|  |
| --- |
| *Exercise 1: Configuring a Basic Spring Application (****Solutions****)* |

### **Step 1: Set Up a Maven Project**

First, we create a Maven project named LibraryManagement.

### **Step 2: Add Spring Core Dependencies in pom.xml**

We add the necessary Spring Core dependencies to our pom.xml file. This allows Maven to download the required Spring libraries.

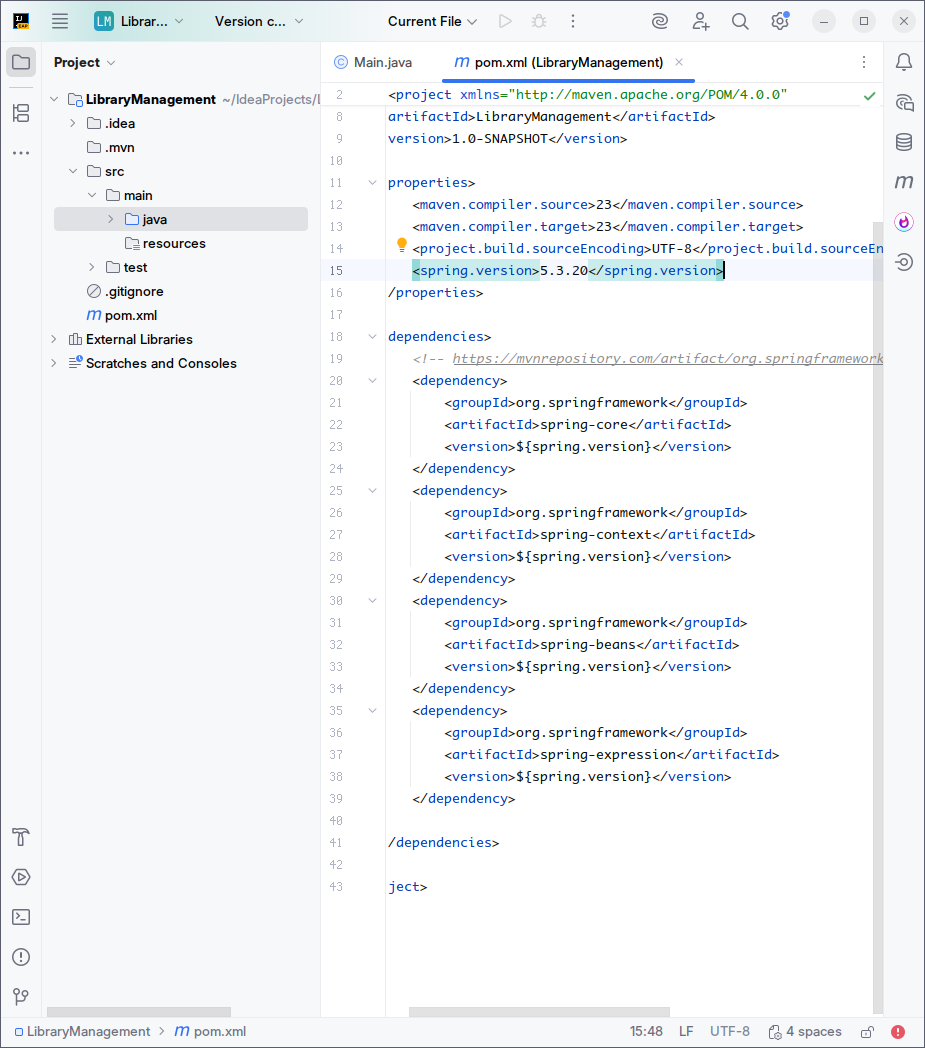
The dependencies are

**spring-core**: This is the most fundamental module. It provides the basic utilities, common data structures, and the core of the IoC container. Almost all other Spring modules depend on spring-core either directly or transitively.

**spring-beans**: This module builds on spring-core and provides the BeanFactory interface, which is a sophisticated implementation of the factory pattern. It's responsible for managing the lifecycle of beans, including their creation, configuration, and destruction.

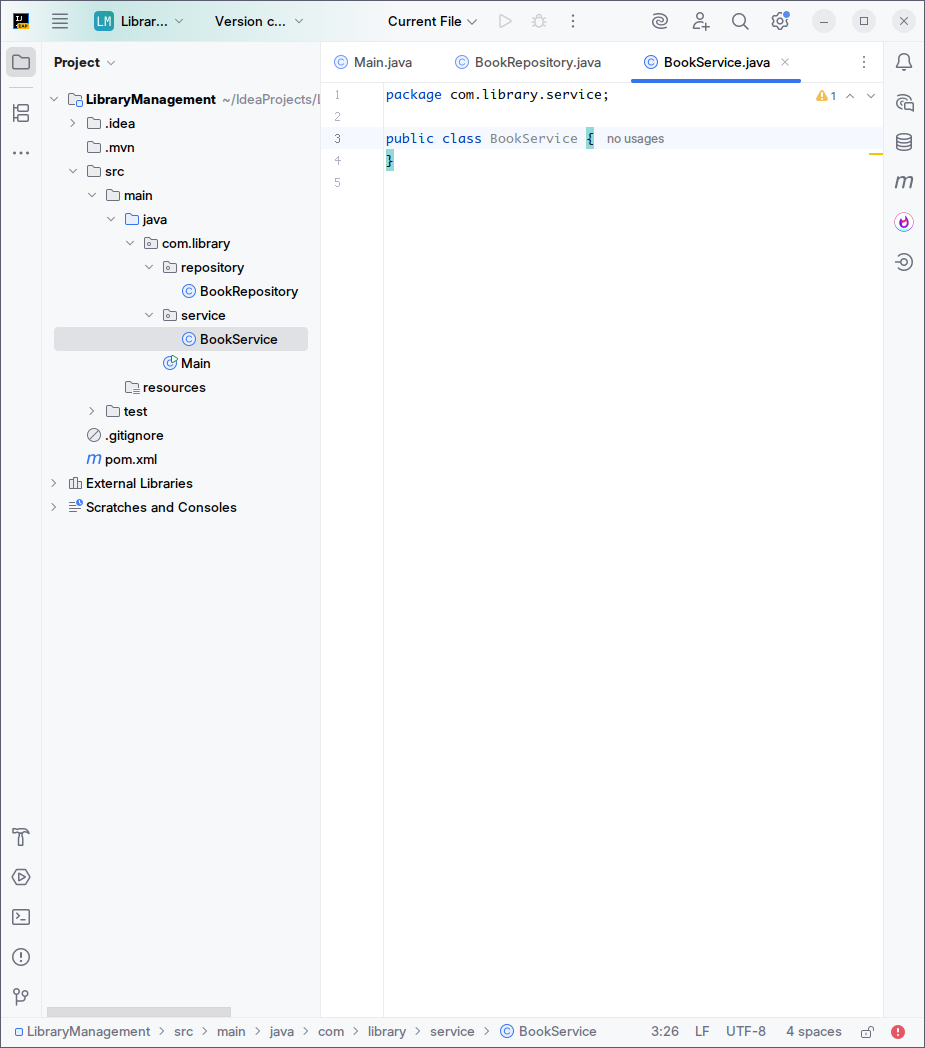
**spring-context**: This module extends the spring-beans module. It provides the ApplicationContext interface, which is an advanced container that adds enterprise-specific functionalities like internationalization (I18n), event propagation, resource loading, and transparent creation of contexts. It's the most commonly used container in Spring applications.

**spring-expression (or spring-expression-language)**: This module provides the Spring Expression Language (SpEL), a powerful expression language for querying and manipulating an object graph at runtime. While not strictly "core" in the sense of IoC/DI, it's often used in conjunction with core features for dynamic configuration



### **Step 3: Define Service and Repository Classes**

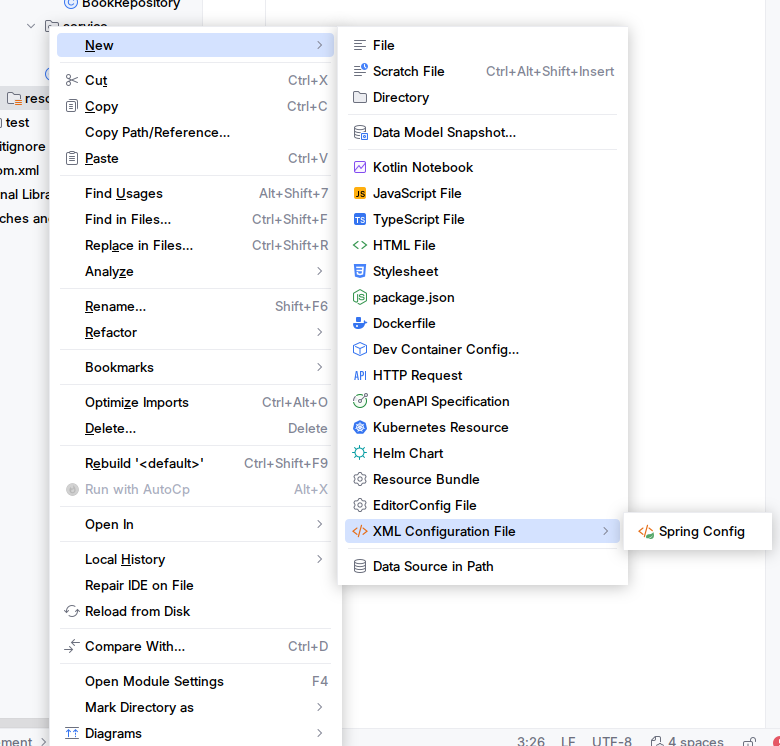
We create the BookRepository and BookService classes. The BookService will later-on depend on BookRepository.

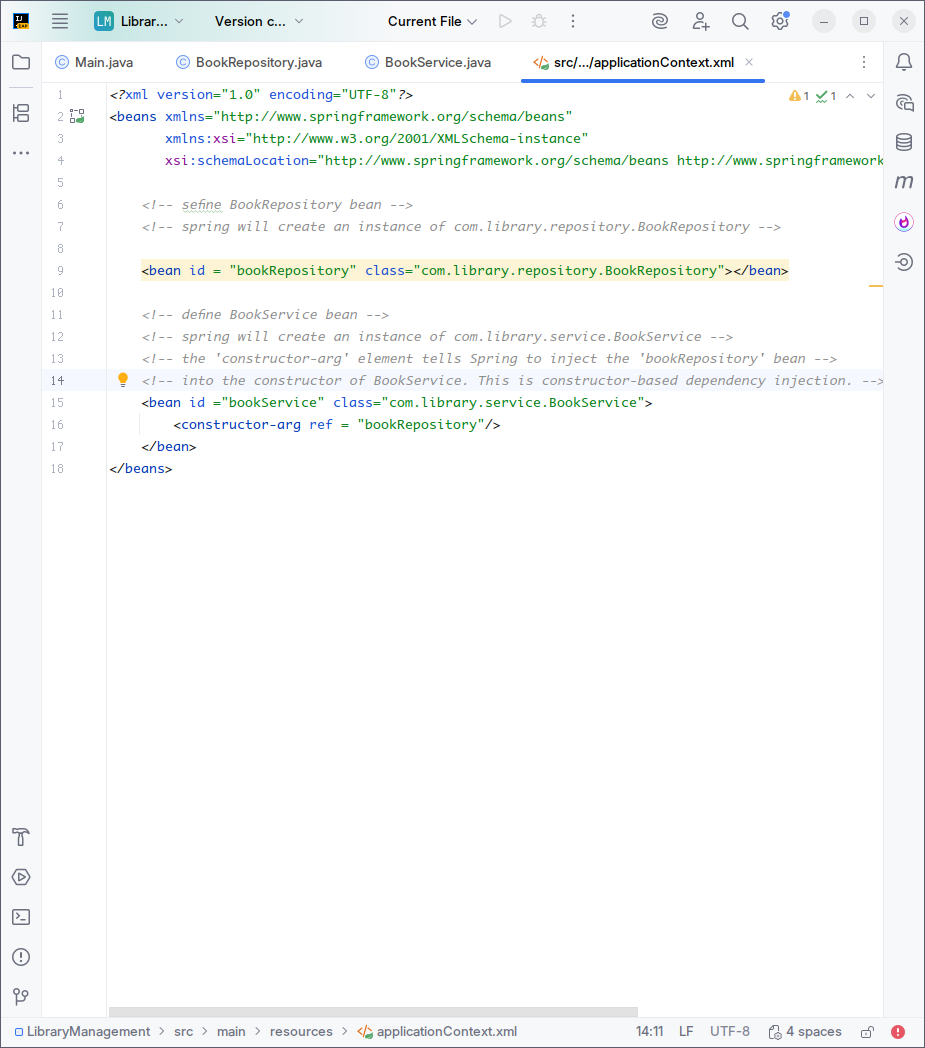


### 

### **Step 4: Configure the Application Context**

We create applicationContext.xml in src/main/resources to define our beans and manage their dependencies.





### 

### **Step 5: Run the Application**

Finally, we create a main class to load the Spring context and test our configuration.

1. Load the Spring application context from the XML file. ClassPathXmlApplicationContext looks for the XML file in the classpath (e.g., src/main/resources)
2. Retrieve the BookService bean from the context. Spring automatically resolves and injects BookRepository into BookService's constructor.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

