# **Exercise 2: Implementing Dependency Injection**

## **Scenario**

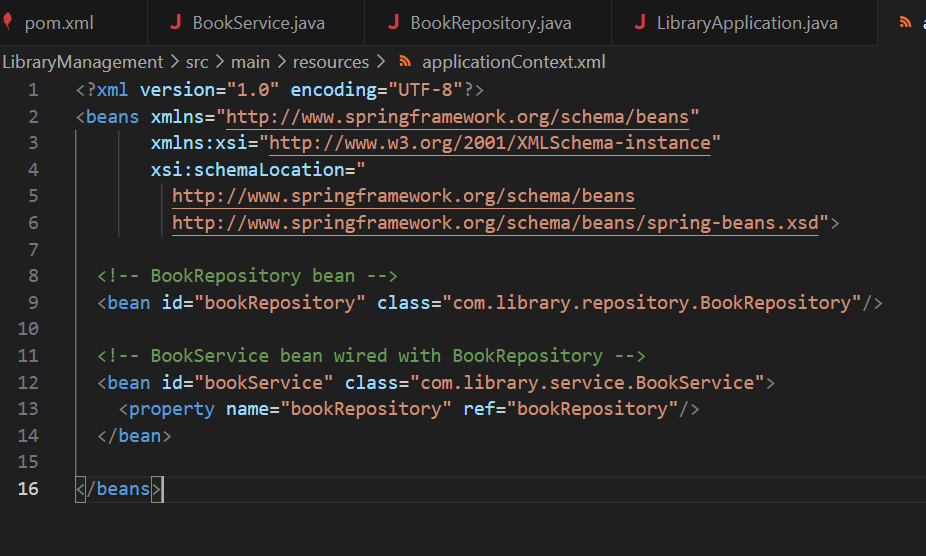
In the Library Management application, you need to manage the dependency between BookService and BookRepository using Spring’s IoC container and Dependency Injection.

## **Step 1: Modify the XML Configuration**

**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">  
  
 <!-- BookRepository bean -->  
 <bean id="bookRepository" class="com.library.repository.BookRepository"/>  
  
 <!-- BookService bean wired with BookRepository -->  
 <bean id="bookService" class="com.library.service.BookService">  
 <property name="bookRepository" ref="bookRepository"/>  
 </bean>  
  
</beans>

This configuration defines both beans and uses setter injection to wire BookRepository into BookService.

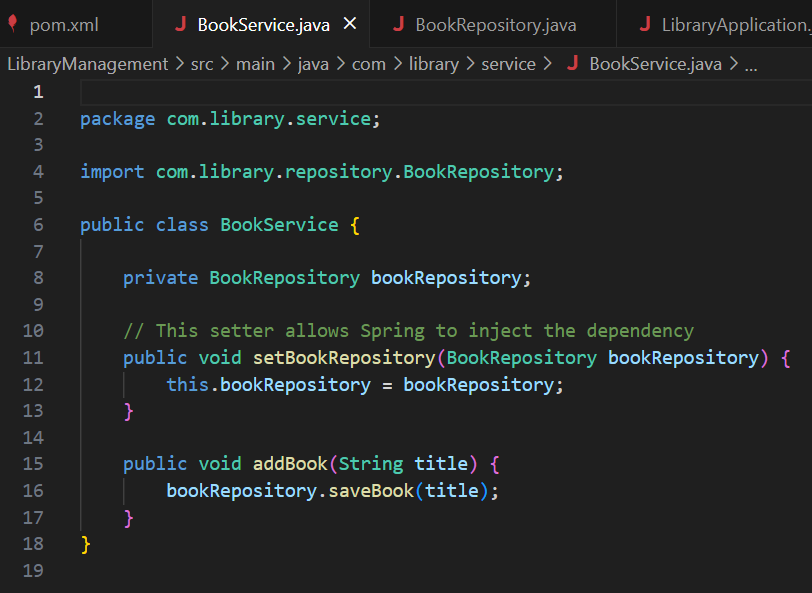


## **Step 2: BookService Class**

**BookService.java:**

package com.library.service;  
  
import com.library.repository.BookRepository;  
  
public class BookService {  
  
 private BookRepository bookRepository;  
  
 // Setter for Dependency Injection  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void addBook(String title) {  
 bookRepository.saveBook(title);  
 }  
}

The setter method enables Spring to inject the dependency.

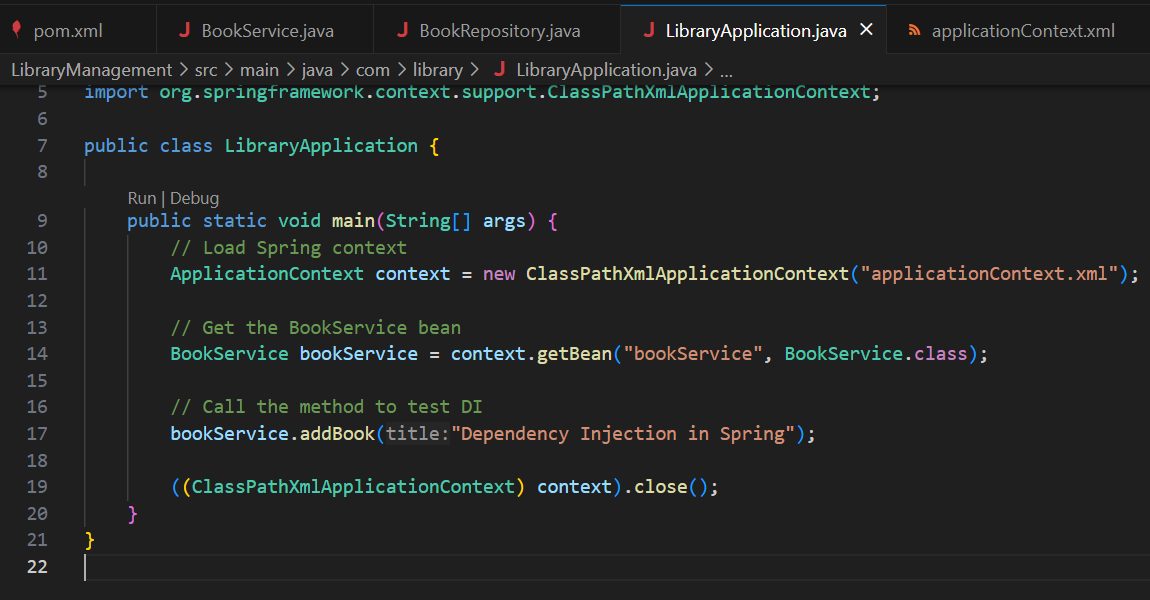


## **Step 3: Test the Configuration**

**LibraryApplication.java:**

package com.library;  
  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class LibraryApplication {  
  
 public static void main(String[] args) {  
  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
   
 BookService bookService = context.getBean("bookService", BookService.class);  
  
   
 bookService.addBook("Dependency Injection in Spring");  
  
 ((ClassPathXmlApplicationContext) context).close();  
 }  
}

This main class loads the Spring configuration, retrieves the bean, and runs the method to check that dependency injection works.



## **Output**

When you run:

mvn compile  
mvn exec:java

You should see:

Book saved: Dependency Injection in Spring

This confirms that Spring IoC and DI are working correctly.

