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

**Steps:**

**1.Set Up a Spring Project:**

**pom.xml file:**

<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/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <dependencies>  
 <!-- Spring Core -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.36</version>  
 </dependency>  
 </dependencies>  
</project>

**2.Configure the Application Context:**

<?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">  
 </bean>  
</beans>

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

**BookService class**

package com.library.service;  
public class BookService {  
 public void serviceStatus(){  
 System.*out*.println("book service bean is ready");  
 }  
}

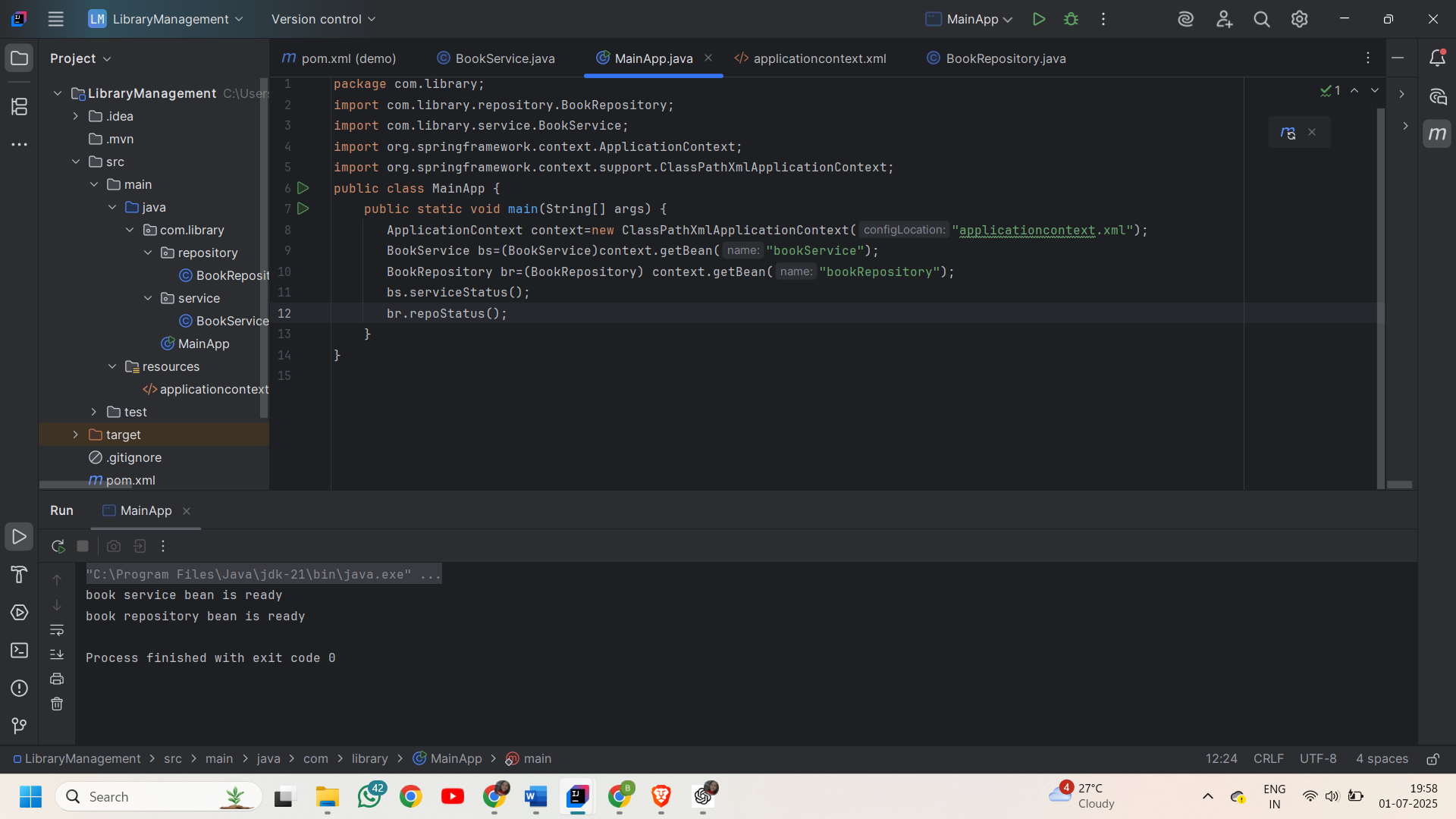
**BookRepository class**

package com.library.repository;  
public class BookRepository {  
 public void repoStatus(){  
 System.*out*.println("book repository bean is ready");  
 }  
}

**4.Run the Application:**

package com.library;  
import com.library.repository.BookRepository;  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
public class MainApp {  
 public static void main(String[] args) {  
 ApplicationContext context=new ClassPathXmlApplicationContext("applicationcontext.xml");  
 BookService bs=(BookService)context.getBean("bookService");  
 BookRepository br=(BookRepository) context.getBean("bookRepository");  
 bs.serviceStatus();  
 br.repoStatus();  
 }  
}

**Output:**



**Exercise 2: Implementing Dependency Injection**

**Steps:**

**1.Modify the XML Configuration:**

**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">  
 <property name="bookRepository" ref="bookRepository" />  
 </bean>  
</beans>

**2.Update the BookService Class:**

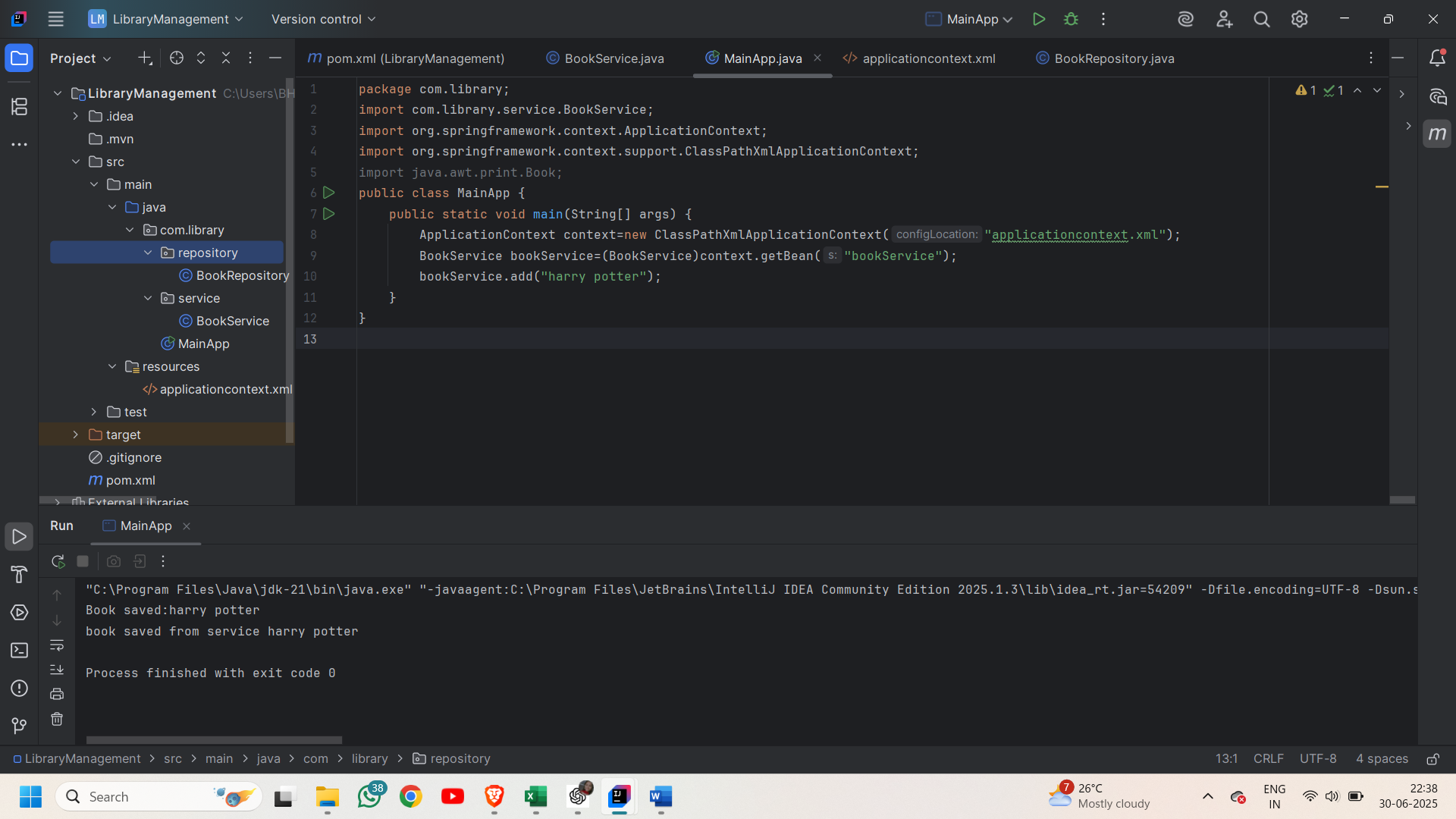
package com.library.service;  
import com.library.repository.BookRepository;  
public class BookService {  
 BookRepository bookRepository;  
 public void setBookRepository(BookRepository bookRepository){  
 this.bookRepository=bookRepository;  
 }  
 public void add(String bookname){  
 bookRepository.saveBook(bookname);  
 System.*out*.println("book saved from service "+bookname);  
 }  
}

**3.BookRepository class**

package com.library.repository;  
public class BookRepository {  
 public void saveBook(String bookName) {  
 System.*out*.println("Book saved:"+bookName);  
 }  
}

**4.Test the Configuration:**package com.library;  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import java.awt.print.Book;  
public class MainApp {  
 public static void main(String[] args) {  
 ApplicationContext context=new ClassPathXmlApplicationContext("applicationcontext.xml");  
 BookService bookService=(BookService)context.getBean("bookService");  
 bookService.add("harry potter");  
 }  
}

**Output:**



**Exercise 3: Implementing Logging with Spring AOP**

**Steps:**

**1.Add Spring AOP Dependency:**

<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/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <dependencies>  
 <!-- Spring Core -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.36</version>  
 </dependency>  
 <!-- Spring AOP -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aspects</artifactId>  
 <version>5.3.35</version>  
 </dependency>  
 </dependencies>  
</project>

**2.Create an Aspect for Logging:**

package com.library.aspect;  
import org.aspectj.lang.JoinPoint;  
import java.util.concurrent.ConcurrentHashMap;  
public class LoggingAspect{  
 ConcurrentHashMap<String,Integer> methodCallCounts=new ConcurrentHashMap<>();  
 public void logMethodCalls(JoinPoint joinPoint){  
 String methodName=joinPoint.getSignature().getName();  
 methodCallCounts.put(methodName, methodCallCounts.getOrDefault(methodName, 0) + 1);  
 int count=methodCallCounts.get(methodName);  
 System.*out*.println("[LOG] Method executed for " + count + " times");  
 }  
}

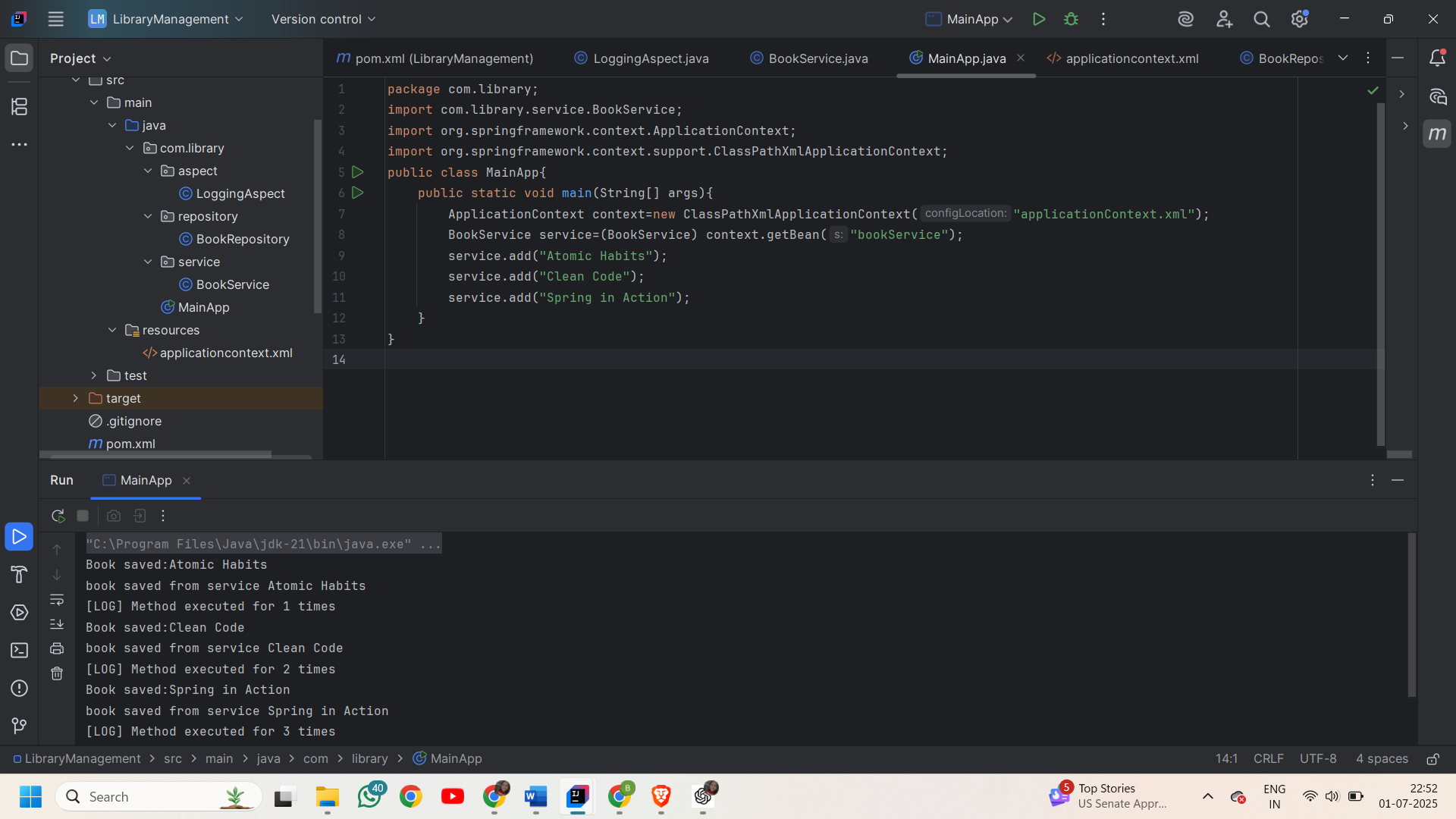
**3.Enable AspectJ Support:**

<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:aop="http://www.springframework.org/schema/aop"  
 xsi:schemaLocation="  
 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">  
<bean id="bookRepository" class="com.library.repository.BookRepository" />  
 <bean id="bookService" class="com.library.service.BookService">  
 <property name="bookRepository" ref="bookRepository" />  
 </bean>  
 <bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />  
 <aop:config>  
 <aop:aspect ref="loggingAspect">  
 <aop:after method="logMethodCalls"  
 pointcut="execution(\* com.library.service.\*.\*(..))"/>  
 </aop:aspect>  
 </aop:config>  
</beans>

**4.Test the Aspect:**

package com.library;  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
public class MainApp{  
 public static void main(String[] args){  
 ApplicationContext context=new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService service=(BookService) context.getBean("bookService");  
 service.add("Atomic Habits");  
 service.add("Clean Code");  
 service.add("Spring in Action");  
 }  
}

**Output:**



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

**Steps:**

**1.Create a New Maven Project:**

Created a new Maven project named **LibraryManagement**.

**2.Add Spring Dependencies in pom.xml:**

<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/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <properties>  
 <maven.compiler.source>1.8</maven.compiler.source>  
 <maven.compiler.target>1.8</maven.compiler.target>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.36</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aop</artifactId>  
 <version>5.3.36</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>5.3.36</version>  
 </dependency>  
 <dependency>  
 <groupId>org.aspectj</groupId>  
 <artifactId>aspectjweaver</artifactId>  
 <version>1.9.20.1</version>  
 </dependency>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
 </project>

**3.Configure Maven Plugins:**

Configure the Maven Compiler Plugin for Java version 1.8 in the pom.xml file.

<build>  
 <plugins>  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <source>1.8</source>  
 <target>1.8</target>  
 </configuration>  
 </plugin>  
 </plugins>  
 </build>

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

**Steps:**

**1.Create Spring Configuration File:**

<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>

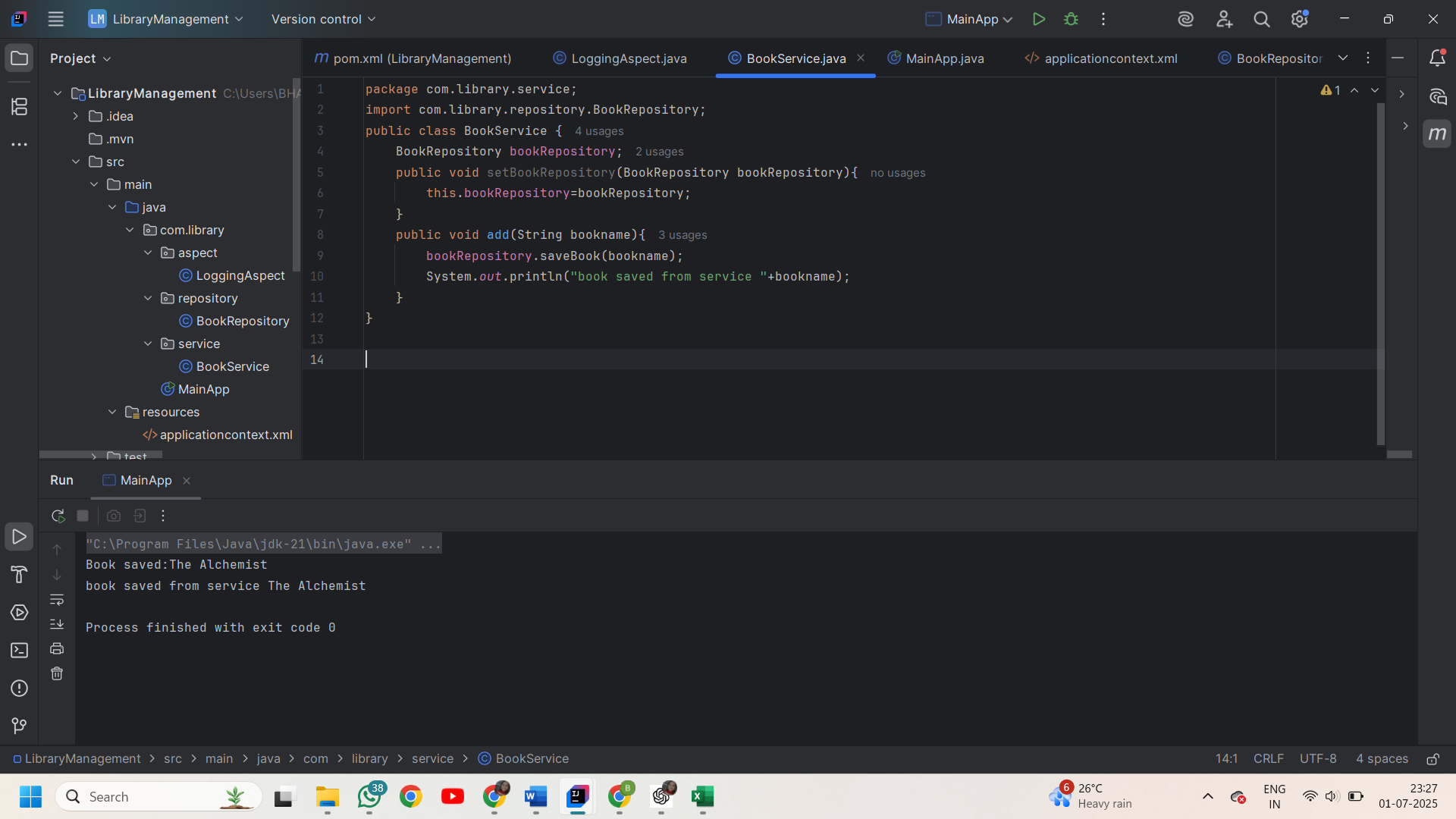
2.**Update the BookService Class:**

package com.library.service;  
import com.library.repository.BookRepository;  
public class BookService {  
 BookRepository bookRepository;  
 public void setBookRepository(BookRepository bookRepository){  
 this.bookRepository=bookRepository;  
 }  
 public void add(String bookname){  
 bookRepository.saveBook(bookname);  
 System.*out*.println("book saved from service "+bookname);  
 }  
}

**3.Run the Application:**

package com.library;  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
public class MainApp{  
 public static void main(String[] args){  
 ApplicationContext context=new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService service=context.getBean(BookService.class);  
 service.add("The Alchemist");  
 }  
}

**Output:**



**Exercise 6: Configuring Beans with Annotations**

**Steps:**

**1.Enable Component Scanning:**

<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="  
 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">  
 <context:component-scan base-package="com.library" />  
</beans>

**2.Annotate Classes:**

**@Service** annotation for the **BookService** class.

package com.library.service;  
import com.library.repository.BookRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
@Service  
public class BookService {  
 @Autowired  
 BookRepository bookRepository;  
 public void setBookRepository(BookRepository bookRepository){  
 this.bookRepository=bookRepository;  
 }  
 public void add(String bookname){  
 bookRepository.saveBook(bookname);  
 System.*out*.println("book saved from service "+bookname);  
 }  
}

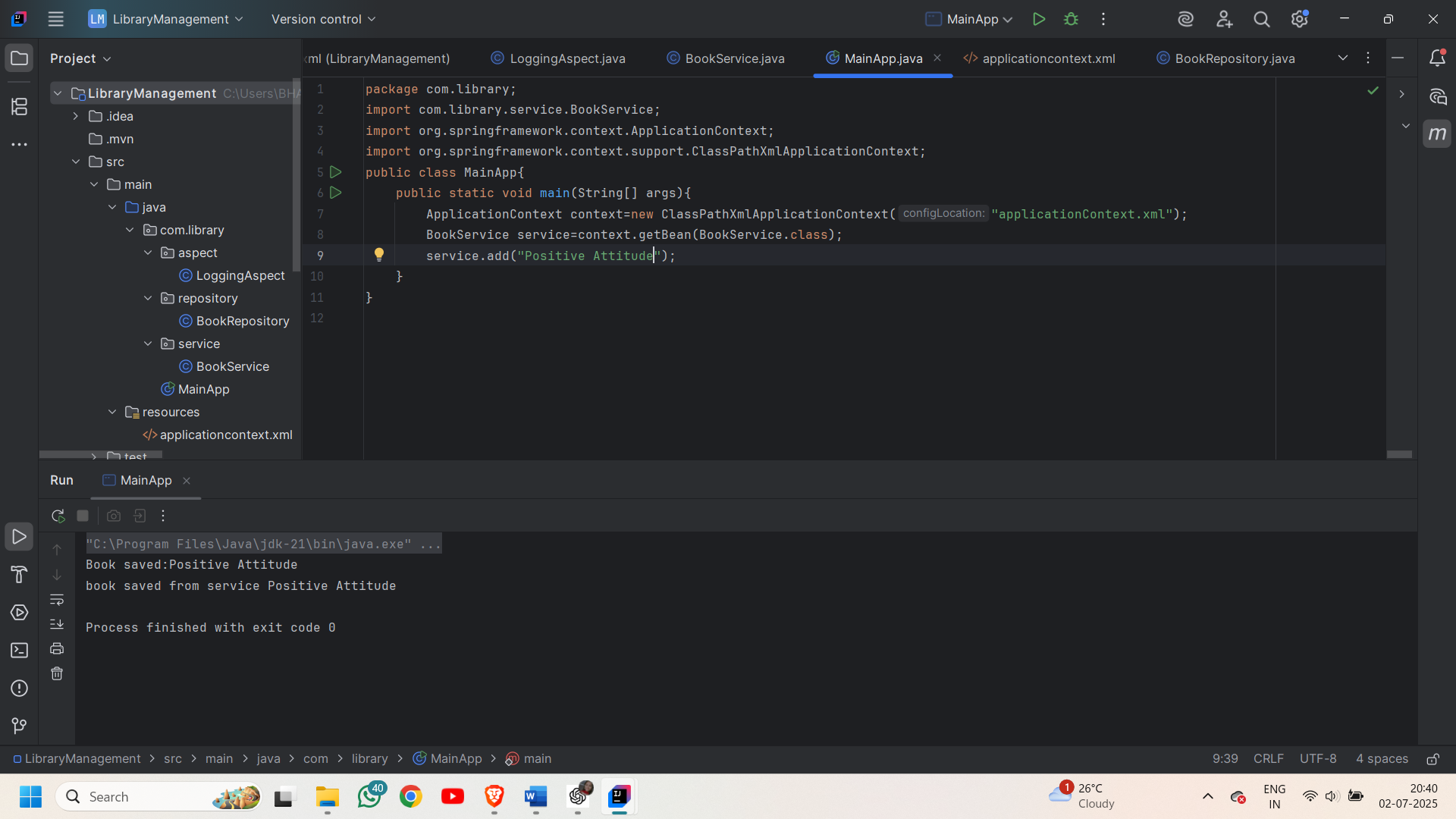
**@Repository** annotation for the **BookRepository** class.

package com.library.repository;  
import org.springframework.stereotype.Repository;  
@Repository  
public class BookRepository {  
 public void saveBook(String bookName) {  
 System.*out*.println("Book saved:"+bookName);  
 }  
}

**3.Test the Configuration:**

package com.library;  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
public class MainApp{  
 public static void main(String[] args){  
 ApplicationContext context=new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService service=context.getBean(BookService.class);  
 service.add("Positive Attitude");  
 }  
}

**Output:**



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

**Steps:**

**1.Configure Constructor Injection:**

<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="  
 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">  
 <bean id="bookService" class="com.library.service.BookService">  
 <constructor-arg value="1"> </constructor-arg>  
 <property name="bookRepository" ref="bookRepository"/>  
 </bean>  
 <bean id="bookRepository" class="com.library.repository.BookRepository"/>  
</beans>

**2.Configure Setter Injection:**

package com.library.service;  
import com.library.repository.BookRepository;  
  
public class BookService {  
 int id;  
 BookService(int id){  
 this.id=id;  
 System.*out*.println("Constructor Injection in bookservice class");  
 }  
 BookRepository bookRepository;  
 public void setBookRepository(BookRepository bookRepository){  
 System.*out*.println("setter injection");  
 this.bookRepository=bookRepository;  
 }  
 public void add(String bookname){  
 bookRepository.saveBook(bookname);  
 System.*out*.println("book saved from service "+bookname);  
 }  
}

**3.Test the Injection:**

package com.library;  
import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
public class MainApp{  
 public static void main(String[] args){  
 ApplicationContext context=new ClassPathXmlApplicationContext("applicationContext.xml");  
 BookService service=context.getBean(BookService.class);  
 service.add("Positive Attitude");  
 }  
}

**Output:**

