**Spring Core and Maven**

**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.

**Pom.xml (added dependancies highlighted)**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <name>Archetype - LibraryManagement</name>  
 <url>http://maven.apache.org</url>  
  
<dependencies>  
 **<dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.30</version> <!-- or similar -->  
 </dependency>**</dependencies>  
  
</project>

**Applicationcontext.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"/>  
</beans>

**BookRepository.java**

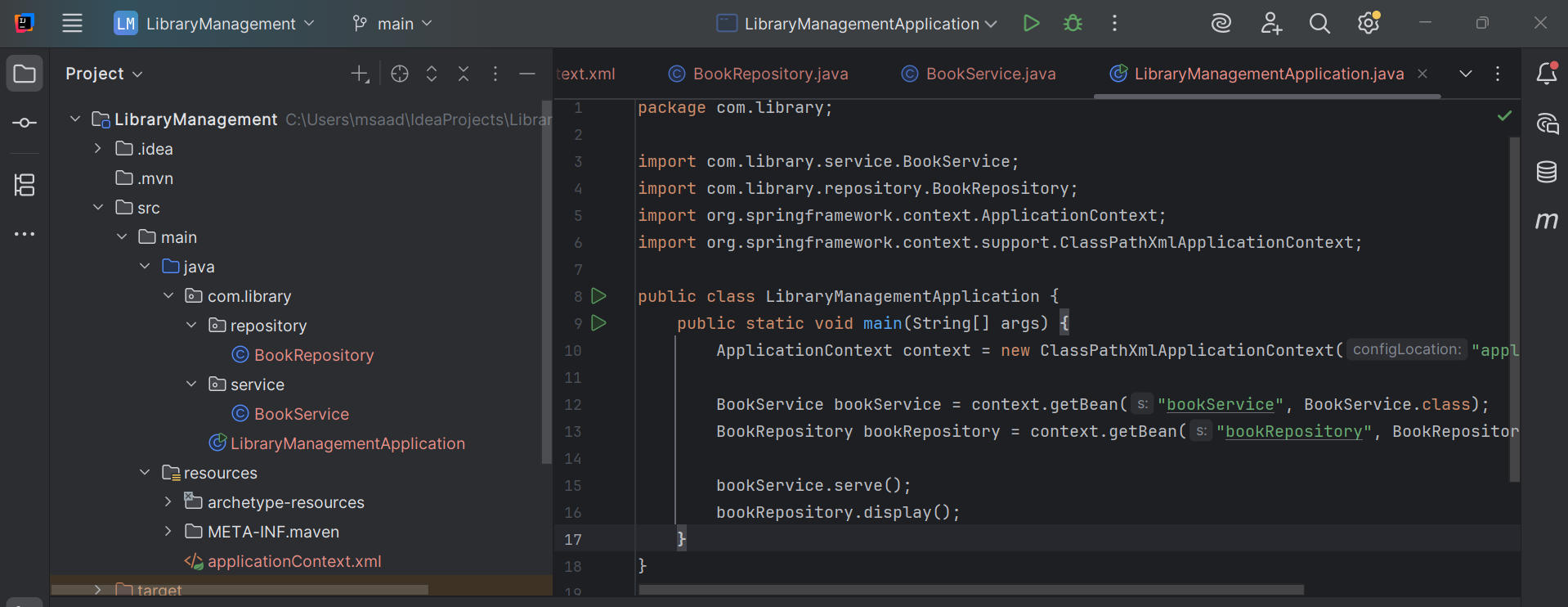
package com.library.repository;  
  
public class BookRepository {  
 public void display() {  
 System.*out*.println("BookRepository is working.");  
 }  
}

**BookService.java**

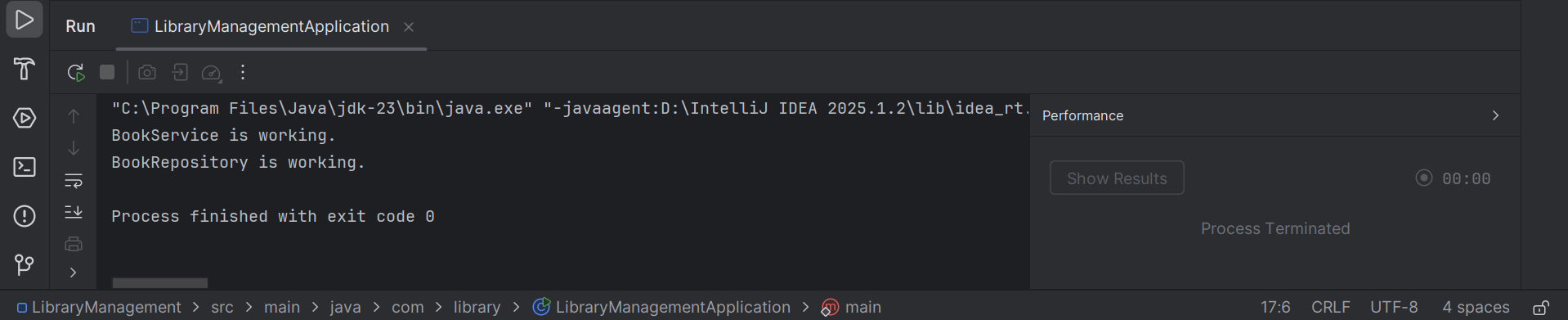
package com.library.service;  
  
public class BookService {  
 public void serve() {  
 System.*out*.println("BookService is working.");  
 }  
}

**LibraryManagementApplication.java**

package com.library;  
  
import com.library.service.BookService;  
import com.library.repository.BookRepository;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class LibraryManagementApplication {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
 BookRepository bookRepository = context.getBean("bookRepository", BookRepository.class);  
  
 bookService.serve();  
 bookRepository.display();  
 }  
}



**Output:**

****

**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.

**Update the BookService.java**

package com.library.service;  
import com.library.repository.BookRepository;  
  
public class BookService {  
  
 private BookRepository bookRepository;  
  
 // Setter for DI  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void serve() {  
 System.*out*.println("BookService is working.");  
 // Use BookRepository to confirm DI works  
 bookRepository.display();  
 }  
}

**Update the applicationContext.xml file:**

<?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">

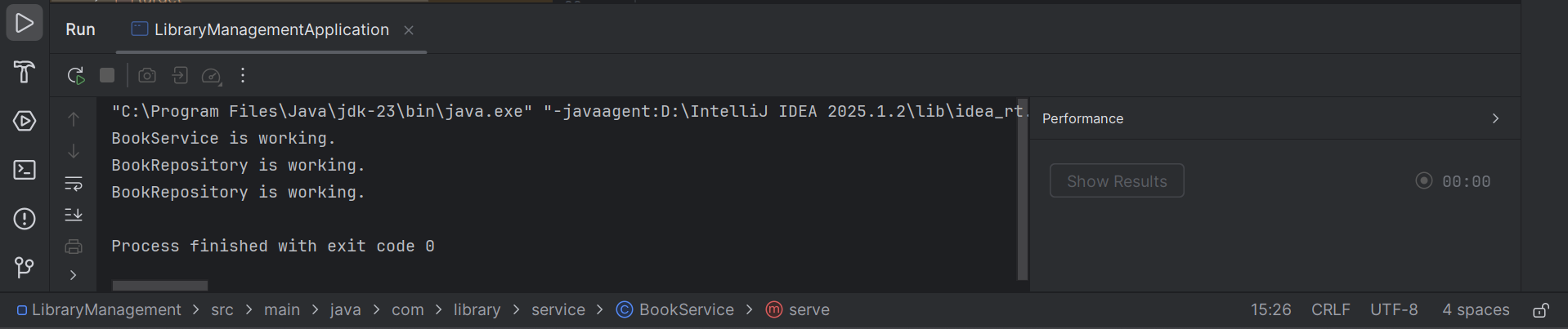
<property name="bookRepository" ref="bookRepository"/>

</bean>

</beans>

**Run the main class**

**Output:**

****

**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.

**I will add Spring AOP and Spring WebMVC dependencies to the POM file**

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.30</version> <!-- Or any stable version -->

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.30</version>

</dependency>

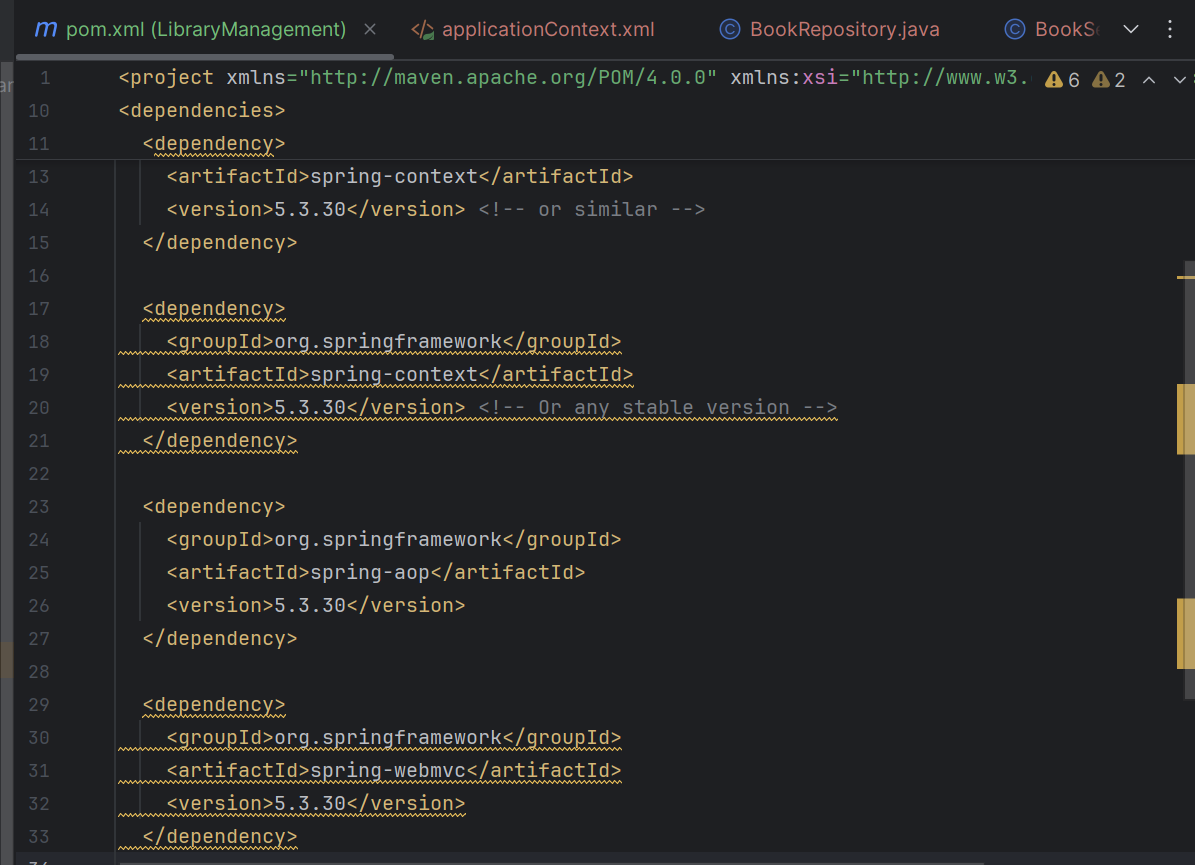
<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.30</version>

</dependency>



**Add the Maven Compiler Plugin:**

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>1.8</source> <!-- Or 17 if you're using JDK 17 -->

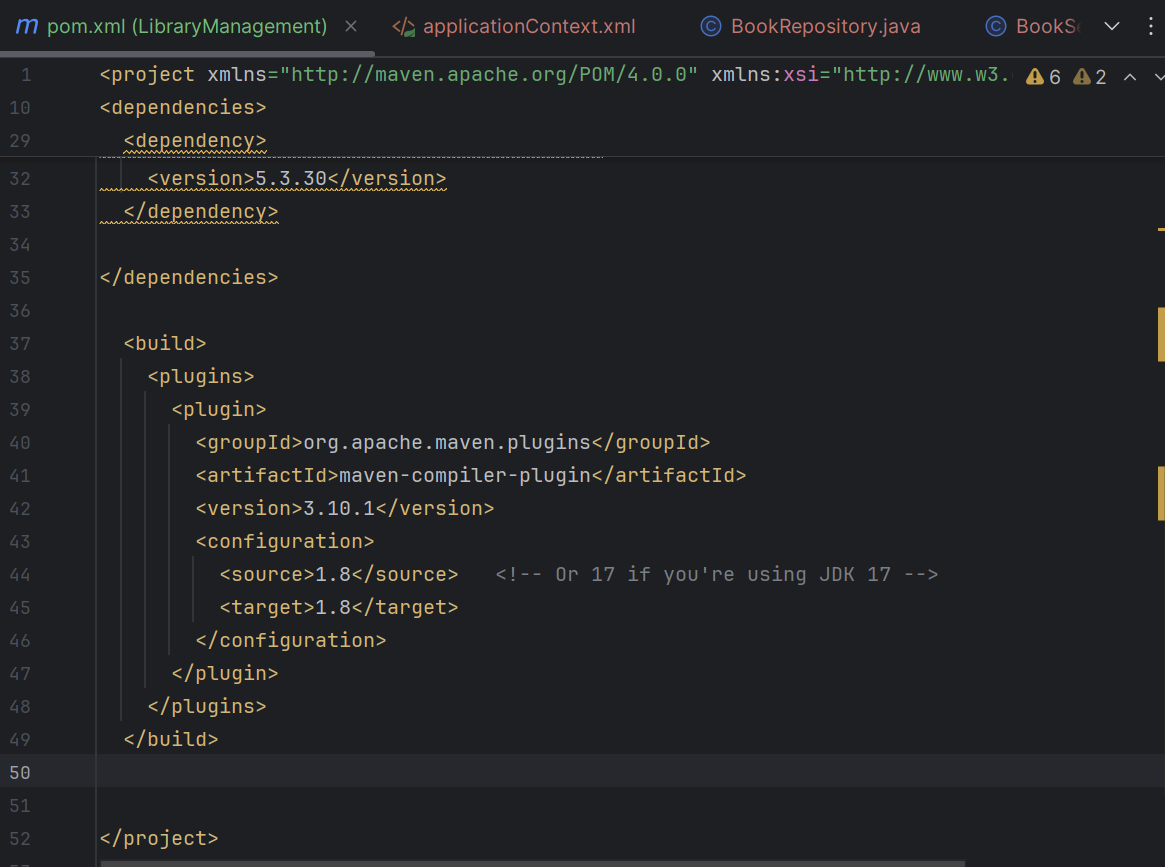
<target>1.8</target>

</configuration>

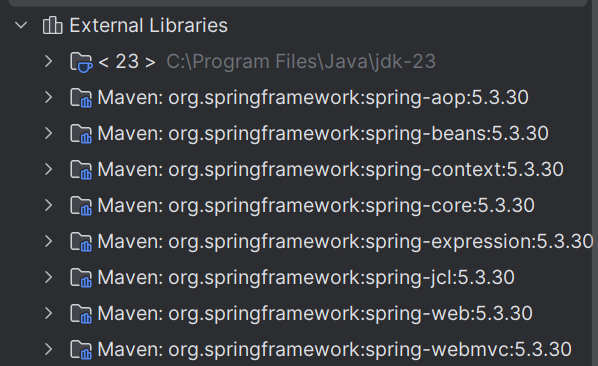
</plugin>

</plugins>

</build>



We can see that the plugins are added successfully



**The Maven project is successfully configured now**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**