**Exercise 8: Implementing Basic AOP with Spring**

**Scenario:**

The library management application requires basic AOP functionality to separate cross-cutting concerns like logging and transaction management.

**LibraryManagementApplication.java :-**

package org.example1;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.example1.service.BookService;  
  
public class LibraryManagementApplication {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
  
 // Call service methods to trigger AOP  
 bookService.addBook("The Art of Computer Programming");  
 bookService.listBooks();  
 }  
}

**LoggingAspect.java :-**

package org.example1.aspect;  
  
import org.aspectj.lang.annotation.After;  
import org.aspectj.lang.annotation.Aspect;  
import org.aspectj.lang.annotation.Before;  
import org.springframework.stereotype.Component;  
  
@Aspect  
@Component  
public class LoggingAspect {  
  
 @Before("execution(\* org.example1.service.BookService.\*(..))")  
 public void logBefore() {  
 System.*out*.println("[LOG] Method execution starting...");  
 }  
  
 @After("execution(\* org.example1.service.BookService.\*(..))")  
 public void logAfter() {  
 System.*out*.println("[LOG] Method execution completed.");  
 }  
}

**BookController.java :-**

package org.example1.controller;  
  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class BookController {  
 @GetMapping("/")  
 public String home() {  
 return "Welcome to the Library Management System!";  
 }  
}

**Book.java :-**

package org.example1.model;  
  
public class Book {  
 private Long id;  
 private String title;  
  
 public Book() {}  
  
 public Book(Long id, String title) {  
 this.id = id;  
 this.title = title;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
}

**BookRepository.java :-**

package org.example1.repository;  
  
public interface BookRepository {  
 // For future database extension  
}

**LibraryManagementApplication.java :-**

package org.example1;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.example1.service.BookService;  
  
public class LibraryManagementApplication {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
  
 // Call service methods to trigger AOP  
 bookService.addBook("The Art of Computer Programming");  
 bookService.listBooks();  
 }  
**}**

**BookService.java :-**

package org.example1.service;  
  
import org.springframework.stereotype.Component;  
  
import java.util.ArrayList;  
import java.util.List;  
  
@Component("bookService")  
public class BookService {  
 private List<String> books = new ArrayList<>();  
  
 public void addBook(String title) {  
 books.add(title);  
 System.*out*.println("Book added: " + title);  
 }  
  
 public List<String> listBooks() {  
 System.*out*.println("Listing books:");  
 books.forEach(System.*out*::println);  
 return books;  
 }  
}

**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"  
 xmlns:context="http://www.springframework.org/schema/context"  
 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/context http://www.springframework.org/schema/context/spring-context.xsd  
 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">  
  
 <!-- Component scanning -->  
 <context:component-scan base-package="org.example1" />  
  
 <!-- Enable AOP -->  
 <aop:aspectj-autoproxy />  
  
</beans>

**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>org.example1</groupId>  
 <artifactId>LibraryManagementSystem</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <packaging>war</packaging>  
  
 <properties>  
 <java.version>1.8</java.version>  
 <spring.version>5.3.30</spring.version>  
 <aspectj.version>1.9.21</aspectj.version>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
  
 <dependencies>  
  
 <!-- Spring Context -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <!-- Spring Web MVC -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <!-- Spring AOP -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aop</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <!-- AspectJ Runtime -->  
 <dependency>  
 <groupId>org.aspectj</groupId>  
 <artifactId>aspectjrt</artifactId>  
 <version>${aspectj.version}</version>  
 </dependency>  
  
 <!-- AspectJ Weaver -->  
 <dependency>  
 <groupId>org.aspectj</groupId>  
 <artifactId>aspectjweaver</artifactId>  
 <version>${aspectj.version}</version>  
 </dependency>  
  
 <!-- Servlet API -->  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>javax.servlet-api</artifactId>  
 <version>4.0.1</version>  
 <scope>provided</scope>  
 </dependency>  
  
 <!-- JSTL (optional if using JSP) -->  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>jstl</artifactId>  
 <version>1.2</version>  
 </dependency>  
  
 <!-- Logging -->  
 <dependency>  
 <groupId>commons-logging</groupId>  
 <artifactId>commons-logging</artifactId>  
 <version>1.2</version>  
 </dependency>  
  
 </dependencies>  
  
 <build>  
 <finalName>LibraryManagementSystem</finalName>  
 <plugins>  
  
 <!-- Compiler plugin -->  
 <plugin>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <source>${java.version}</source>  
 <target>${java.version}</target>  
 </configuration>  
 </plugin>  
  
 <!-- WAR packaging plugin -->  
 <plugin>  
 <artifactId>maven-war-plugin</artifactId>  
 <version>3.3.2</version>  
 <configuration>  
 <failOnMissingWebXml>false</failOnMissingWebXml>  
 </configuration>  
 </plugin>  
  
 </plugins>  
 </build>  
  
</project>

**Output :-**

