**WEEK – 3 SPRING CORE MAVEN**

**Exercise 2: Implementing Dependency Injection**

**Scenario:**

In the library management application, managing the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

**Modify the XML Configuration:**

<?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 definition for BookRepository -->

<bean id="bookRepository" class="com.beginsecure.main.repository.BookRepository" />

<!-- Bean definition for BookService with DI via setter -->

<bean id="bookService" class="com.beginsecure.main.service.BookService">

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

</bean>

</beans>

**Update the BookService Class:**

package com.beginsecure.main.service;

import com.beginsecure.main.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void showBooks() {

bookRepository.displayBooks();

}

}

**Test the Configuration:**

package com.beginsecure.main;

import com.beginsecure.main.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

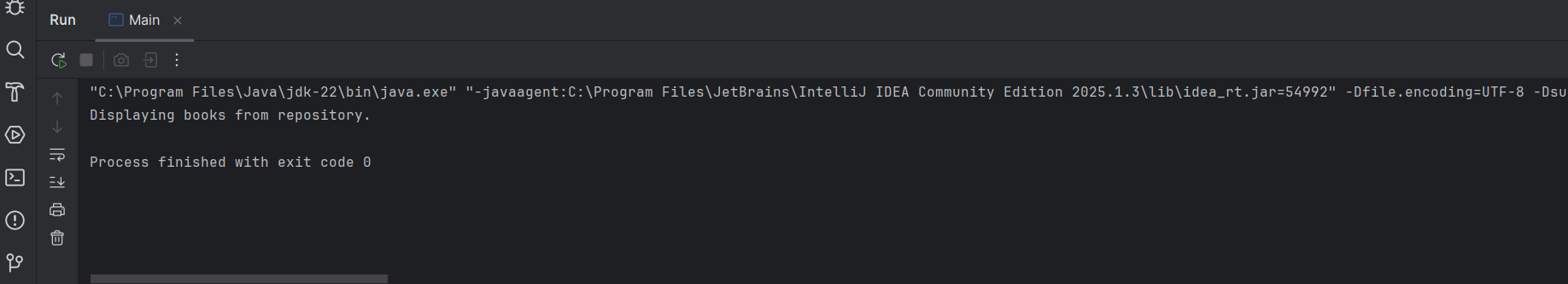
ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

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

service.showBooks(); // This will call BookRepository.displayBooks()

}

}

**Sample Out:**