**Spring Core and Maven**

**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.

**SOLUTION:**

**CODE:**

**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

https://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>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.38</version>

</dependency>

</dependencies>

</project>

**applicationContext.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>

**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 displayBook() {

String bookDetails = bookRepository.getBookDetails();

System.***out***.println("Book Details: " + bookDetails);

}

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

public String getBookDetails() {

return "Book: The Count of Monte Cristo, Author: Alexandre Dumas";

}

}

**LibraryApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryApp {

public static void main(String[] args) {

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

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

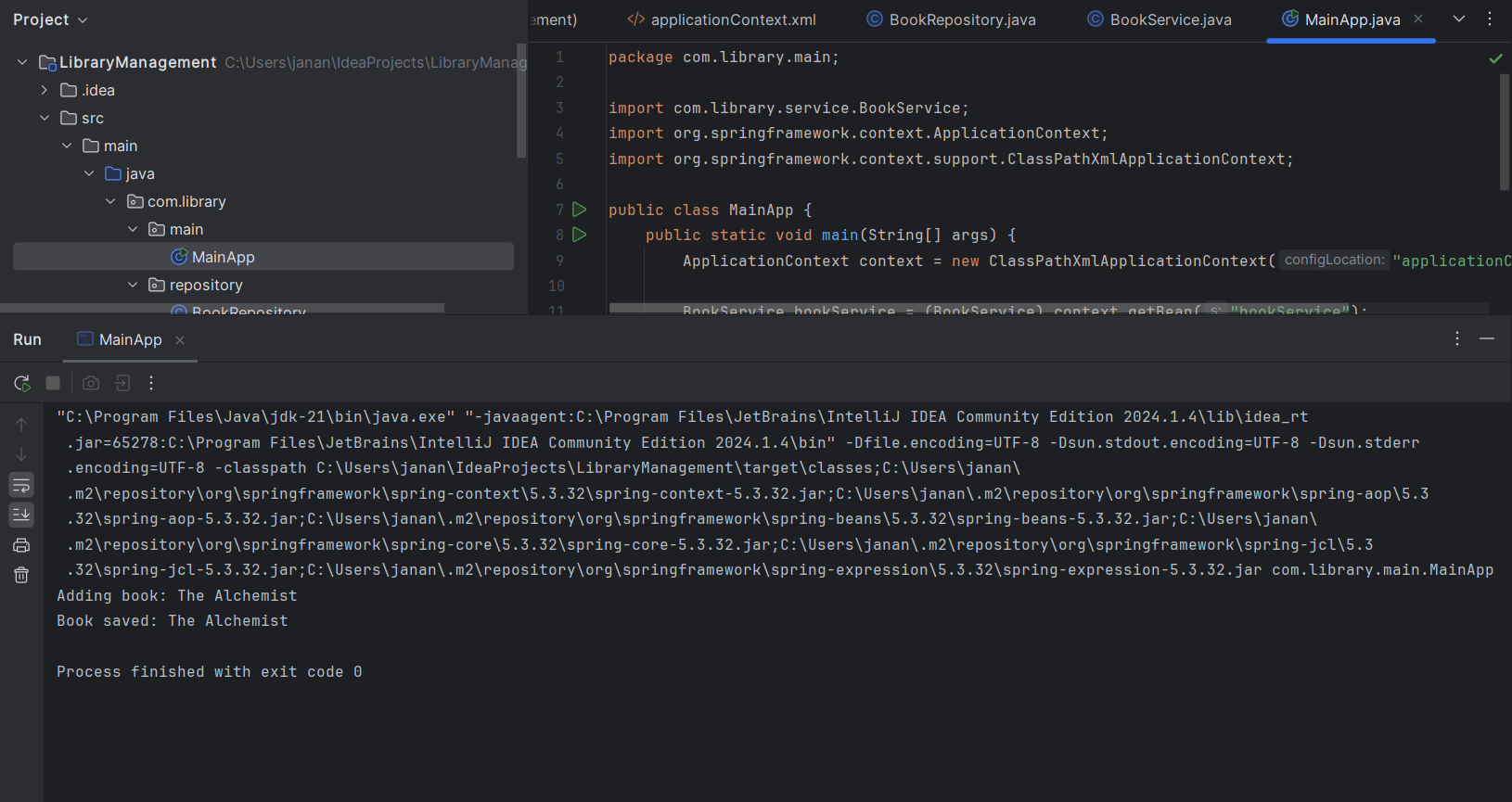
bookService.displayBook();

((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.

**SOLUTION:**

**CODE:**

**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

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

<modelVersion>4.0.0</modelVersion>

<groupId>com.example.library</groupId>

<artifactId>LibraryManagement2</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

</dependencies>

</project>

**applicationContext.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.example.library.BookRepository" />

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

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

</bean>

</beans>

**BookRepository.java**

package com.example.library;

public class BookRepository {

public void displayBooks() {

System.***out***.println("Displaying all books from the repository.");

}

}

**BookService.java**

package com.example.library;

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void listBooks() {

bookRepository.displayBooks();

}

}

**LibraryApplicationManagement.java**

package com.example.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class LibraryManagementApplication {

public static void main(String[] args) {

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

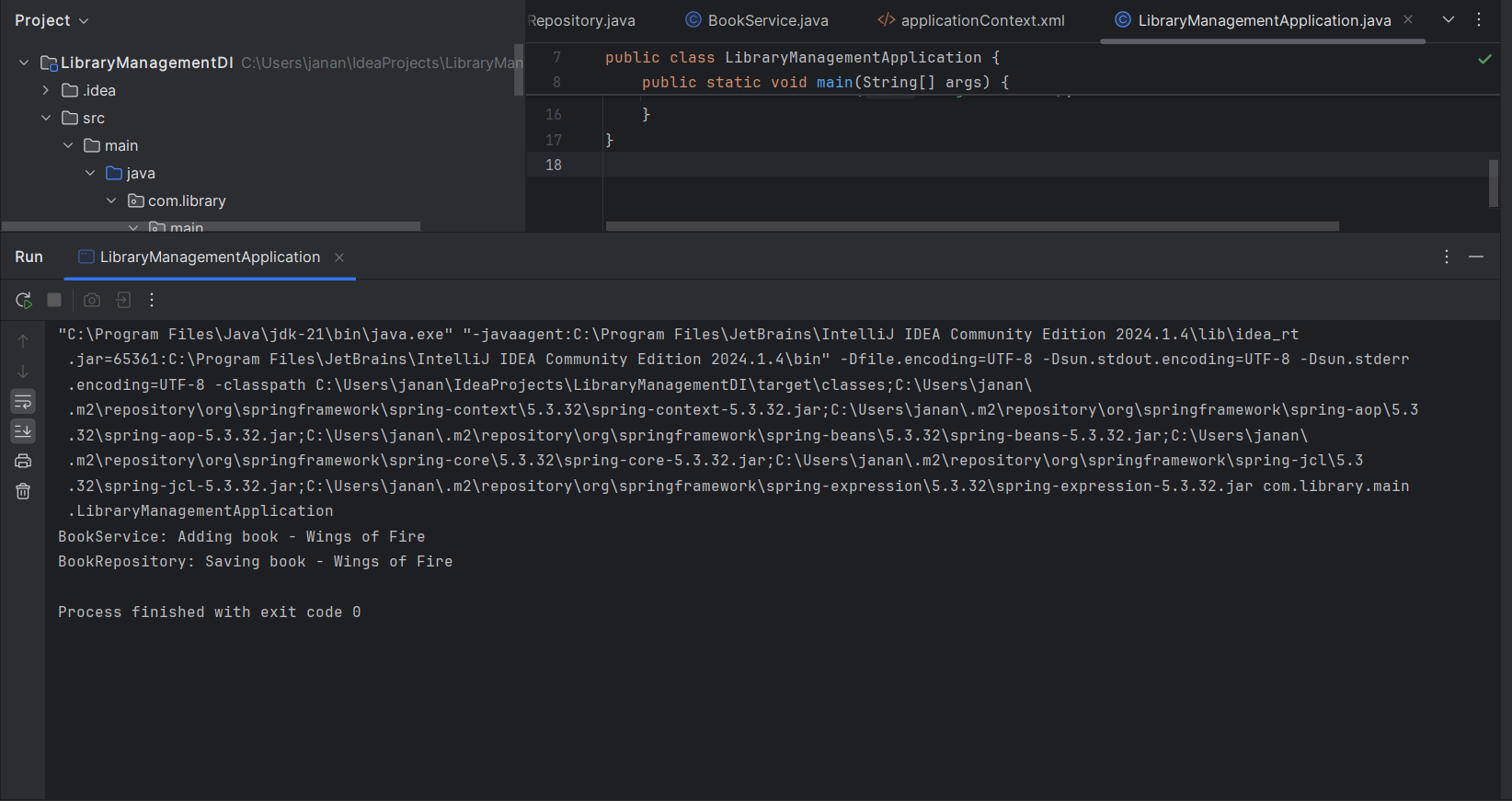
BookService bookService = context.getBean("bookService", BookService.class);

bookService.listBooks();

}

}

**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.

**SOLUTION:**

**CODE:**

**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

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

<modelVersion>4.0.0</modelVersion>

<groupId>com.example.library</groupId>

<artifactId>LibraryManagement3</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>

<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>

**Book class**

package org.example;  
  
public class Book {  
 private String title;  
 private String author;  
 private String genre;  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
  
 public String getAuthor() {  
 return author;  
 }  
  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
  
 public String getGenre() {  
 return genre;  
 }  
  
 public void setGenre(String genre) {  
 this.genre = genre;  
 }  
  
 public Book() {  
 }  
  
 public Book(String author, String genre, String title) {  
 this.author = author;  
 this.genre = genre;  
 this.title = title;  
 }  
  
 @Override  
 public String toString() {  
 return "Book{" +  
 "title='" + title + '\'' +  
 ", author='" + author + '\'' +  
 ", genre='" + genre + '\'' +  
 '}';  
 }  
}

**LibraryService**

package org.example;  
  
import java.util.\*;  
  
public class LibraryService {  
 private List<Book> book;  
   
 public List<Book> getBook() {  
 return book;  
 }  
  
 public void setBook(List<Book> book) {  
 this.book = book;  
 }  
  
 @Override  
 public String toString() {  
 return "LibraryService{" +  
 "book=" + book +  
 '}';  
 }  
}

**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 name="book1" class="org.example.Book">  
 <property name="title" value="Harry Potter"/>  
 <property name="author" value="J.K Rowling"/>  
 <property name="genre" value="Fiction"/>  
 </bean>  
 <bean name="book2" class="org.example.Book">  
 <property name="title" value="The Silent Patient"/>  
 <property name="author" value="Alex Michaelides"/>  
 <property name="genre" value="Psychological Thriller"/>  
 </bean>  
 <bean name="library" class="org.example.LibraryService">  
 <property name="book">  
 <list>  
 <ref bean="book1"/>  
 <ref bean="book2"/>  
 </list>  
 </property>  
 </bean>  
  
  
</beans>

**MAIN class**

package org.example;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.springframework.core.annotation.Order;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");  
 LibraryService libraryService = (LibraryService) context.getBean("library");  
 System.*out*.println(libraryService.getBook());  
  
  
 }  
}

**OUTPUT:**

