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

**Scenario:**

Your company is developing a web application for managing a library. You need to use the Spring Framework to handle the backend operations.

**IMPLEMENTATION:**

**Pom.xml File:**

1.Adding dependency

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

**ApplicationContext.xml**

Location: src/main/resources

<?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) {

// **TODO** Auto-generated method stub

System.***out***.println("BookRepository: Saving book - " + bookName);

}

}

**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("BookService: Adding book - " + bookName);

bookRepository.saveBook(bookName);

}

}

**App.java**

package com.library.LibraryManagement;

import com.library.service.BookService;

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");

BookService bookService = (BookService) context.getBean("bookService"); bookService.addBook("Spring in Action");

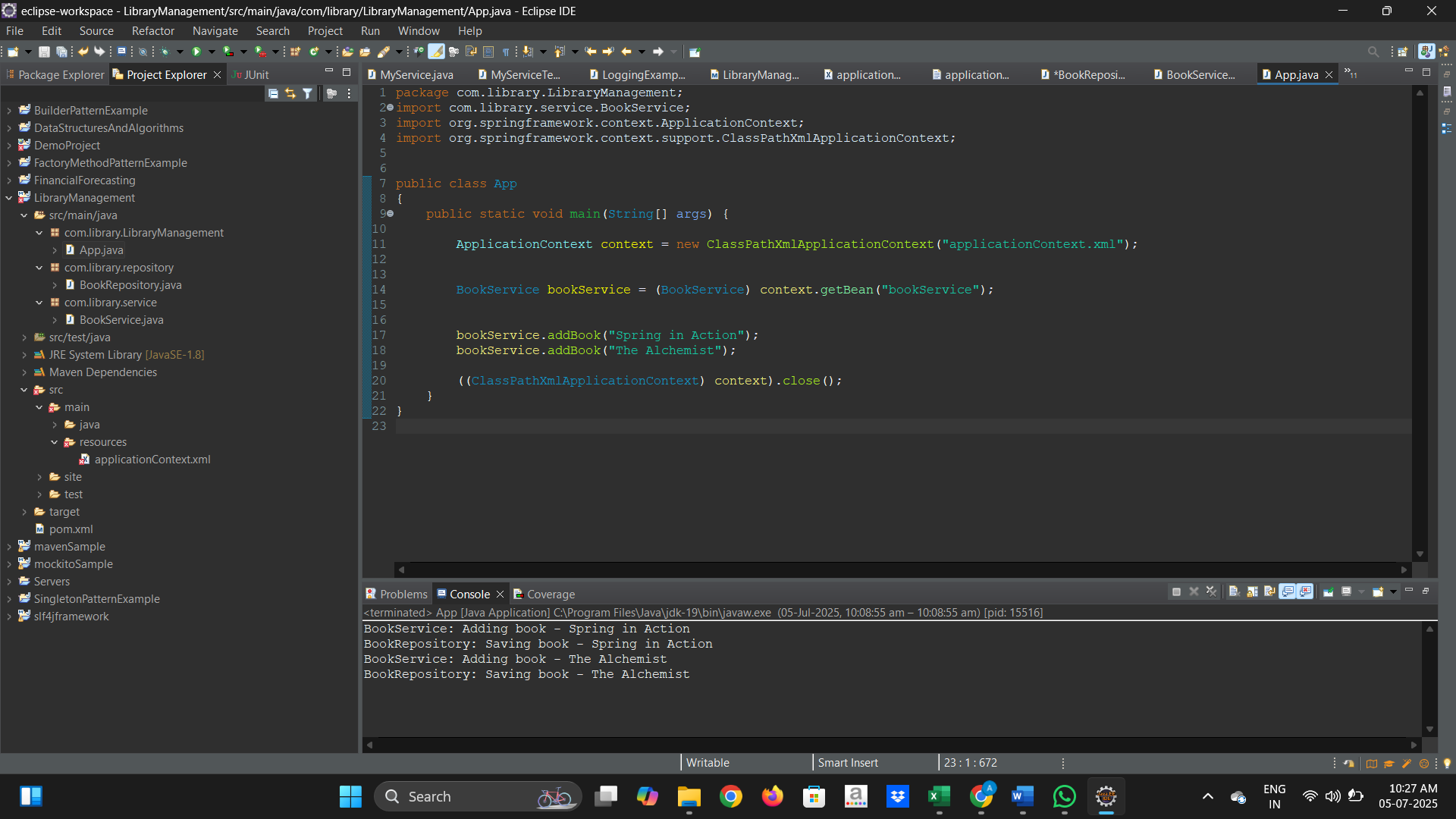
bookService.addBook("The Alchemist");

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT:**



**Exercise 2: Implementing Dependency Injection**

**Scenario:**

In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

**IMPLEMENTATION:**

**Pom.xml File:**

1.Adding dependency

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

**ApplicationContext.xml**

Location: src/main/resources

<?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"/>

<! - - Define Book Service bean with Dependency Injection - - >

<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) {

// **TODO** Auto-generated method stub

System.***out***.println("BookRepository: Saving book - " + bookName);

}

}

**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("BookService: Adding book - " + bookName);

bookRepository.saveBook(bookName);

}

}

**App.java**

package com.library.LibraryManagement;

import com.library.service.BookService;

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");

BookService bookService = (BookService) context.getBean("bookService"); bookService.addBook("Spring in Action");

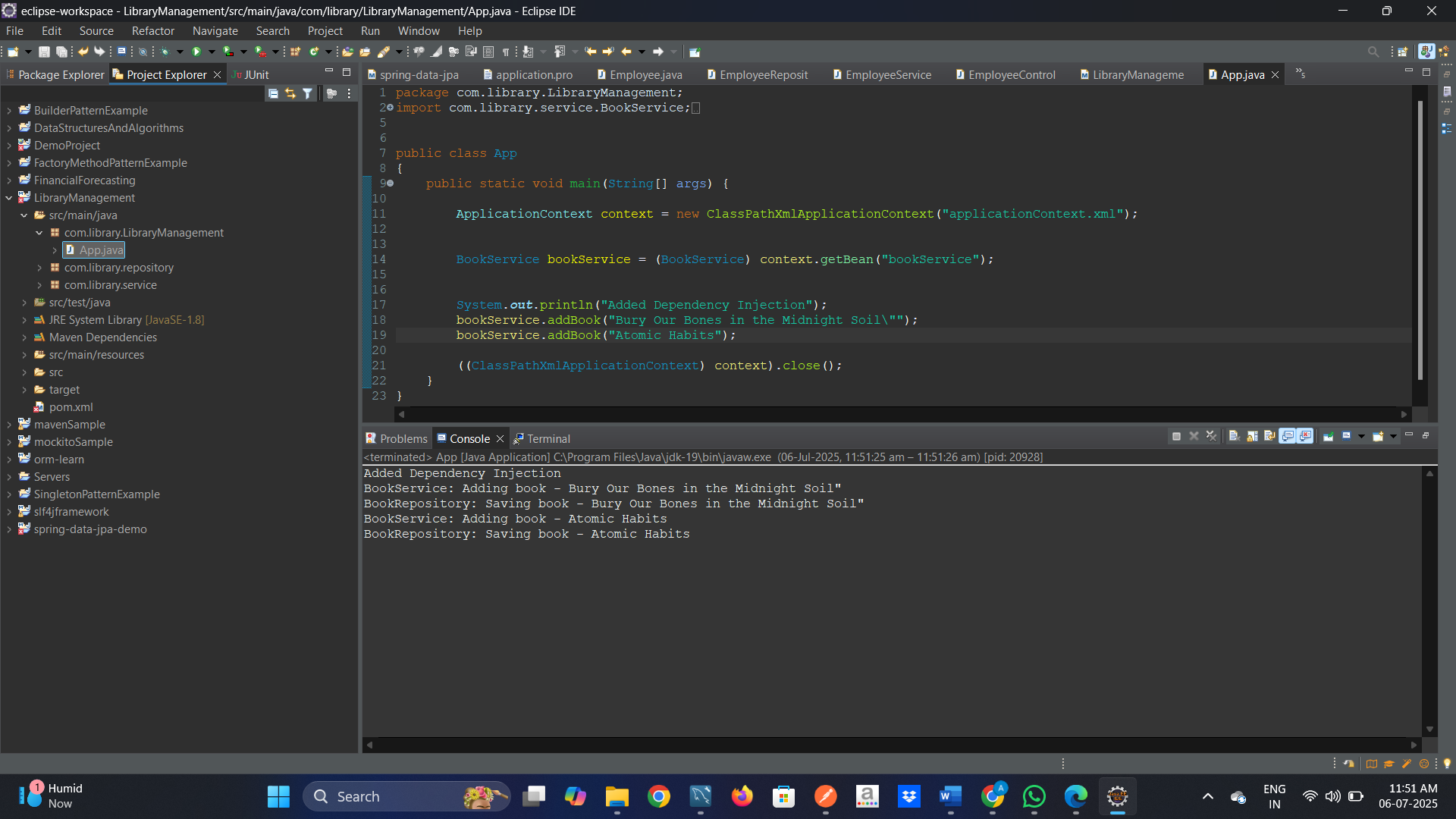
bookService.addBook("The Alchemist");

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT:**



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

**Scenario:**

**You need to set up a new Maven project for the library management application and add Spring dependencies.**

**IMPLEMENTATION:**

**1.Adding dependencies in pom.xml file**

* **Spring AOP**
* **Spring Webmvc**
* **Spring context**

**Pom.xml File**

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

**<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>0.0.1-SNAPSHOT</version>**

**<name>LibraryManagement</name>**

**<description>A simple LibraryManagement.</description>**

**<!-- FIXME change it to the project's website -->**

**<url>http://www.example.com</url>**

**<properties>**

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

**<maven.compiler.source>8</maven.compiler.source>**

**<maven.compiler.target>8</maven.compiler.target>**

**</properties>**

**<dependencies>**

**<dependency>**

**<groupId>junit</groupId>**

**<artifactId>junit</artifactId>**

**<version>3.8.1</version>**

**</dependency>**

**<dependency>**

**<groupId>org.springframework</groupId>**

**<artifactId>spring-context</artifactId>**

**<version>5.3.32</version>**

**</dependency>**

**<dependency>**

**<groupId>org.springframework</groupId>**

**<artifactId>spring-aop</artifactId>**

**<version>5.3.34</version>**

**</dependency>**

**<dependency>**

**<groupId>org.springframework</groupId>**

**<artifactId>spring-webmvc</artifactId>**

**<version>5.3.34</version>**

**</dependency>**

**</dependencies>**

**<build>**

**<pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->**

**<plugins>**

**<plugin>**

**<artifactId>maven-clean-plugin</artifactId>**

**<version>3.4.0</version>**

**</plugin>**

**<plugin>**

**<artifactId>maven-site-plugin</artifactId>**

**<version>3.12.1</version>**

**</plugin>**

**<plugin>**

**<artifactId>maven-project-info-reports-plugin</artifactId>**

**<version>3.6.1</version>**

**</plugin>**

**<!-- see http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_jar\_packaging -->**

**<plugin>**

**<artifactId>maven-resources-plugin</artifactId>**

**<version>3.3.1</version>**

**</plugin>**

**<plugin>**

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

**<version>3.8.1</version>**

**<configuration>**

**<source>1.8</source>**

**<target>1.8</target>**

**</configuration>**

**</plugin>**

**<plugin>**

**<artifactId>maven-surefire-plugin</artifactId>**

**<version>3.3.0</version>**

**</plugin>**

**<plugin>**

**<artifactId>maven-jar-plugin</artifactId>**

**<version>3.4.2</version>**

**</plugin>**

**<plugin>**

**<artifactId>maven-install-plugin</artifactId>**

**<version>3.1.2</version>**

**</plugin>**

**<plugin>**

**<artifactId>maven-deploy-plugin</artifactId>**

**<version>3.1.2</version>**

**</plugin>**

**</plugins>**

**</pluginManagement>**

**</build>**

**<reporting>**

**<plugins>**

**<plugin>**

**<artifactId>maven-project-info-reports-plugin</artifactId>**

**</plugin>**

**</plugins>**

**</reporting>**

**</project>**

**OUTPUT**

