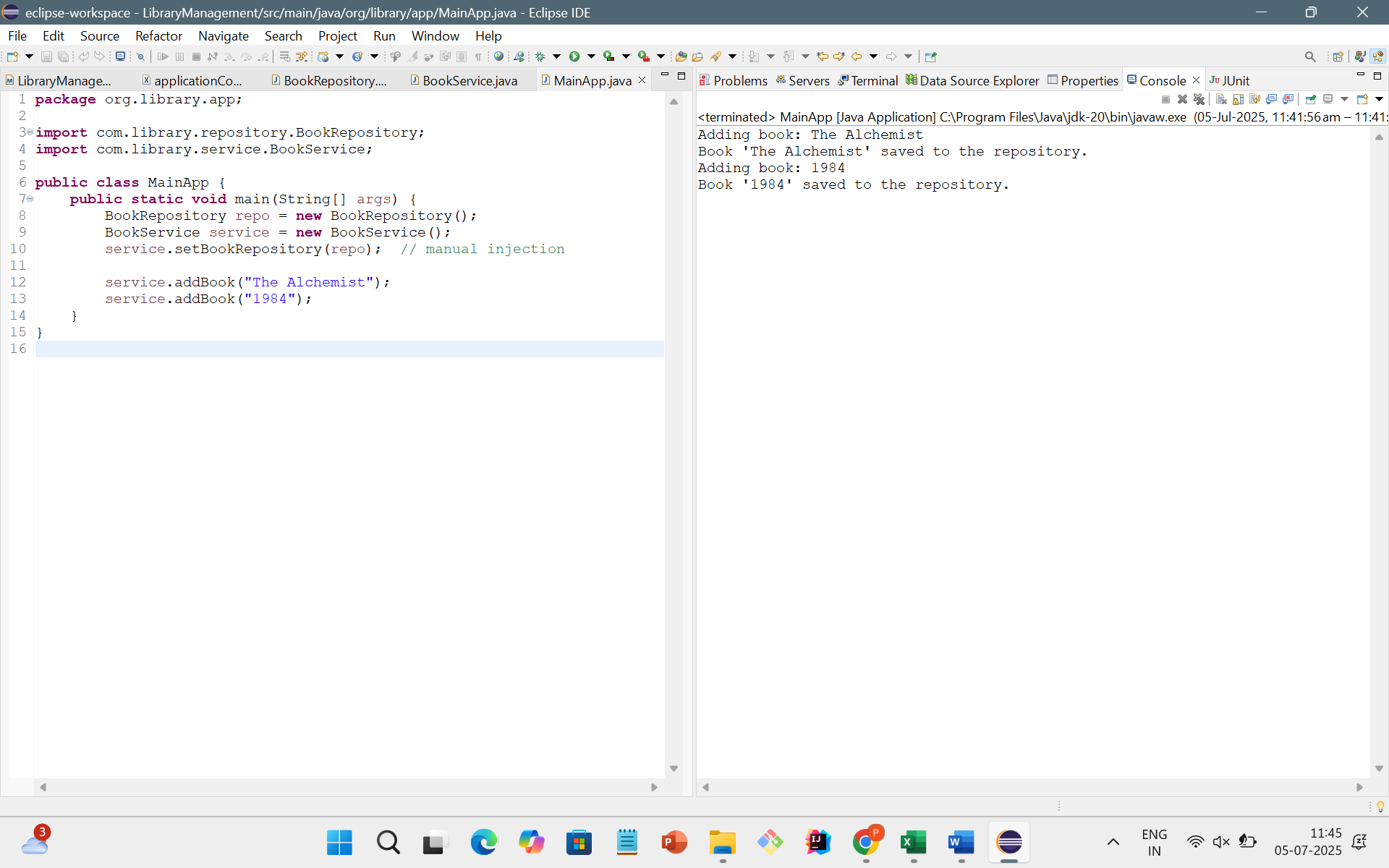
service.addBook("The Alchemist");

service.addBook("1984");

}

}

**OUTPUT:**



**Exercise 2: Implementing Dependency Injection**

**CODE:**

applicationContent.xml

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

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

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

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

</bean>

</beans>

BookRepository.java

**package** com.library.repository;

**public** **class** BookRepository {

**public** **void** saveBook(String bookName) {

System.***out***.println("Book '" + bookName + "' saved to the repository.");

}

}

BookService.java

**package** com.library.service;

**import** com.library.repository.BookRepository;

**public** **class** BookService {

**private** BookRepository bookRepository;

**public** **void** setBookRepository(BookRepository bookRepository) {

**this**.bookRepository = bookRepository;

}

**public** **void** addBook(String bookName) {

System.***out***.println("Adding book: " + bookName);

bookRepository.saveBook(bookName);

}

}

Mainapp.java

**package** org.library.app;

**import** com.library.service.BookService;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** MainApp {

**public** **static** **void** main(String[] args) {

// Load Spring application context

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

BookService bookService = context.getBean("bookService", BookService.**class**);

bookService.addBook("The Alchemist");

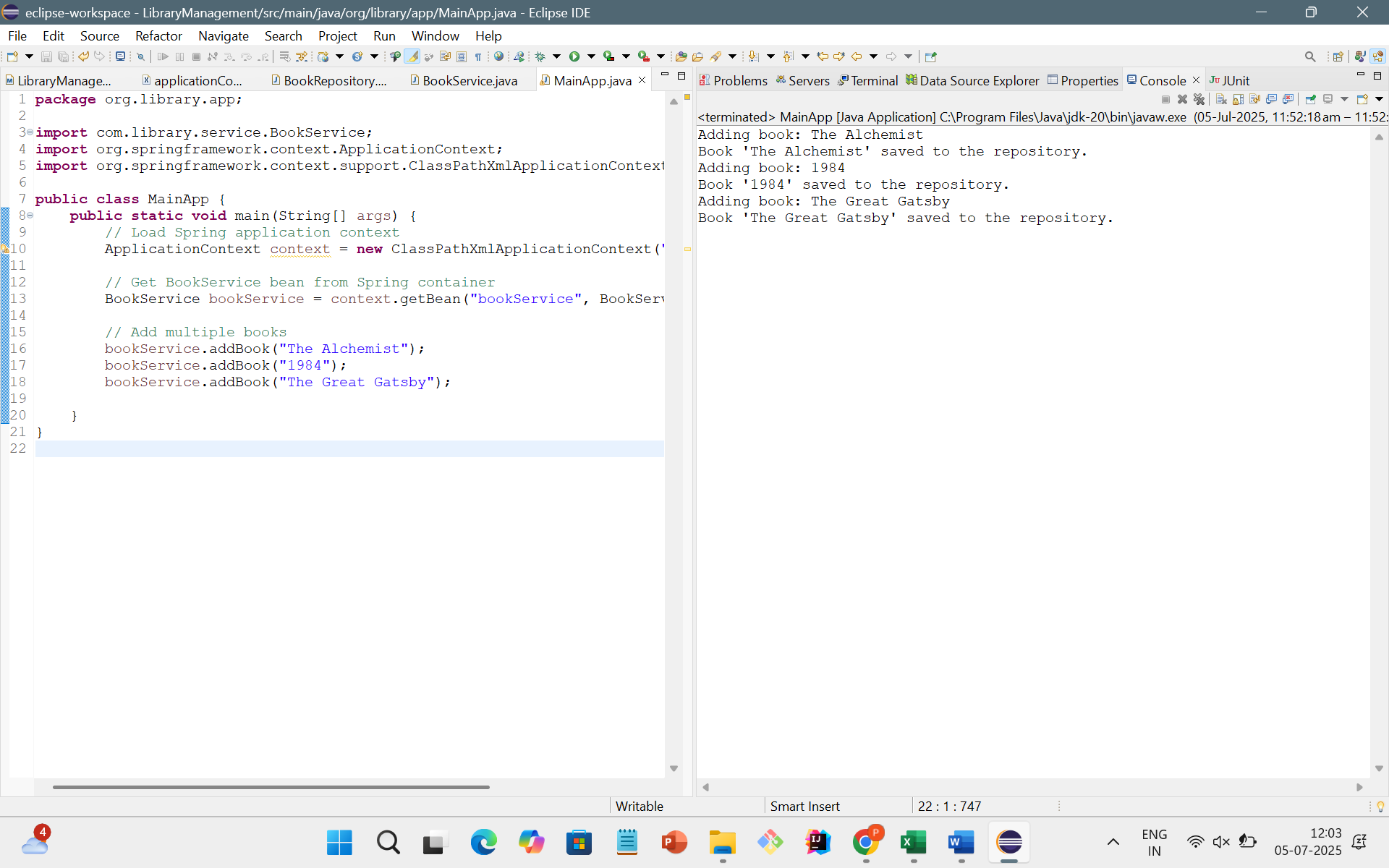
bookService.addBook("1984");

bookService.addBook("The Great Gatsby");

}

}

**OUTPUT:**



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

**CODE:**

**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.library</groupId>

<artifactId>LibraryManagement</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.36</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.36</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.36</version>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Java Compiler Settings -->

<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>

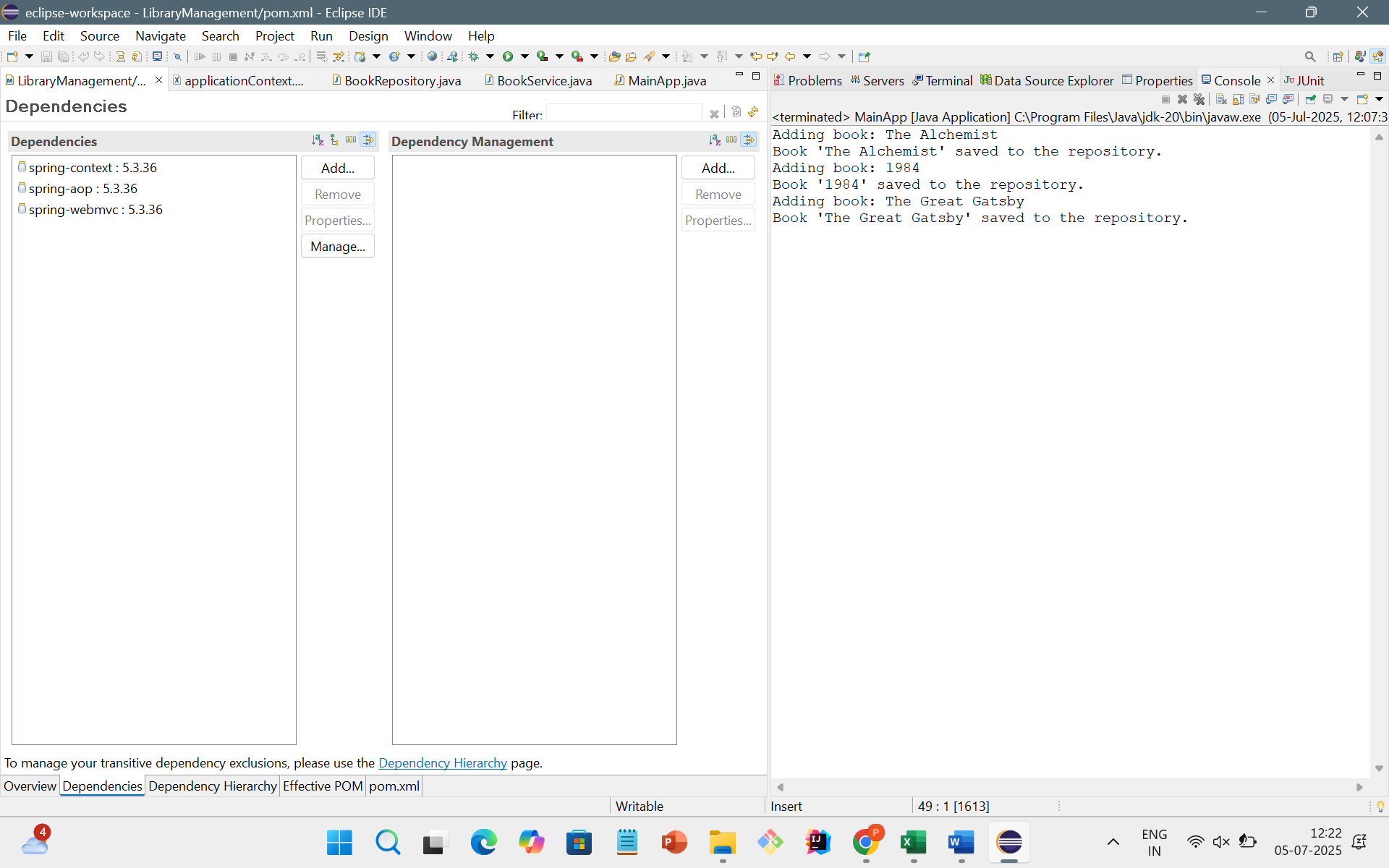
</plugin>

</plugins>

</build>

</project>

**OUTPUT:**



**SPRING DATA JPA WITH SPRING BOOT,HIBERNATE**

**Hands on 1**

**Spring Data JPA - Quick Example**

**CODE:**

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 = "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 + "]";

}

}

CountryService.java

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Country;

import java.util.List;

public interface CountryService {

List<Country> getAllCountries();

}

CountryRepository.java

package com.cognizant.ormlearn.repository;

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

import com.cognizant.ormlearn.model.Country;

public interface CountryRepository extends JpaRepository<Country, String> {

}

OrmLearnApplications.java

**package** com.cognizant.ormlearn;

**import** com.cognizant.ormlearn.model.Country;

**import** com.cognizant.ormlearn.service.CountryService;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ApplicationContext;

**import** java.util.List;

@SpringBootApplication

**public** **class** OrmLearnApplication {

**private** **static** CountryService *countryService*;

**public** **static** **void** main(String[] args) {

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

System.***out***.println("===== Application Started =====");

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

*testGetAllCountries*();

System.***out***.println("===== Application Finished =====");

}

**private** **static** **void** testGetAllCountries() {

System.***out***.println("\nFetching all countries from the database...");

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

System.***out***.println("\n--- Country List ---");

**for** (Country country : countries) {

System.***out***.println("Code: " + country.getCode() + ", Name: " + country.getName());

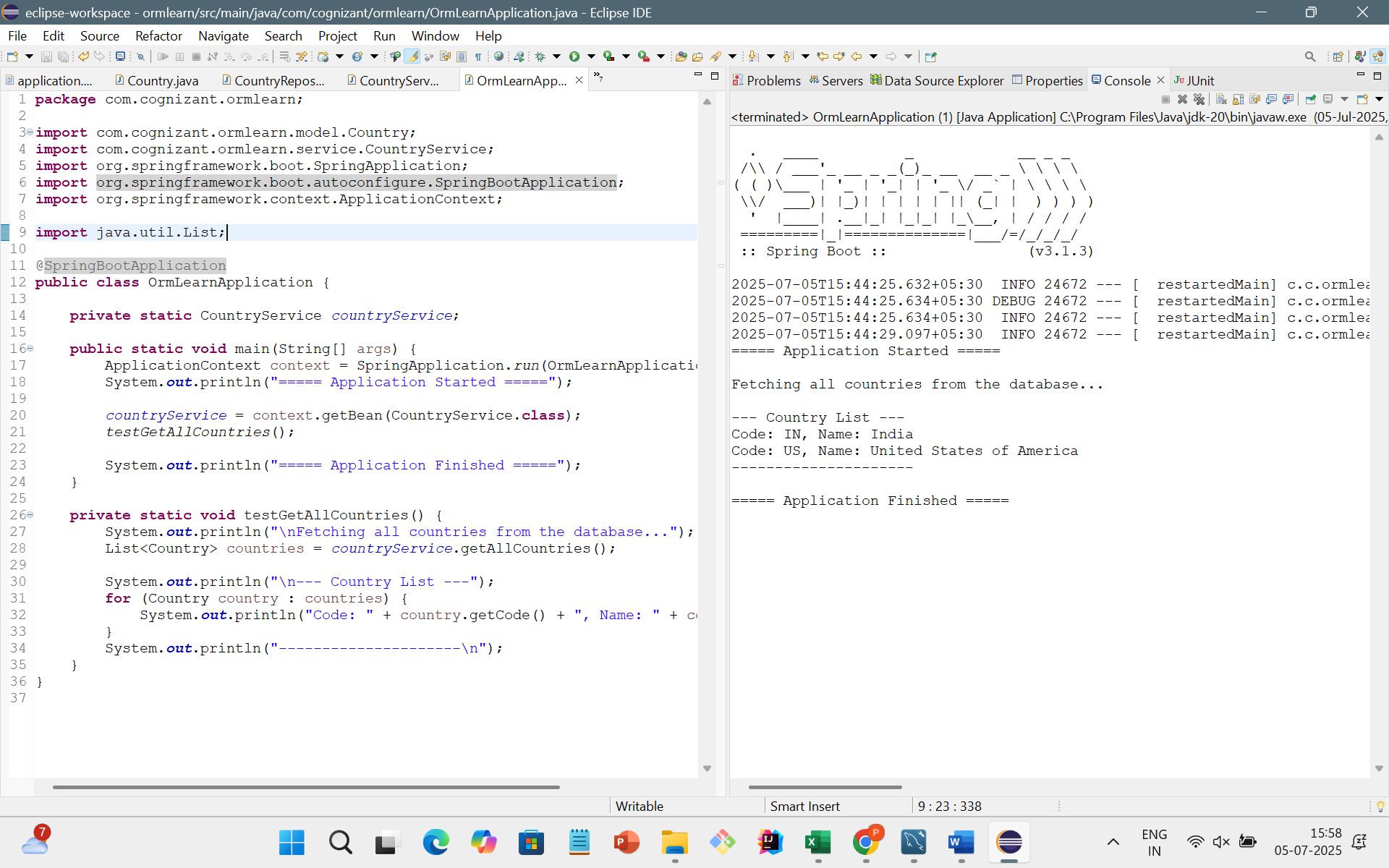
}

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

}

}

**OUTPUT:**



**Hands on 4**

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

**CODE:**

Employee.java

**package** com.cognizant.ormlearn.model;

**import** jakarta.persistence.Entity;

**import** jakarta.persistence.Id;

**import** jakarta.persistence.GeneratedValue;

**import** jakarta.persistence.GenerationType;

**import** jakarta.persistence.Column;

**import** jakarta.persistence.Table;

@Entity

@Table(name = "employee")

**public** **class** Employee {

@Id

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

**private** **int** id;

@Column(name = "name") // Optional if column name is same as field name

**private** String name;

@Column(name = "department")

**private** String department;

@Column(name = "salary")

**private** **double** salary;

// Constructors

**public** Employee() {

}

**public** Employee(**int** id, String name, String department, **double** salary) {

**this**.id = id;

**this**.name = name;

**this**.department = department;

**this**.salary = salary;

}

// Getters and Setters

**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** String getDepartment() {

**return** department;

}

**public** **void** setDepartment(String department) {

**this**.department = department;

}

**public** **double** getSalary() {

**return** salary;

}

**public** **void** setSalary(**double** salary) {

**this**.salary = salary;

}

// toString (optional for logging/debugging)

@Override

**public** String toString() {

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

}

}

EmployeeRepository.java:

**package** com.cognizant.ormlearn.repository;

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

**import** org.springframework.stereotype.Repository;

**import** com.cognizant.ormlearn.model.Employee;

@Repository

**public** **interface** EmployeeRepository **extends** JpaRepository<Employee, Integer> {

// You can add custom query methods here if needed

}

EmployeeService.java

**package** com.cognizant.ormlearn.service;

**import** com.cognizant.ormlearn.model.Employee;

**import** com.cognizant.ormlearn.repository.EmployeeRepository;

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

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

**import** java.util.List;

**import** java.util.Optional;

@Service

**public** **class** EmployeeService {

@Autowired

**private** EmployeeRepository employeeRepository;

**public** Employee addEmployee(Employee employee) {

**return** employeeRepository.save(employee);

}

**public** List<Employee> getAllEmployees() {

**return** employeeRepository.findAll();

}

**public** Optional<Employee> getEmployeeById(**int** id) {

**return** employeeRepository.findById(id);

}

**public** **void** deleteEmployeeById(**int** id) {

employeeRepository.deleteById(id);

}

}

OrmLearnApplication.java

**package** com.cognizant.ormlearn;

**import** com.cognizant.ormlearn.model.Employee;

**import** com.cognizant.ormlearn.service.EmployeeService;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ApplicationContext;

**import** java.util.List;

@SpringBootApplication

**public** **class** OrmLearnApplication {

**private** **static** EmployeeService *employeeService*;

**public** **static** **void** main(String[] args) {

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

*employeeService* = context.getBean(EmployeeService.**class**);

System.***out***.println("===== Application Started =====");

*clearAllEmployees*(); //

**if** (*employeeService*.getAllEmployees().isEmpty()) {

*addTestEmployees*();

}

System.***out***.println("All Employees:");

List<Employee> employees = *employeeService*.getAllEmployees();

employees.forEach(System.***out***::println);

System.***out***.println("Employee with ID 13:");

*employeeService*.getEmployeeById(13).ifPresent(System.***out***::println);

System.***out***.println("===== Application Finished =====");

}

**private** **static** **void** clearAllEmployees() {

List<Employee> all = *employeeService*.getAllEmployees();

all.forEach(e -> *employeeService*.deleteEmployeeById(e.getId()));

System.***out***.println("All existing employees deleted.");

}

**private** **static** **void** addTestEmployees() {

System.***out***.println("Adding test employees...");

*employeeService*.addEmployee(**new** Employee(0, "Jane Smith", "HR", 45000));

*employeeService*.addEmployee(**new** Employee(0, "Alice Johnson", "Finance", 60000));

}

}

**OUTPUT:**

