**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.

**Steps:**

1. **Modify the XML Configuration:**
   * Update **applicationContext.xml** to wire **BookRepository** into **BookService**.
2. **Update the BookService Class:**
   * Ensure that **BookService** class has a setter method for **BookRepository**.
3. **Test the Configuration:**
   * Run the **LibraryManagementApplication** main class to verify the dependency injection.

**Source Code:**

**Appli.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>

**Bookser.java:**

**package** com.library.service;

**import** com.library.repository.BookRepository;

**public** **class** Bookser {

**private** Bookrep book;

**public** **void** setBookrep(Bookrep book) {

**this**.book=book;

}

**public** **void** display() {

System.***out***.println("Book title: " + book.getTitle());

}

}

**Lib.java:**

**package** com.library;

**import** com.library.service.Bookser;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** Lib {

**public** **static** **void** main(String[] args) {

ApplicationContext context = **new** ClassPathXmlApplicationContext("appli.xml");

Bookser ser = context.getBean("ser", Bookser.**class**);

ser.display();

}

}

**Output:**

Book title: The Kite Runner