**SPRING CORE AND MAVEN**

**MANDATORY HANDS-ON QUESTIONS**

**Question 1**

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

**Scenario:**

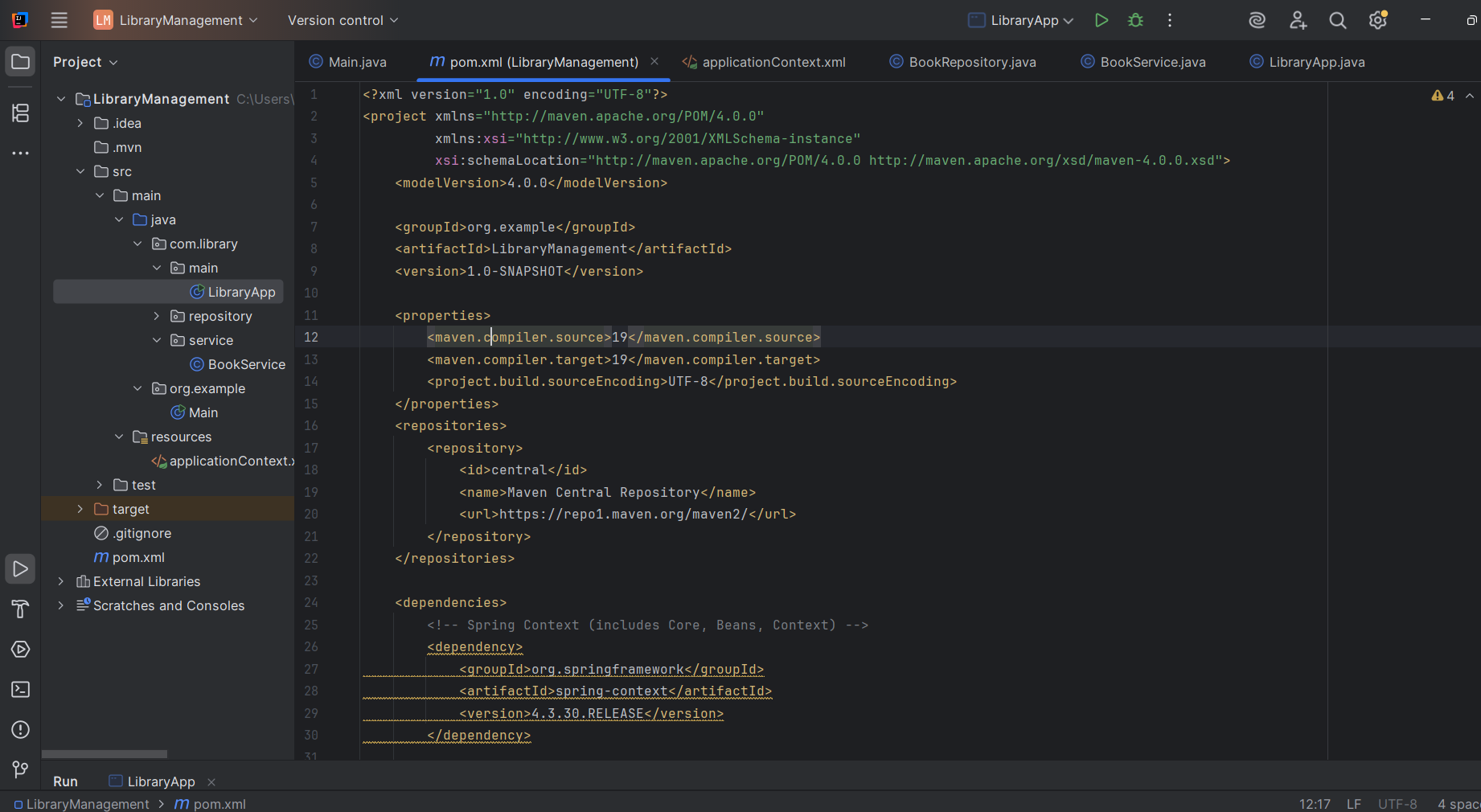
Your company is developing a web application for managing a library. You need to use the Spring Framework to handle the backend operations.

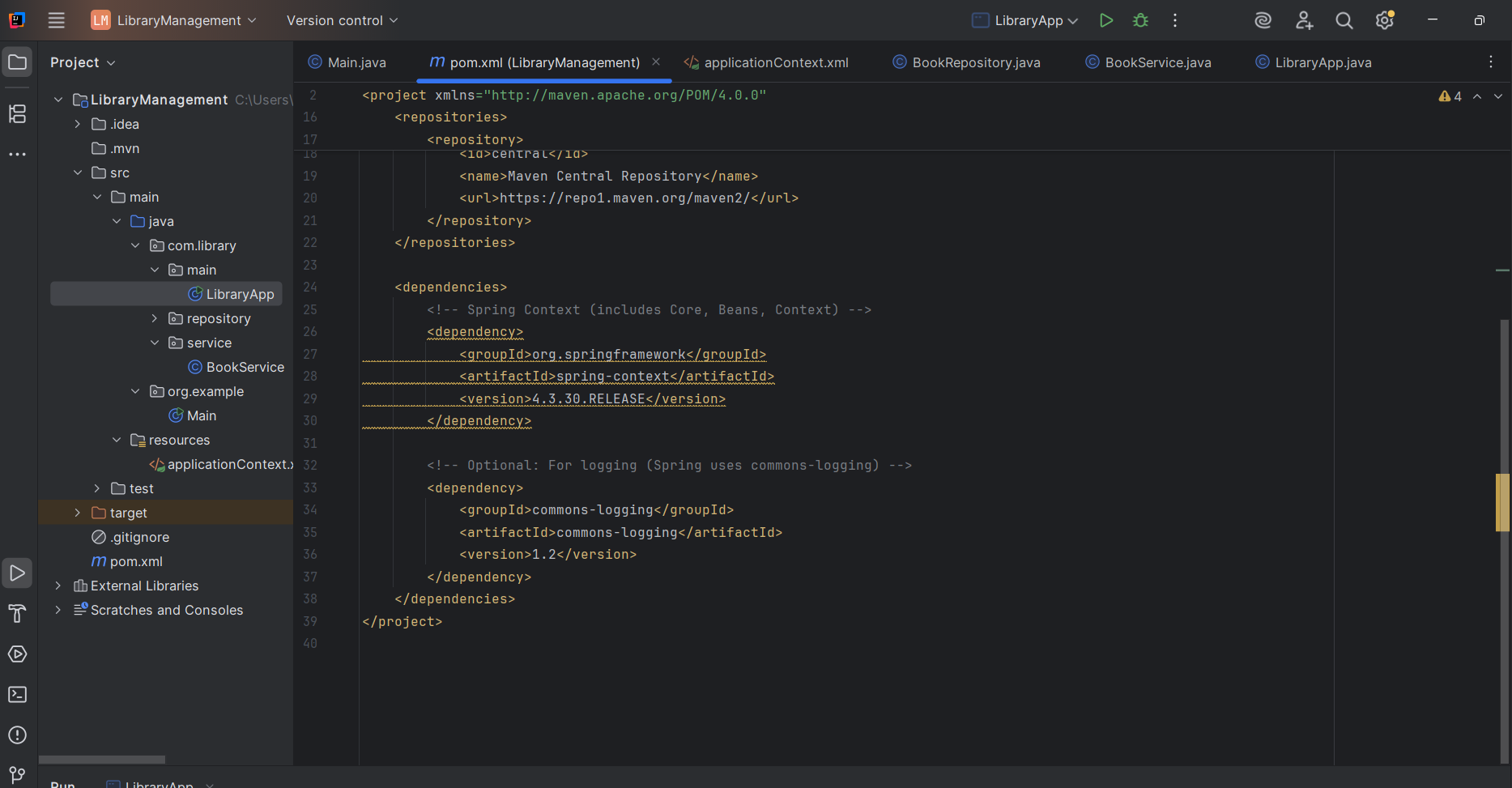
**Steps:**

1. **Set Up a Spring Project:**
   * Create a Maven project named **LibraryManagement**.
   * Add Spring Core dependencies in the **pom.xml** file.
2. **Configure the Application Context:**
   * Create an XML configuration file named **applicationContext.xml** in the **src/main/resources** directory.
   * Define beans for **BookService** and **BookRepository** in the XML file.
3. **Define Service and Repository Classes:**
   * Create a package **com.library.service** and add a class **BookService**.
   * Create a package **com.library.repository** and add a class **BookRepository**.
4. **Run the Application:**
   * Create a main class to load the Spring context and test the configuration.

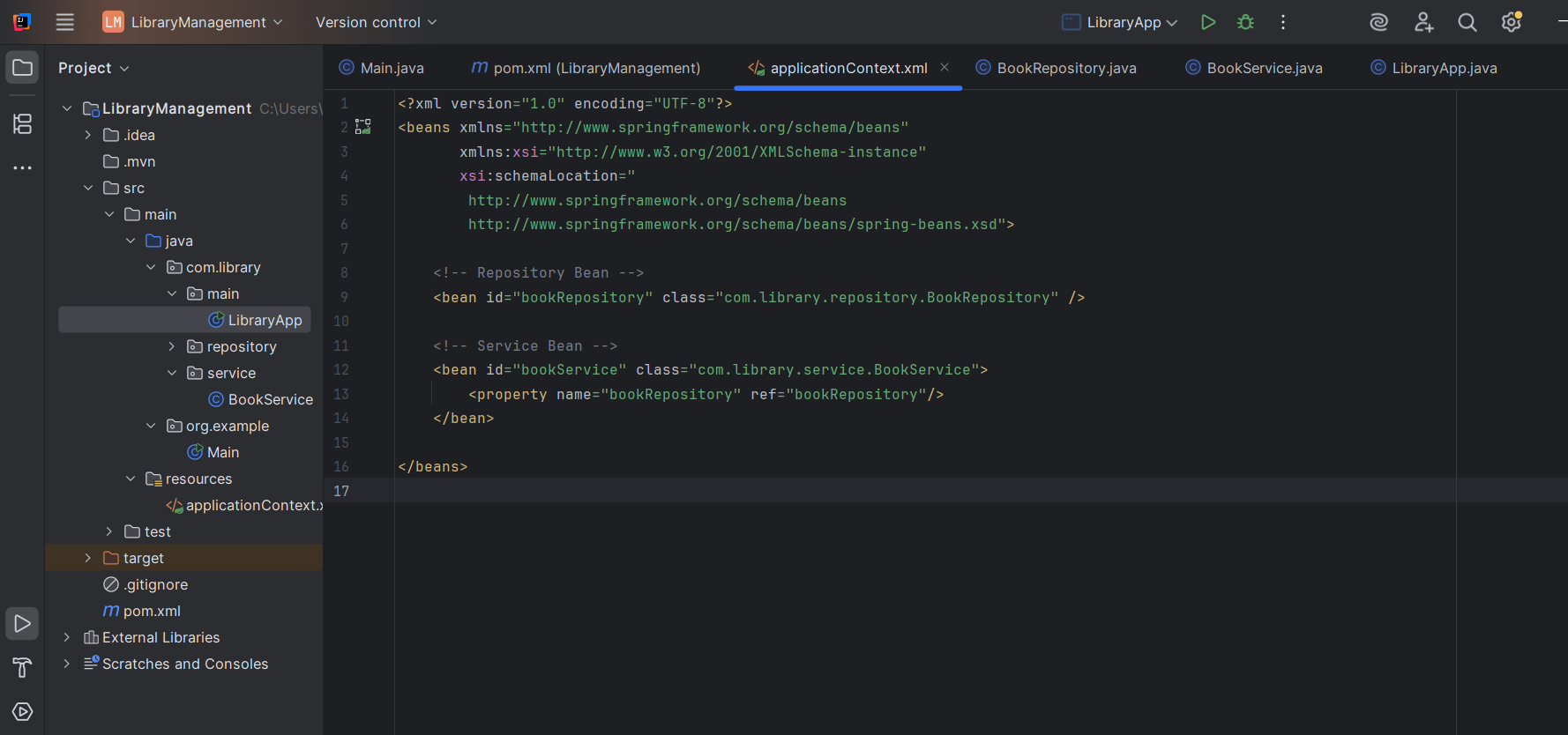
Answer and output:

Step 1: pom.xml file and project

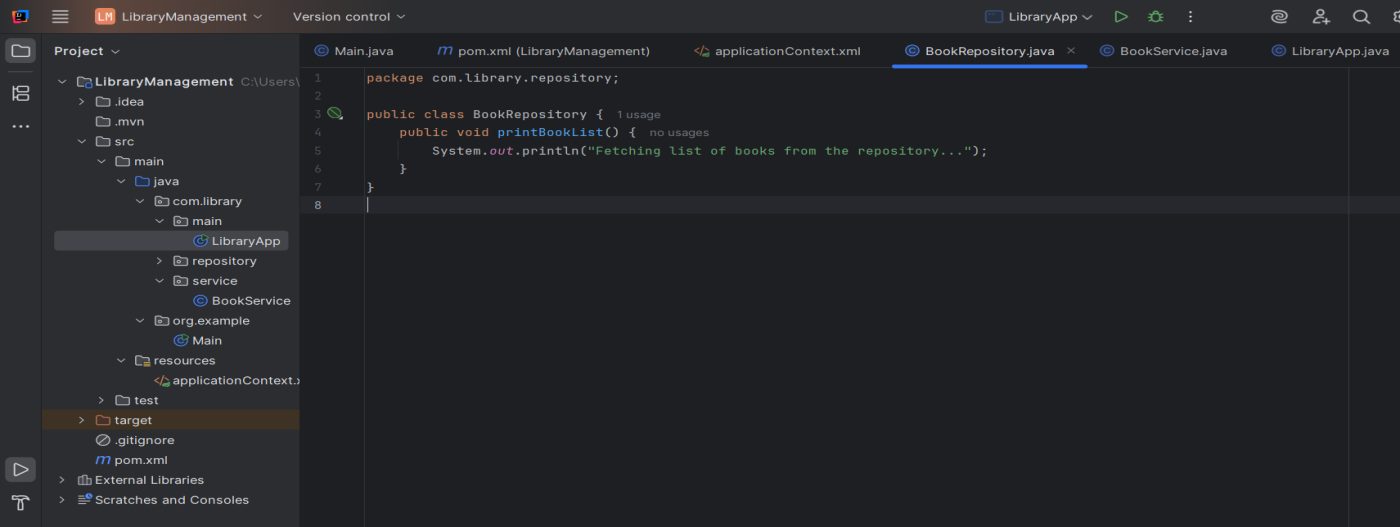




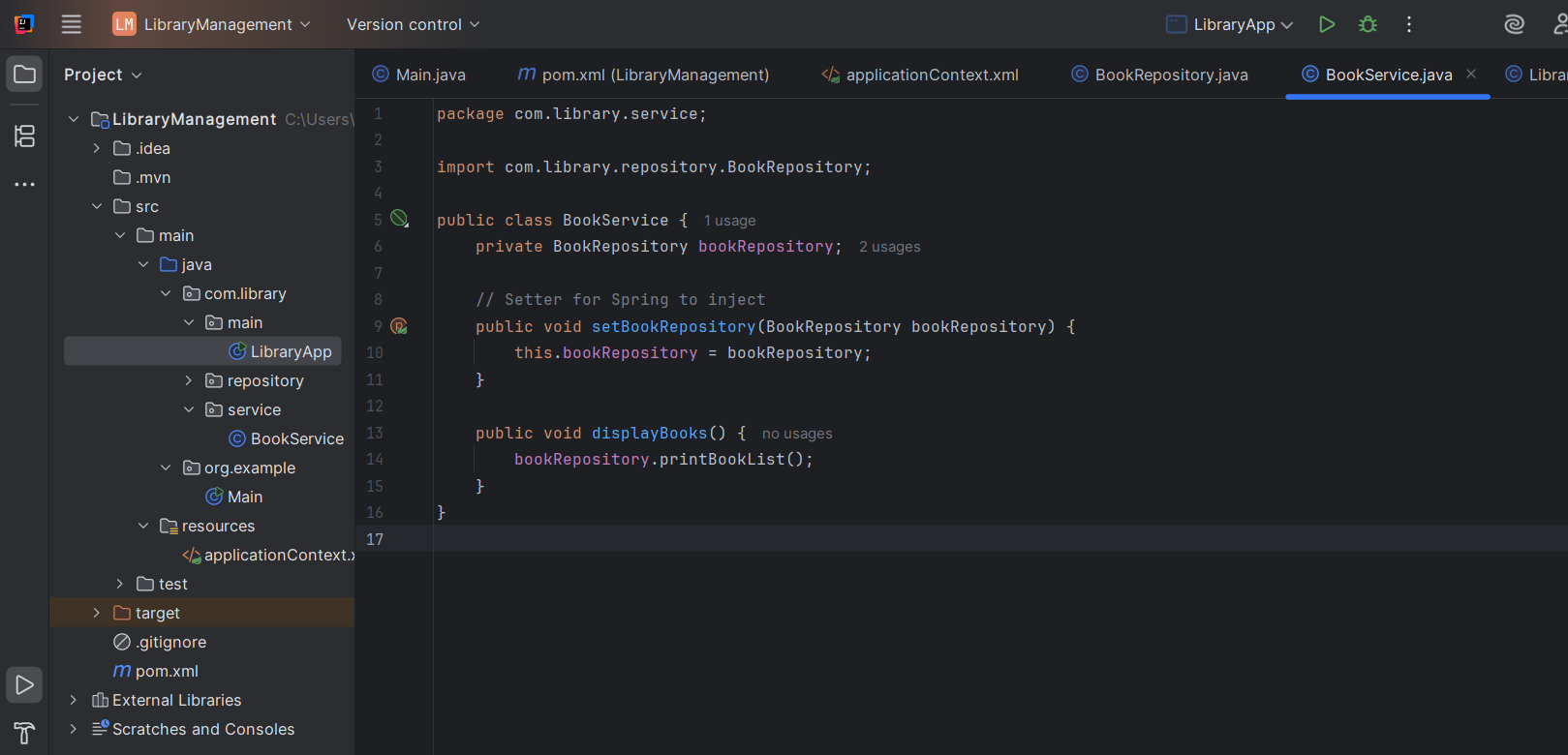
Step 2: appcontext.xml and beans



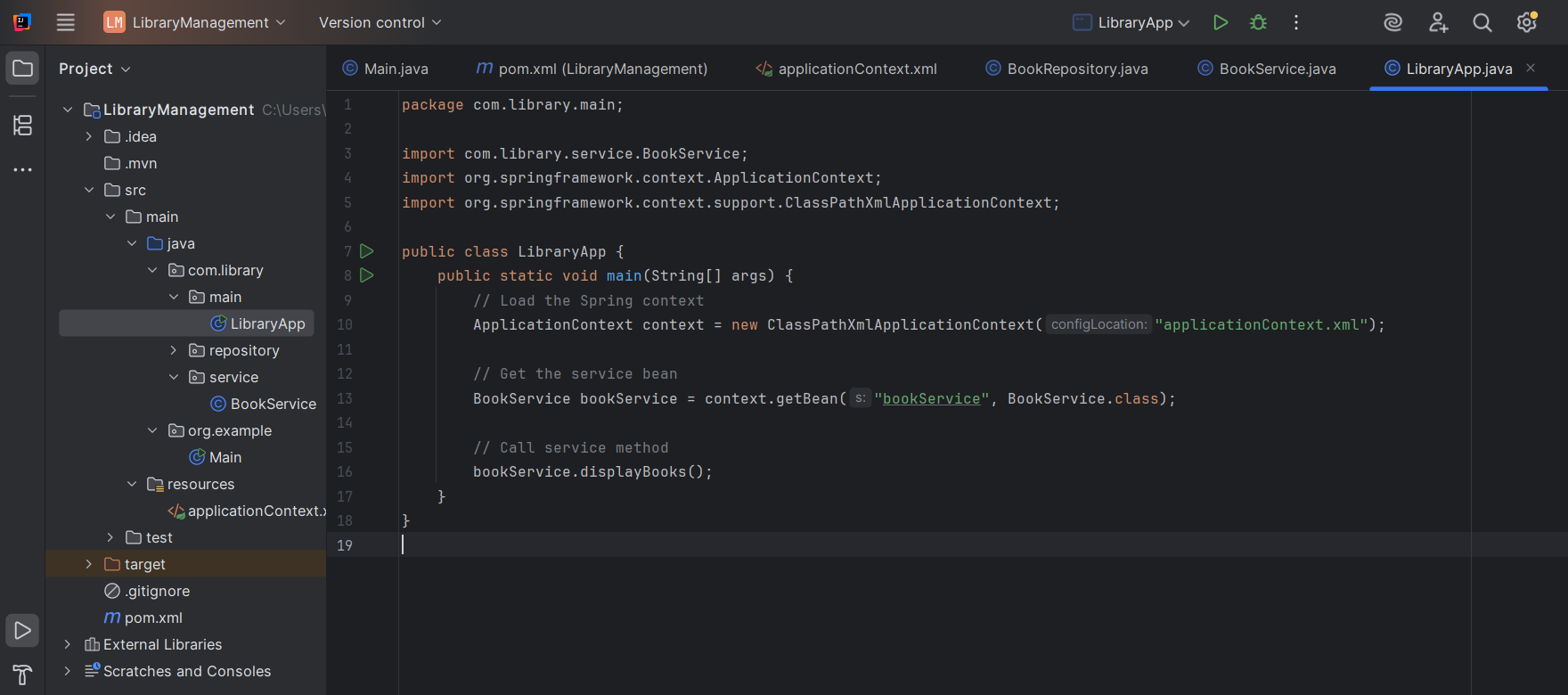
Bookrepository:



Bookservice:

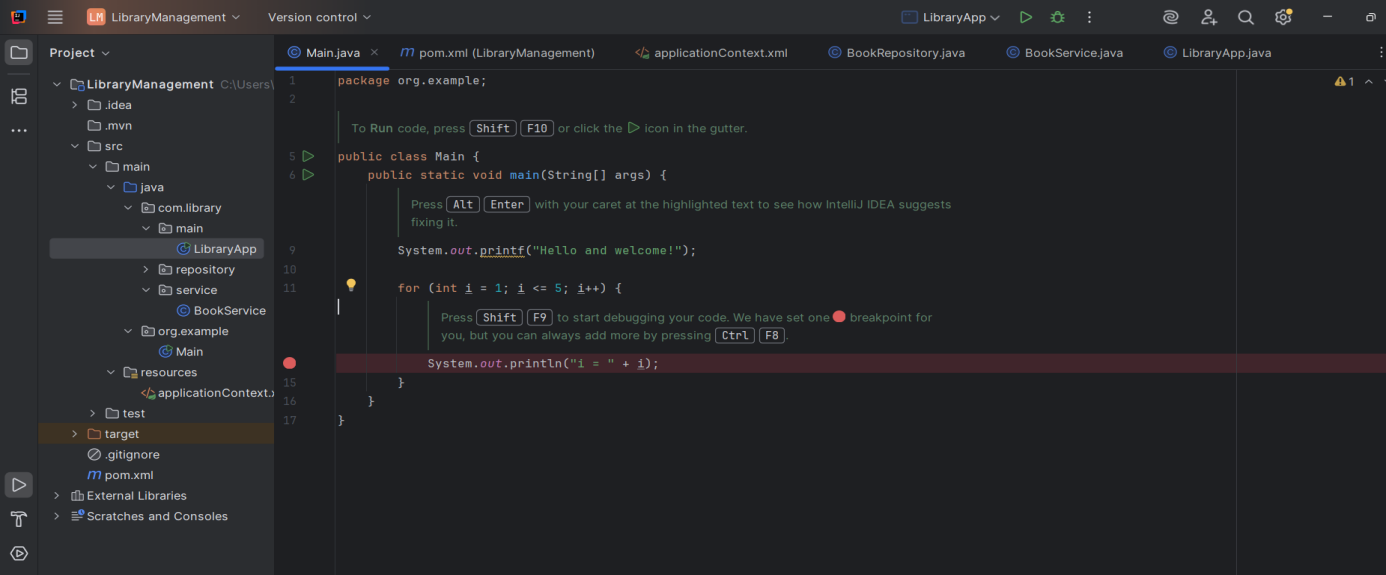


LibraryApp:

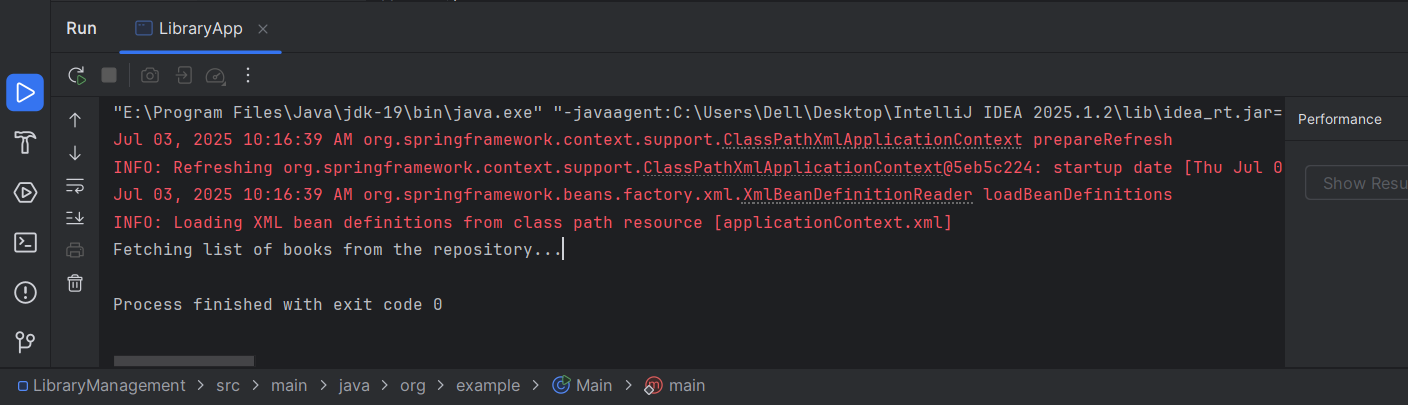


Step 3:running the application

Main:



**OUTPUT:**



**Question 2**

**Exercise 2: Implementing Dependency Injection**

**Scenario:**

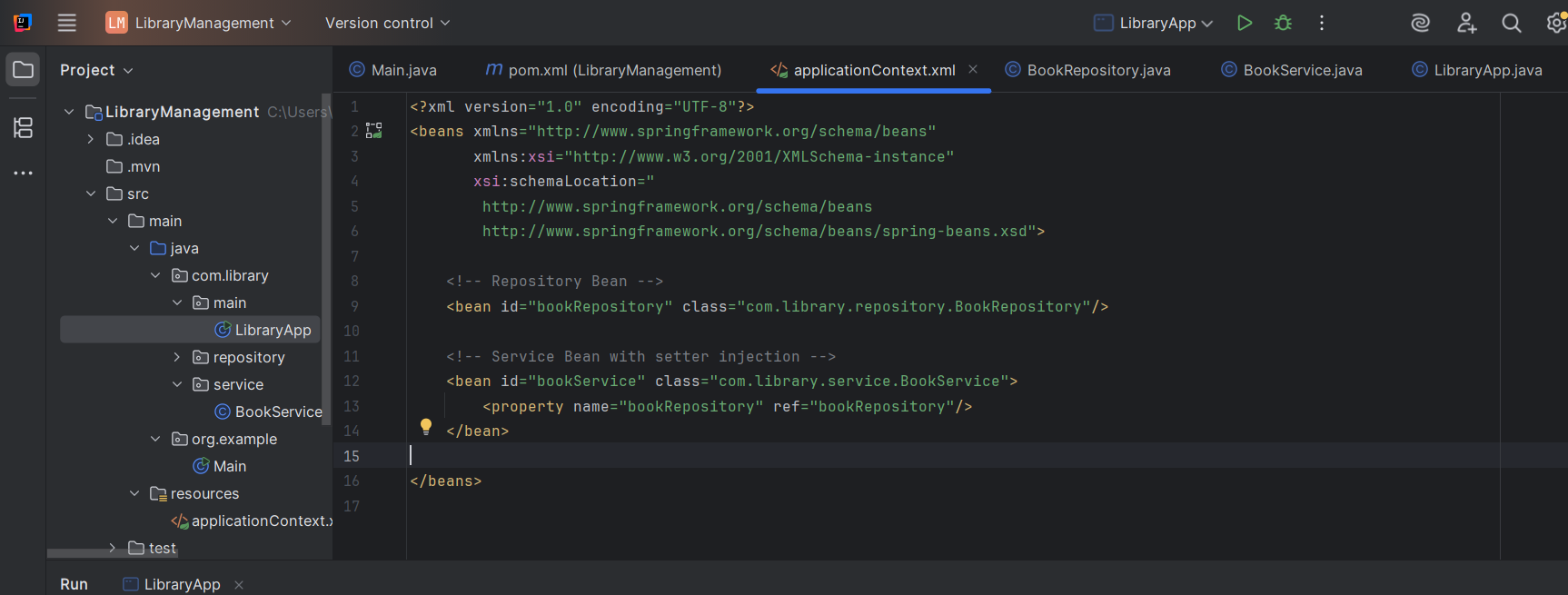
In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

**Steps:**

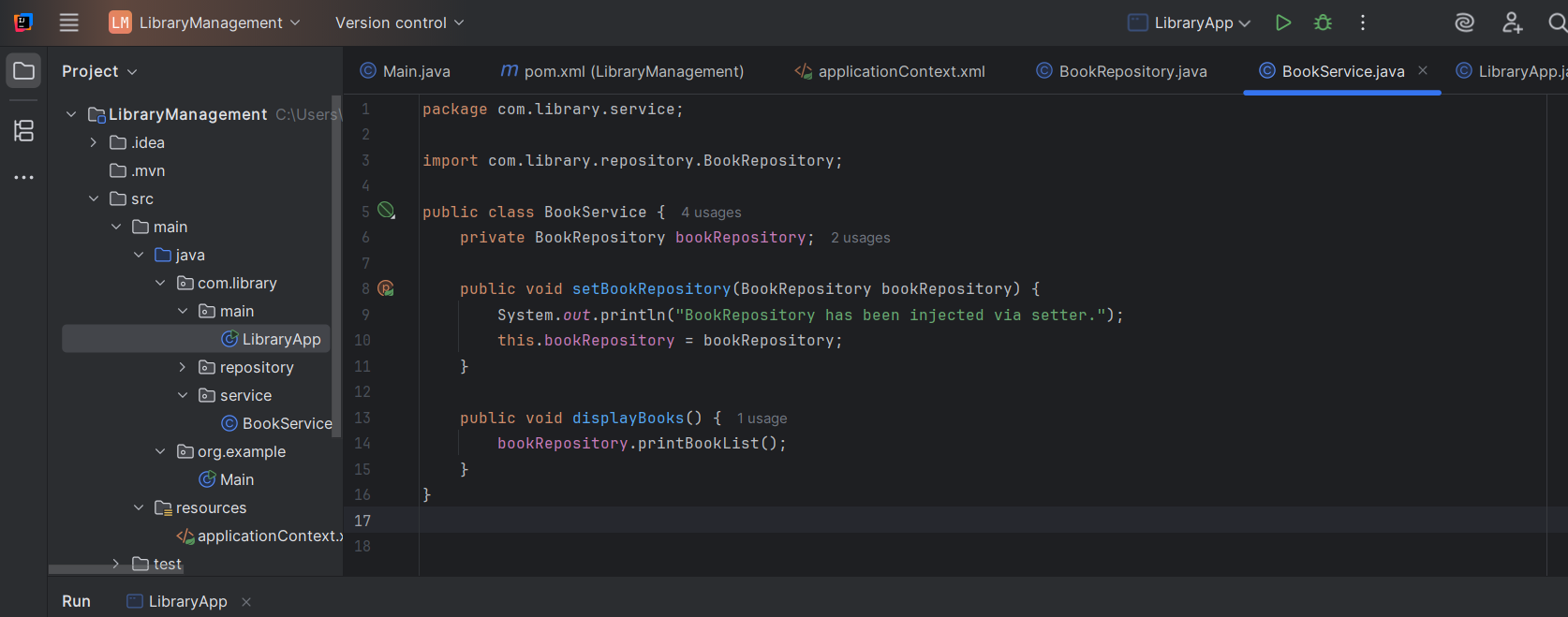
1. **Modify the XML Configuration:**
   * Update **applicationContext.xml** to wire **BookRepository** into **BookService**.
2. **Update the BookService Class:**
   * Ensure that **BookService** class has a setter method for **BookRepository**.
3. **Test the Configuration:**
   * Run the **LibraryManagementApplication**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

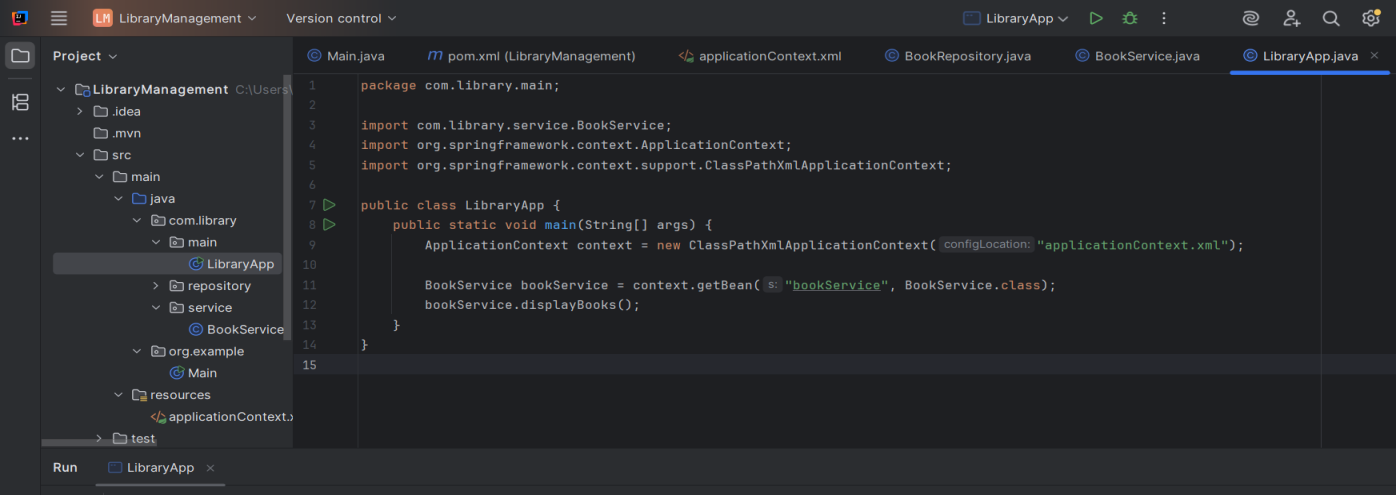
Answer and Output:

Step 1: modifying appcontext.xml

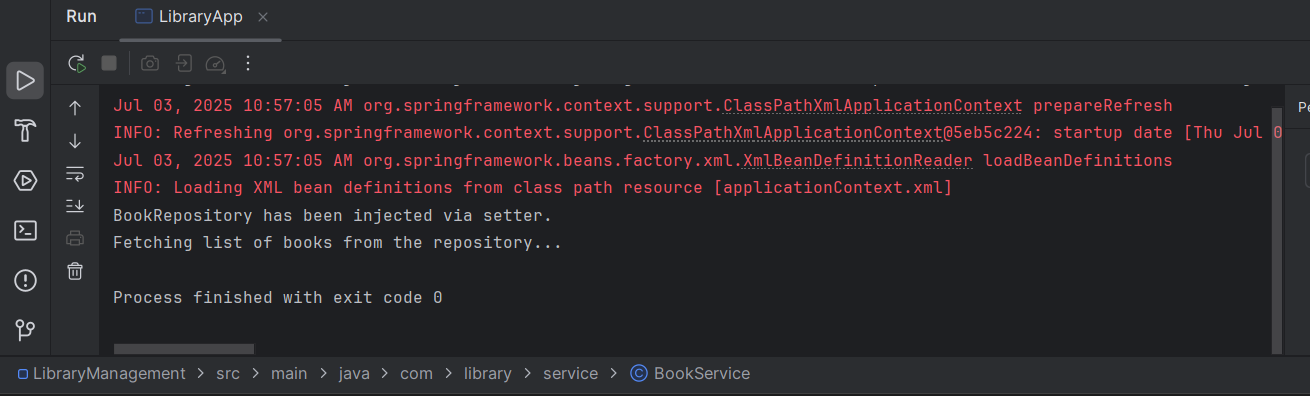


Step 2: ensuring setter method and main class





**OUTPUT:**



**Question 4**

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

**Scenario:**

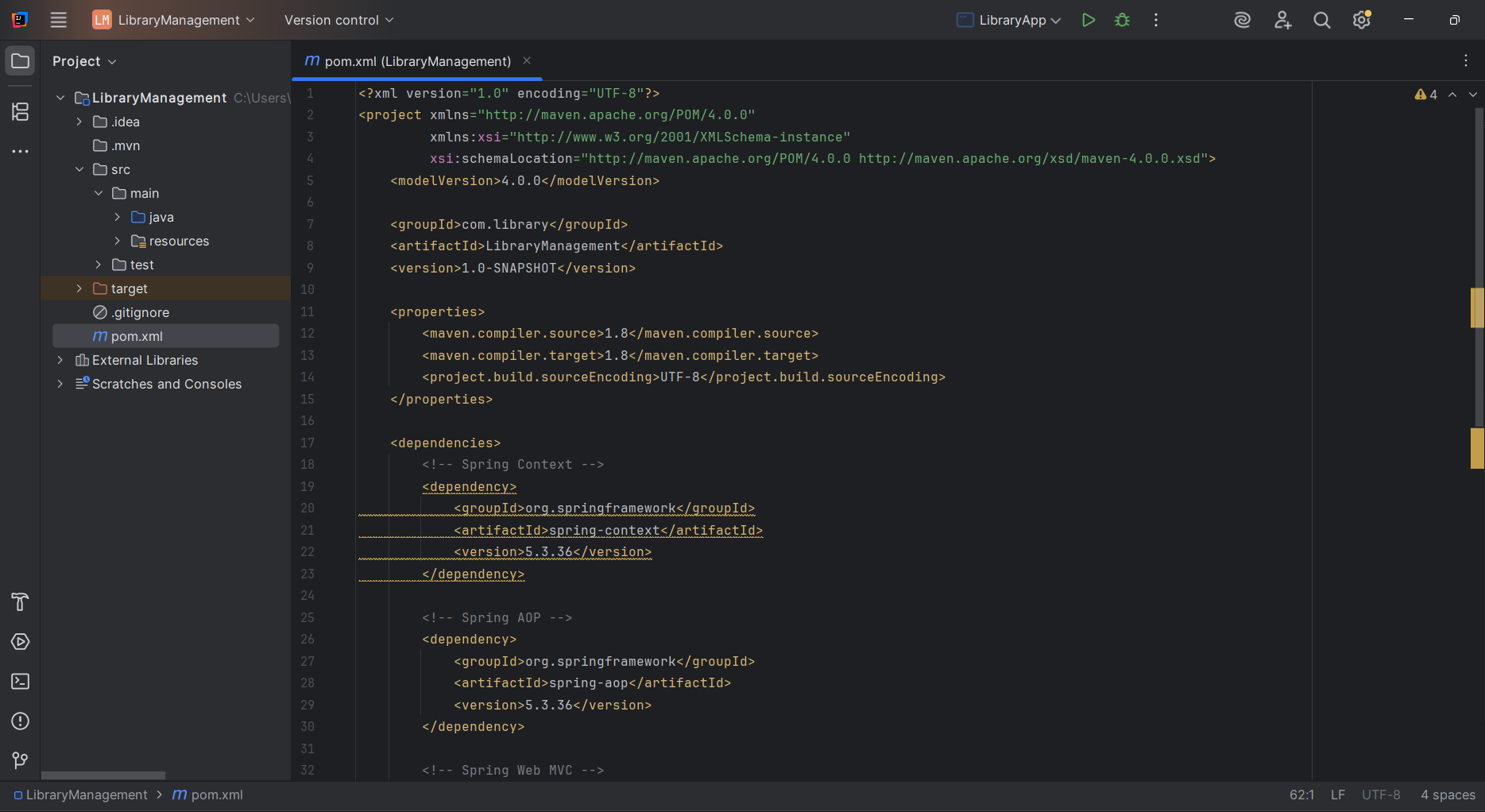
You need to set up a new Maven project for the library management application and add Spring dependencies.

**Steps:**

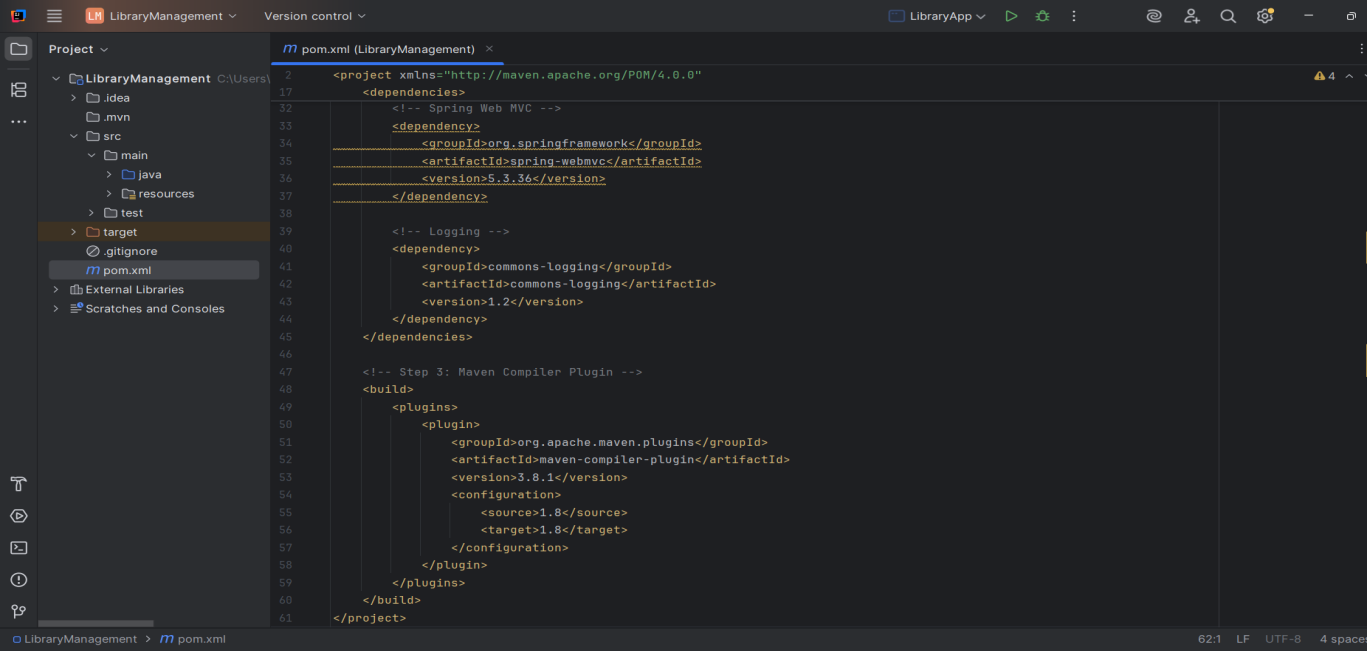
1. **Create a New Maven Project:**
   * Create a new Maven project named **LibraryManagement**.
2. **Add Spring Dependencies in pom.xml:**
   * Include dependencies for Spring Context, Spring AOP, and Spring WebMVC.
3. **Configure Maven Plugins:**
   * Configure the Maven Compiler Plugin for Java version 1.8 in the pom.xml file.

Answer and Output:

Step 1 and Step 2: created Maven project and added dependencies



Step 3: configured java version 1.8



**ADDITIONAL HANDS-ON QUESTIONS**

**QUESTION 1**

**Exercise 5: Configuring the Spring IoC Container**

**Scenario:**

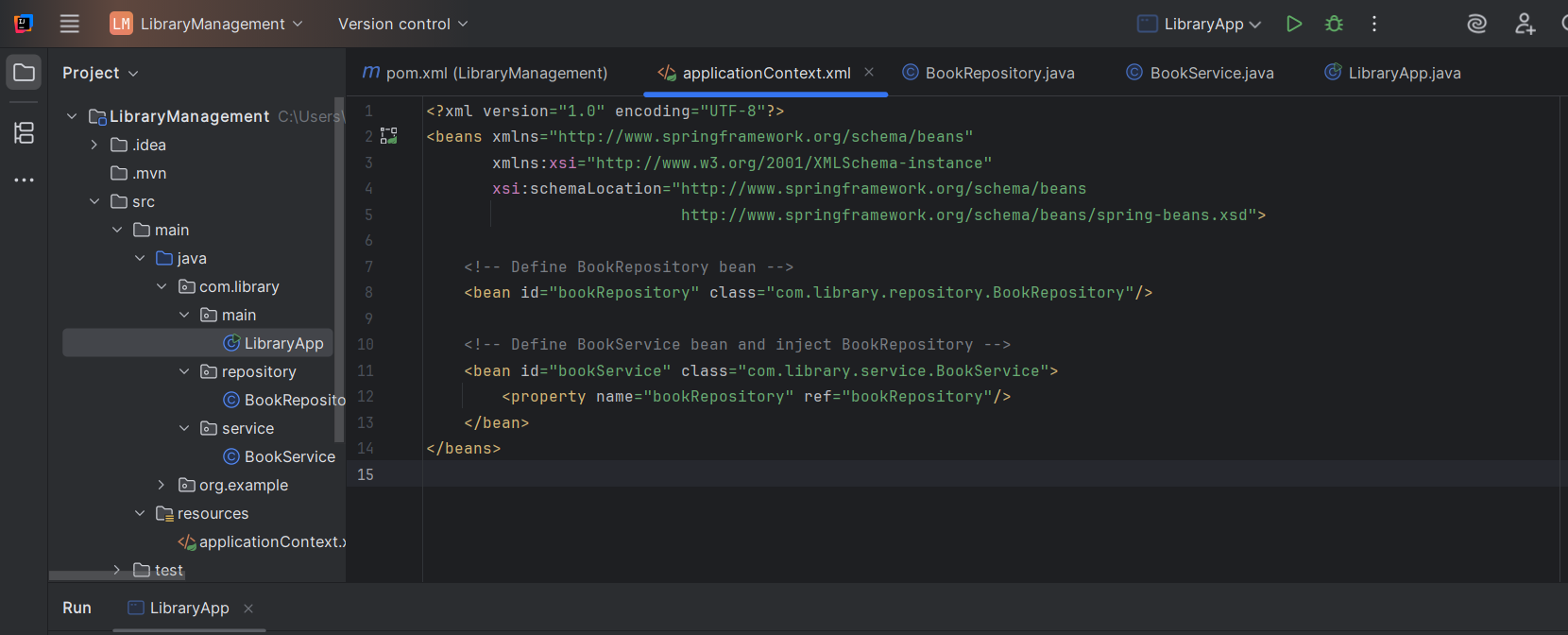
The library management application requires a central configuration for beans and dependencies.

**Steps:**

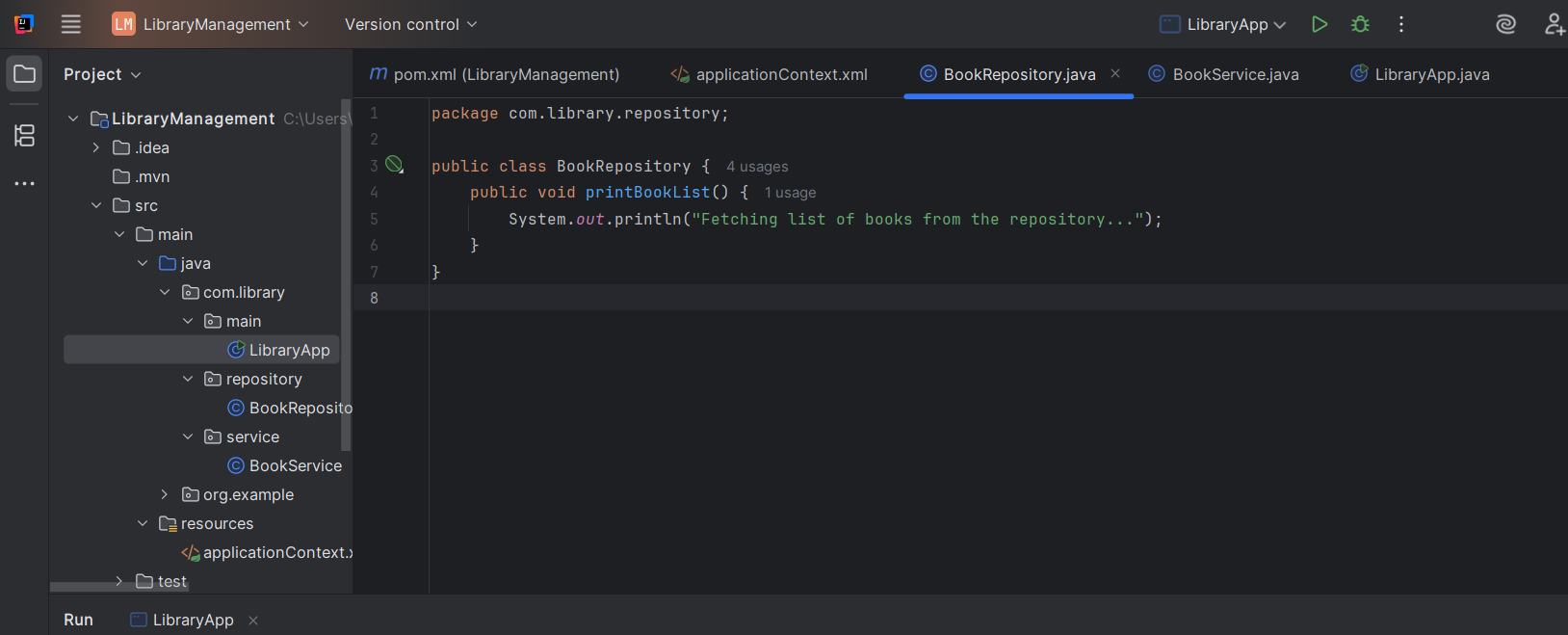
1. **Create Spring Configuration File:**
   * Create an XML configuration file named **applicationContext.xml** in the **src/main/resources** directory.
   * Define beans for **BookService** and **BookRepository** in the XML file.
2. **Update the BookService Class:**
   * Ensure that the **BookService** class has a setter method for **BookRepository**.
3. **Run the Application:**
   * Create a main class to load the Spring context and test the configuration.

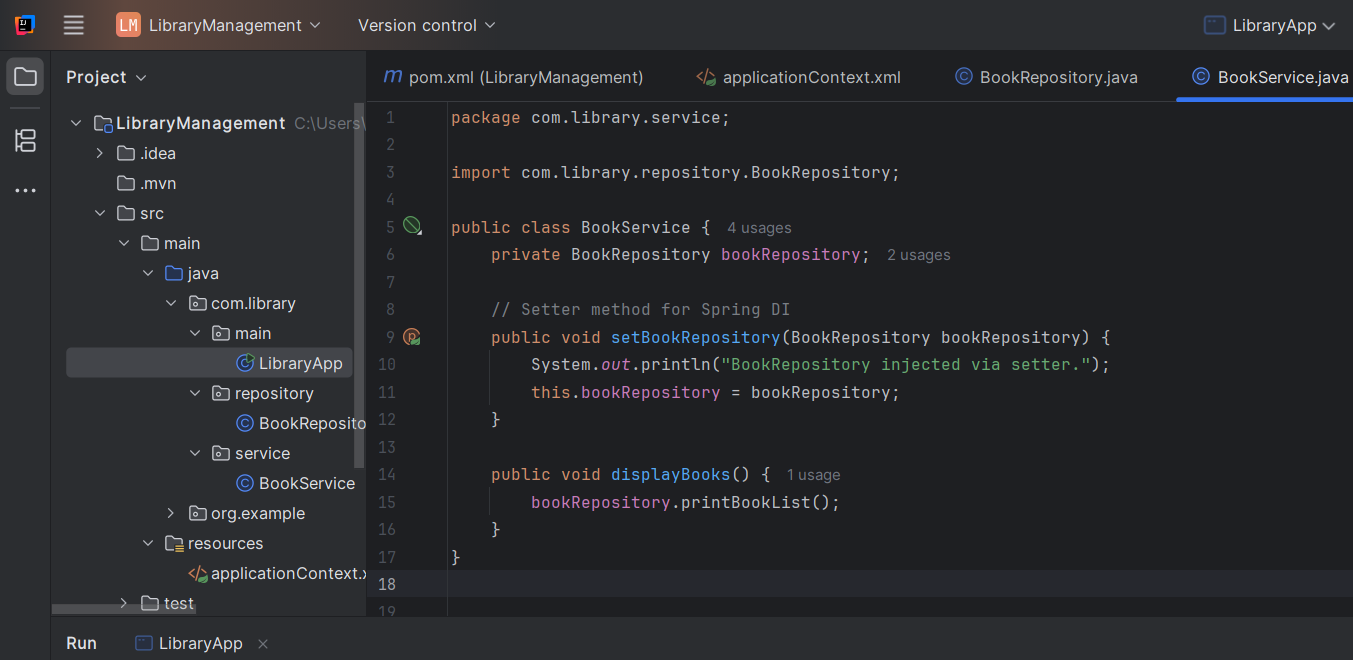
Answer and Output:

Step 1: configured appContext.xml and defined beans

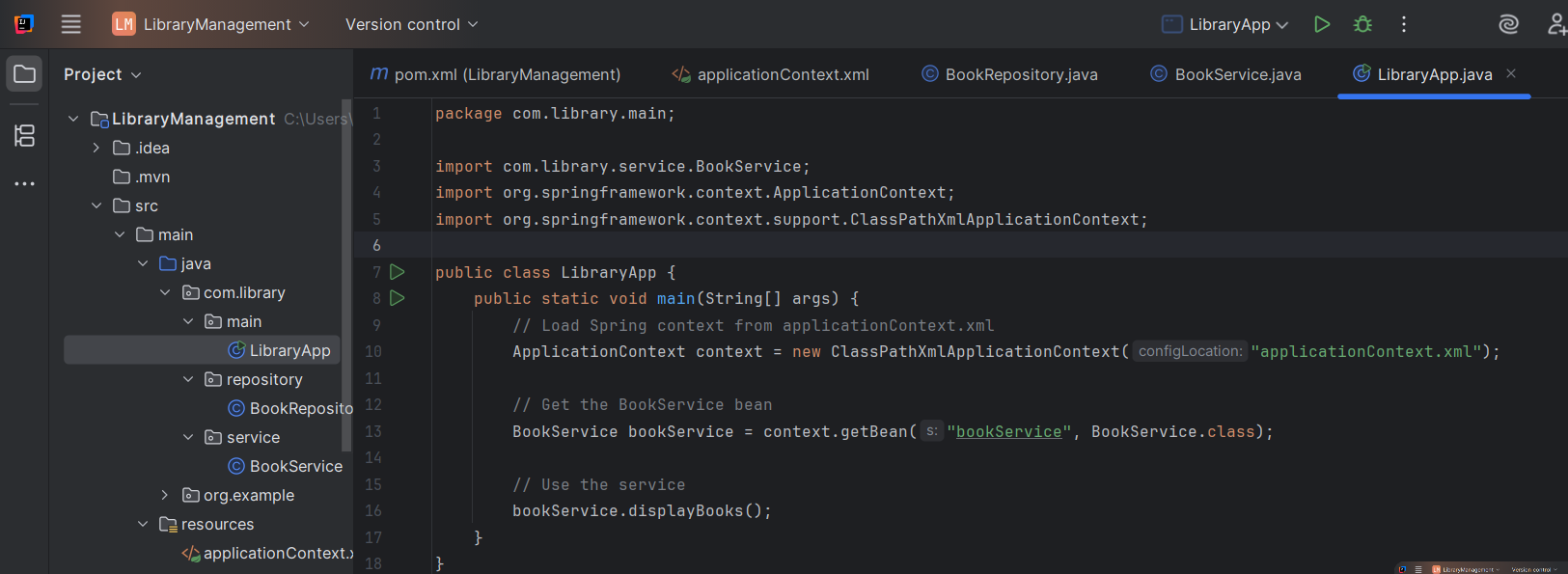


Step 2: Setter method

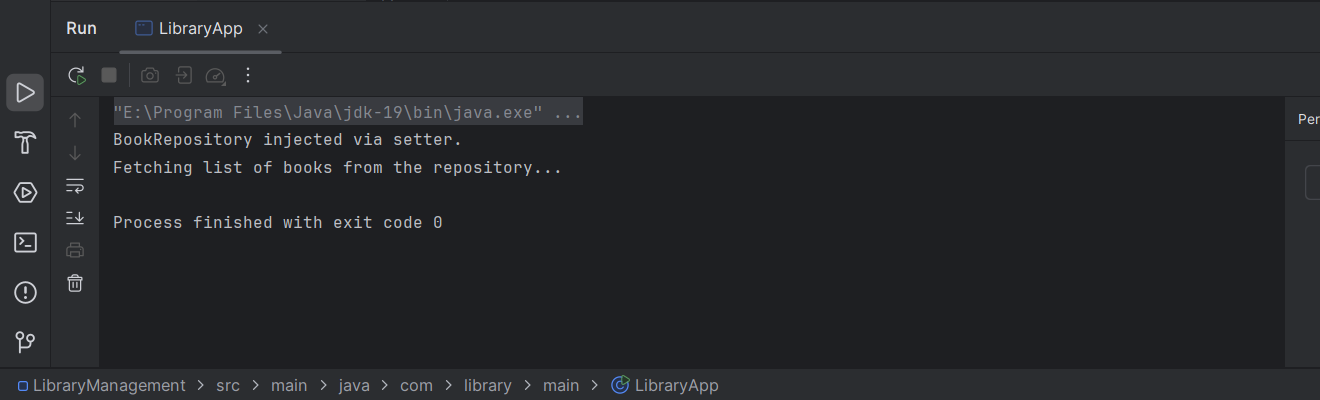




Step 3: main class



**OUTPUT:**



**Question 7**

**Exercise 7: Implementing Constructor and Setter Injection**

**Scenario:**

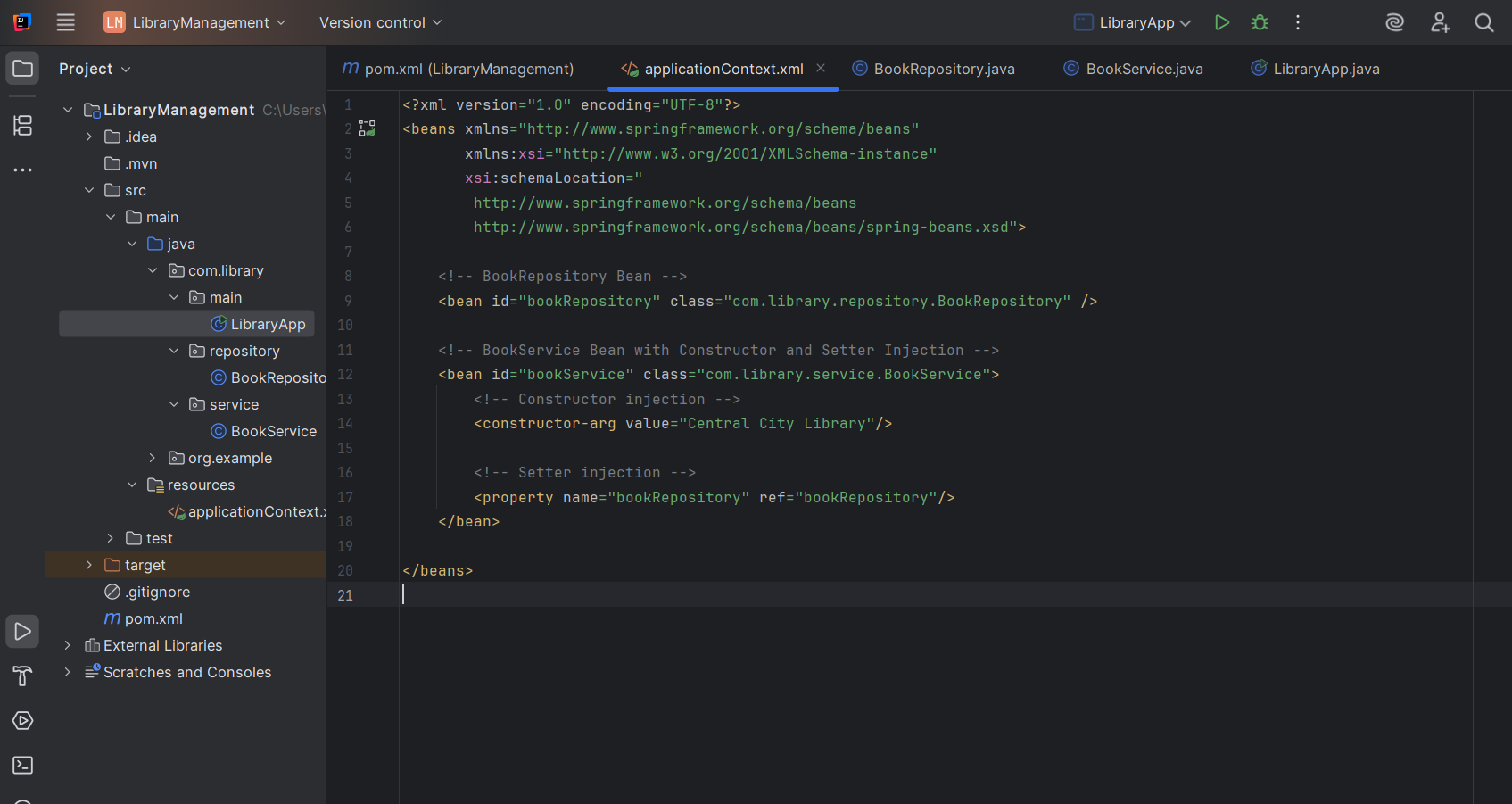
The library management application requires both constructor and setter injection for better control over bean initialization.

**Steps:**

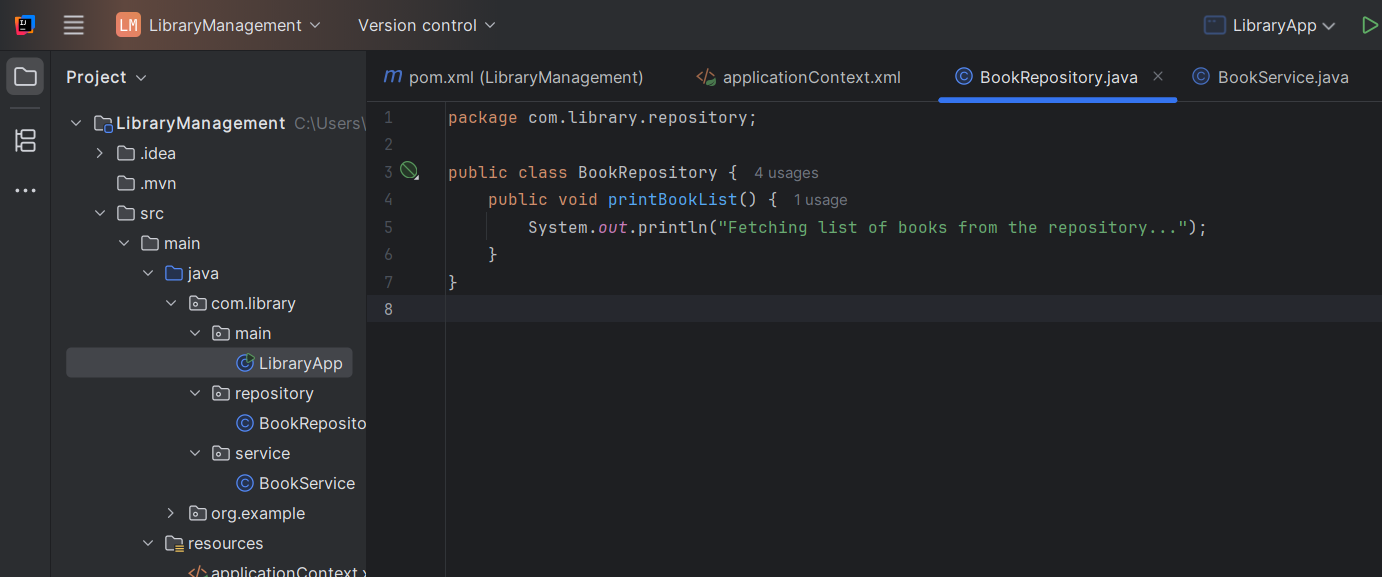
1. **Configure Constructor Injection:**
   * Update applicationContext.**xml** to configure constructor injection for **BookService**.
2. **Configure Setter Injection:**
   * Ensure that the **BookService** class has a setter method for **BookRepository** and configure it in **applicationContext.xml**.
3. **Test the Injection:**
   * Run the **LibraryManagementApplication** main class to verify both constructor and setter injection.

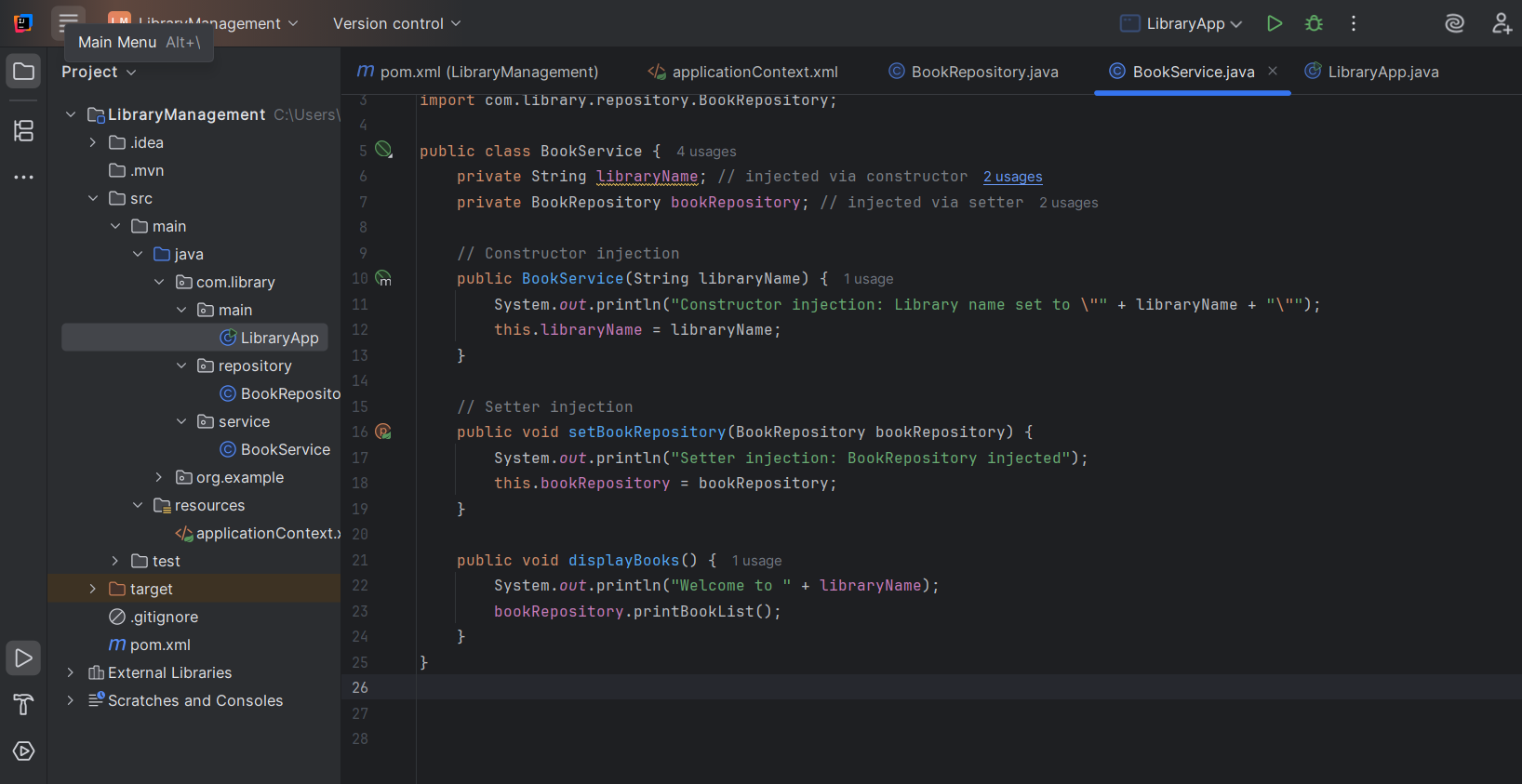
Answer and Output:

Step 1: updated xml file

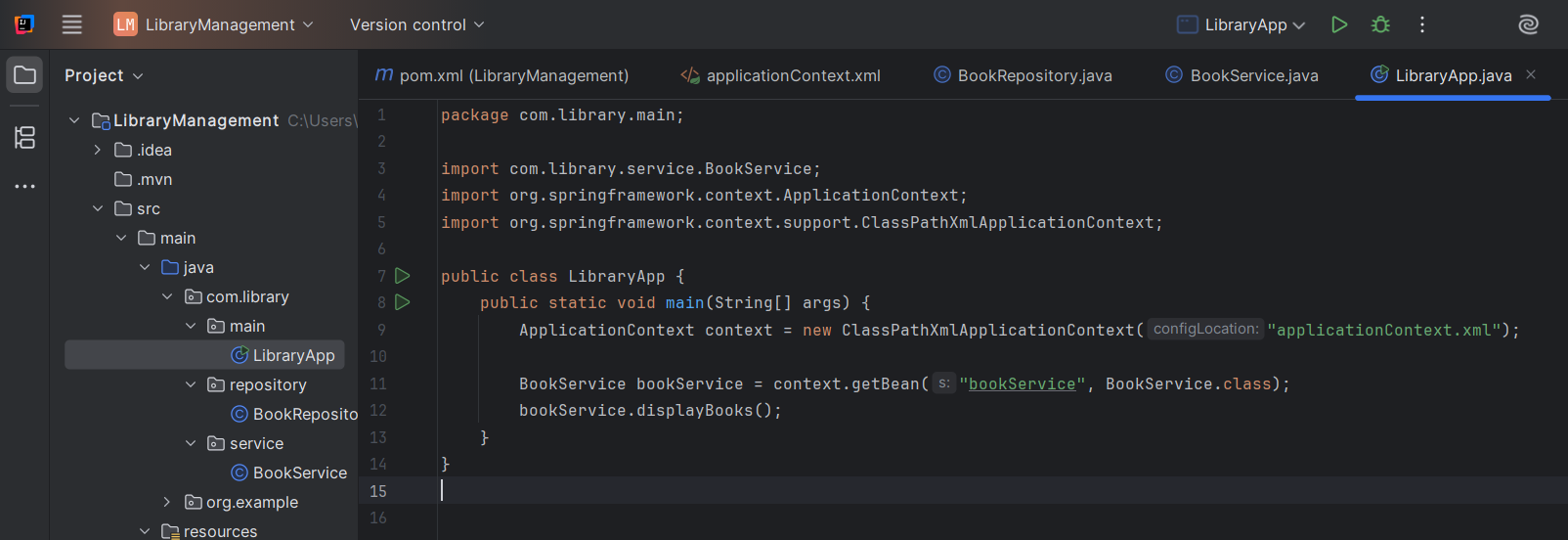


Step 2: configuring setter injection





Step 3: testing



**OUTPUT:**

