**Name:- Dharan Gowda Superset id:-6387532**

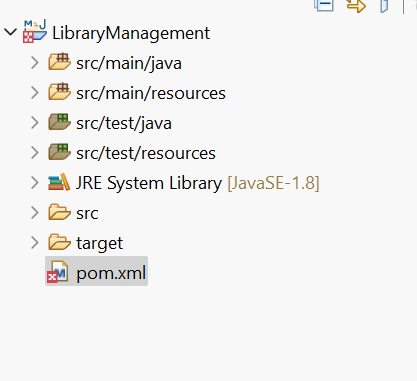
**Exercise 1: Configuring a Basic Spring Application**

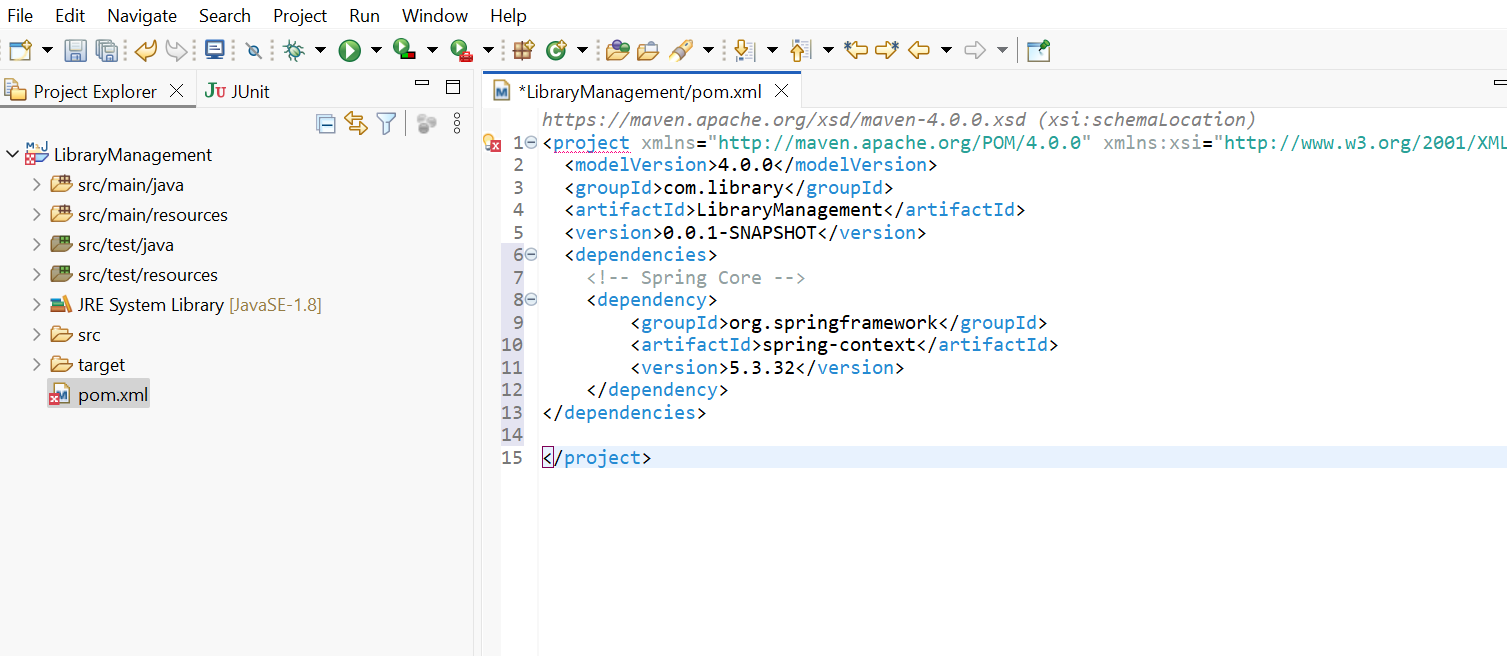
**Scenario:**

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

**Solution:-**

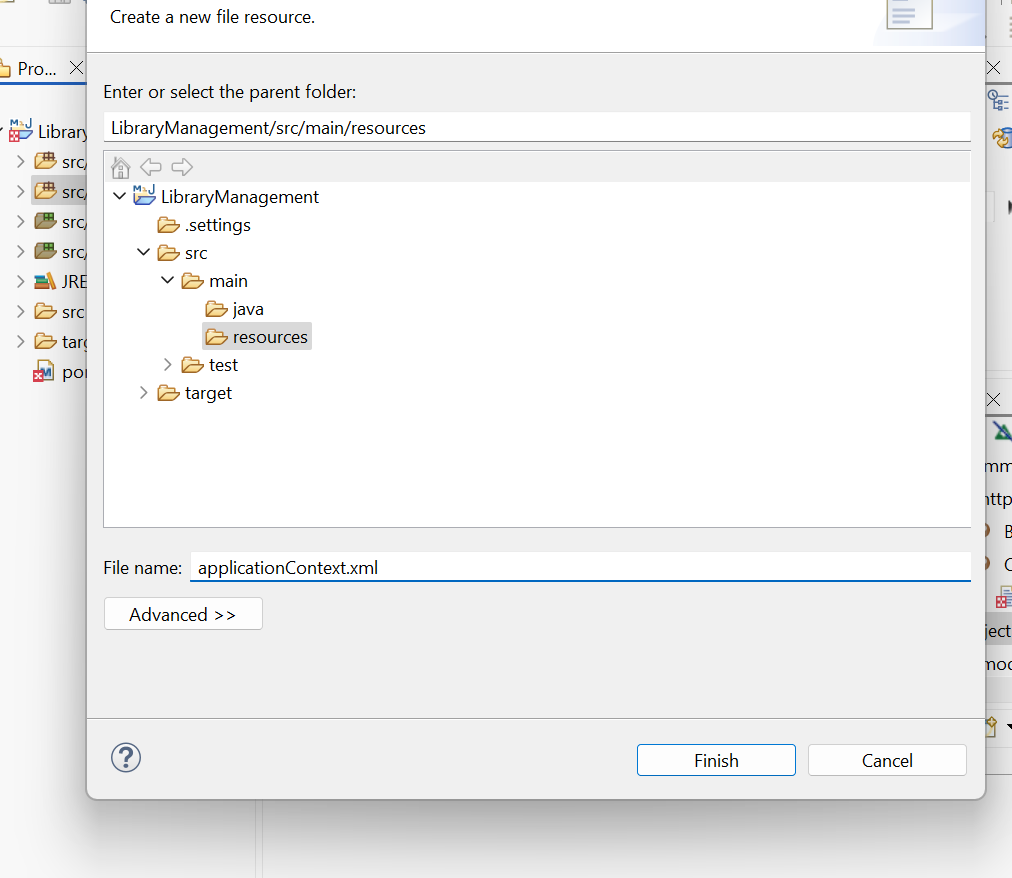
1. **Set Up a Spring Project:**
   * Create a Maven project named **LibraryManagement**.
   * Add Spring Core dependencies in the **pom.xml** file.

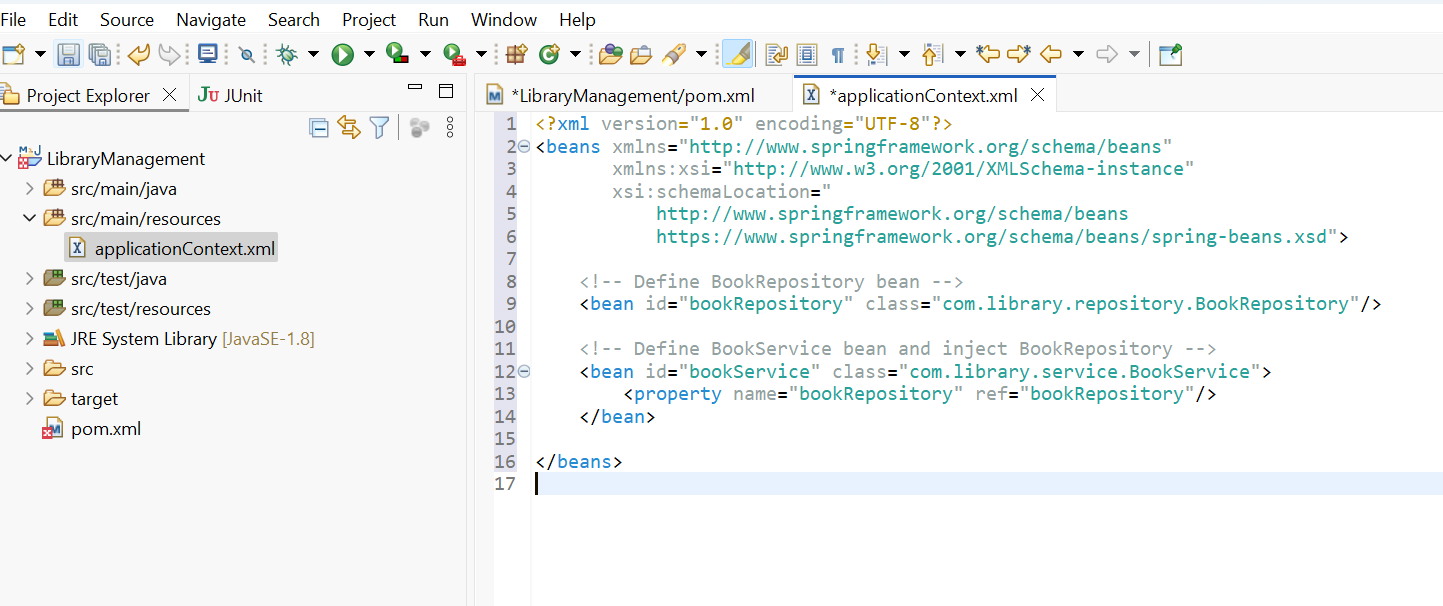
****

****

**2. Configure the Application Context:**

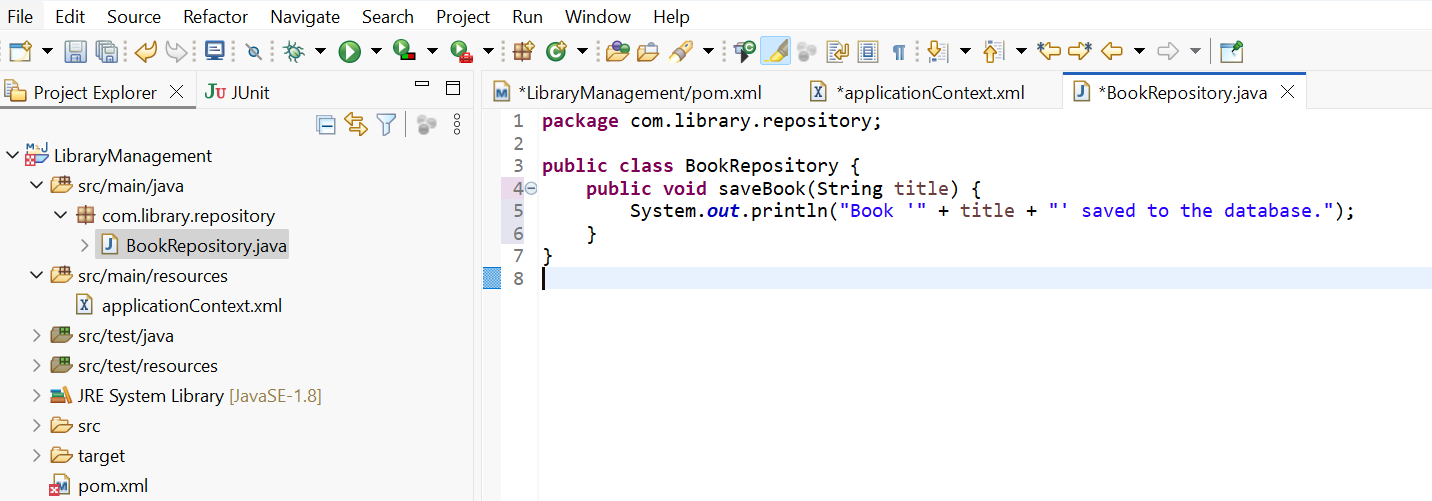
* + Create an XML configuration file named **applicationContext.xml** in the **src/main/resources** directory.
  + Define beans for **BookService** and **BookRepository** in the XML file.

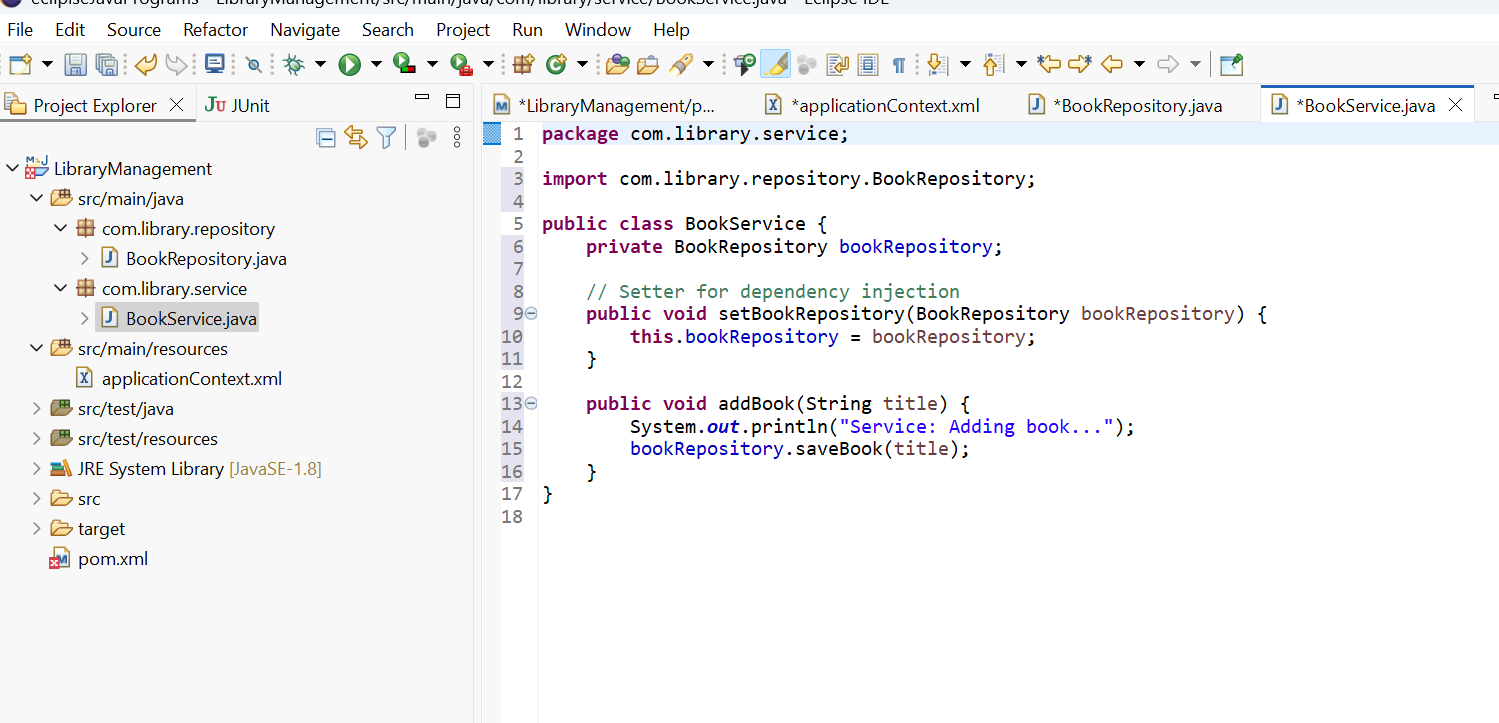
****

****

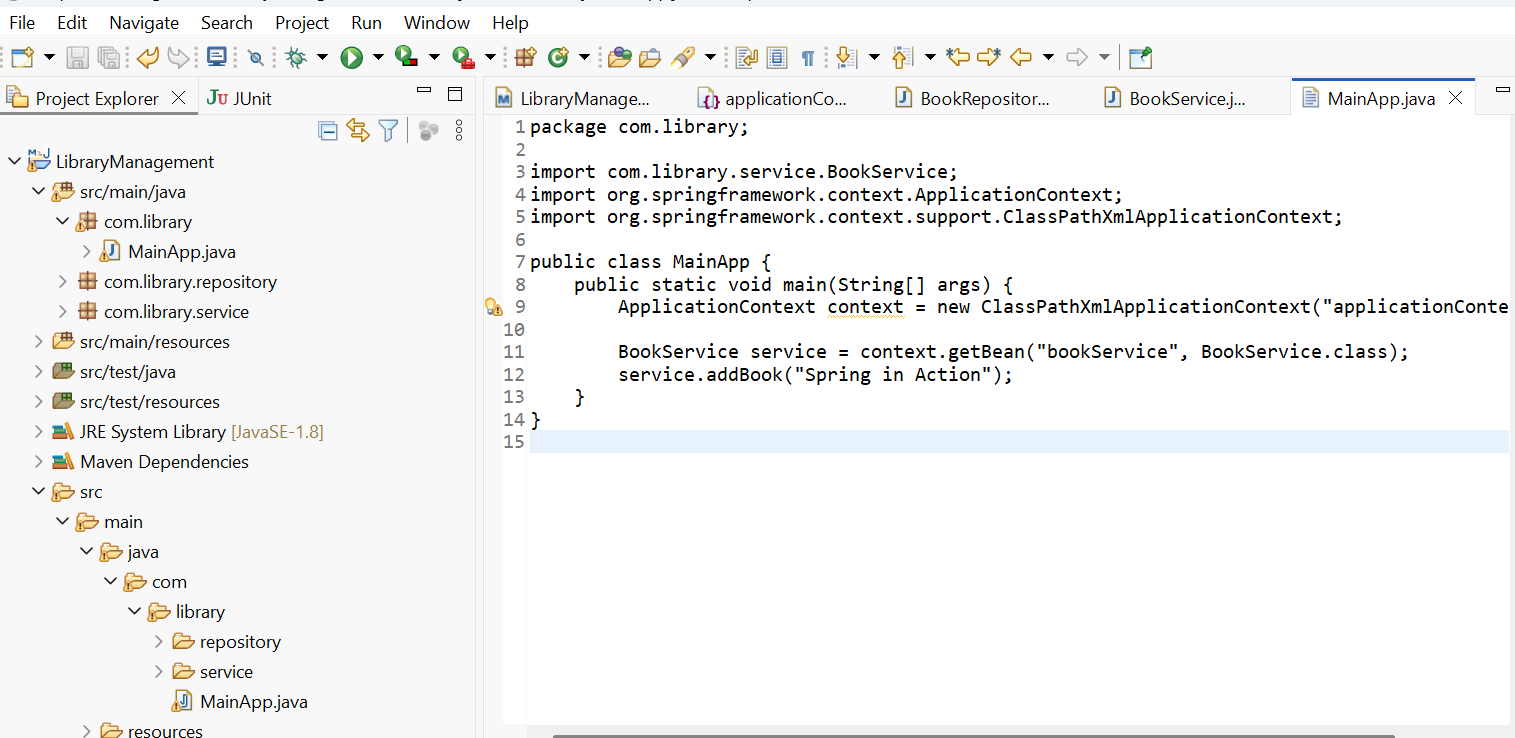
**3. Define Service and Repository Classes:**

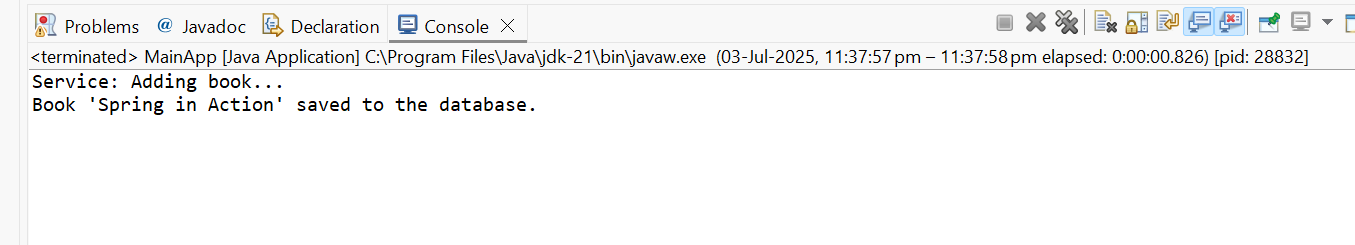
* + Create a package **com.library.service** and add a class **BookService**.
  + Create a package **com.library.repository** and add a class **BookRepository**



.

1. **Run the Application:**
   * Create a main class to load the Spring context and test the configuration.





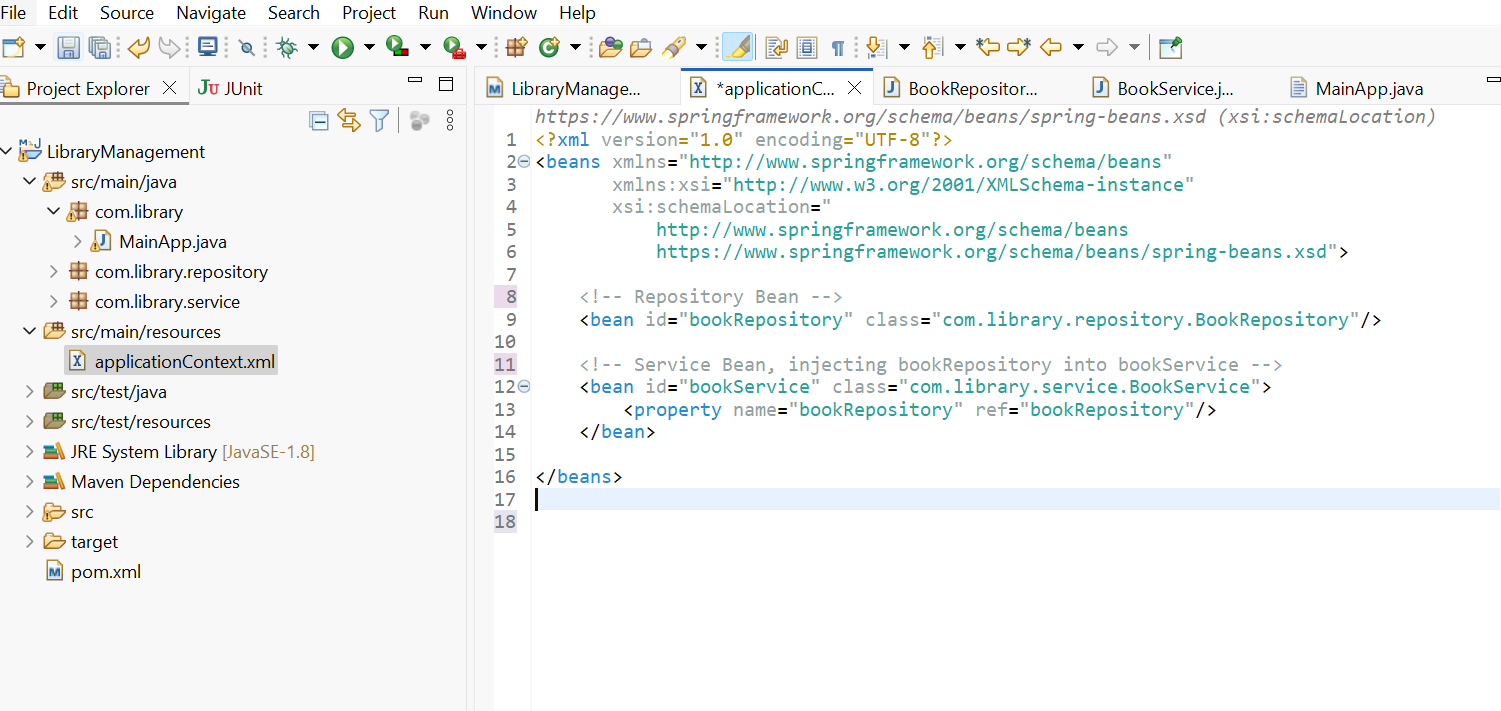
**Exercise 2: Implementing Dependency Injection**

**Scenario:**

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

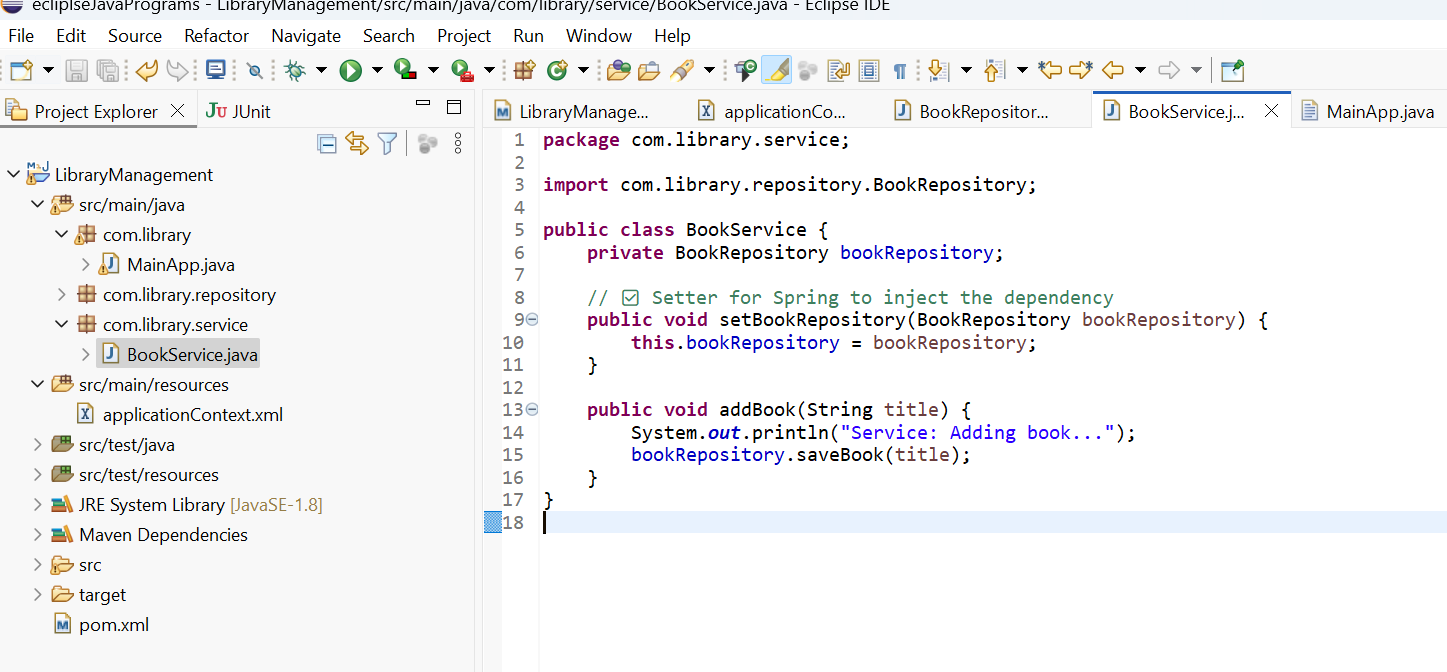
1. **Modify the XML Configuration:**

Update **applicationContext.xml** to wire **BookRepository** into **BookService**

****

1. **Update the BookService Class:**

Ensure that **BookService** class has a setter method for **BookRepository**

****

1. **Test the Configuration:**

Run the **LibraryManagementApplication** main class to verify the dependency injection

