Exercise 2: Implementing Dependency Injection

# Step 1: Modify the XML Configuration

We update the applicationContext.xml file to define dependency injection. Specifically, we inject BookRepository into BookService using the <property> tag. This tells Spring to automatically call the setter method of BookService and provide it with an instance of BookRepository.

Updated applicationContext.xml:

<bean id="bookRepository" class="com.library.repository.BookRepository"/>  
  
<bean id="bookService" class="com.library.service.BookService">  
 <property name="bookRepository" ref="bookRepository"/>  
</bean>

# Step 2: Update the BookService Class

Here, we add a setter method setBookRepository

Updated BookService.java:

public class BookService {  
 private BookRepository bookRepository;  
  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void performOperation() {  
 System.out.println("BookService: Performing operation...");  
 bookRepository.saveBook();  
 }  
}

# Step 3: Test the Configuration

In the main class LibraryManagementApplication, we load the Spring context from applicationContext.xml and retrieve the BookService bean. We then call the performOperation() method to verify that the dependency injection works correctly.

## Output Screenshot: