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

**Book.java**

package com.library.model;

public class Book {

private int id;

private String title;

private boolean available = true;

private int daysBorrowed = 0;

public Book() {}

public Book(int id, String title) {

this.id = id;

this.title = title;

}

public int getId() { return id; }

public String getTitle() { return title; }

public boolean isAvailable() { return available; }

public int getDaysBorrowed() { return daysBorrowed; }

public void setAvailable(boolean available) { this.available = available; }

public void setDaysBorrowed(int daysBorrowed) { this.daysBorrowed = daysBorrowed; }

@Override

public String toString() {

return "[" + id + "] " + title + (available ? " (Available)" : " (Borrowed for " + daysBorrowed + " days)");

}

}

**BookRepository.java**

package com.library.repository;

import com.library.model.Book;

import java.util.List;

public interface BookRepository {

List<Book> findAll();

}

**LibraryManagementApplication.java**

package com.library.main;

import com.library.service.LibraryService;

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");

LibraryService service = context.getBean("libraryService", LibraryService.class);

service.listAvailableBooks();

service.borrowBook(2, 10);

service.showBorrowedBooksAndFine();

service.listAvailableBooks();

}

}

**LibraryService.java**

package com.library.service;

public interface LibraryService {

void listAvailableBooks();

void borrowBook(int id, int days);

void showBorrowedBooksAndFine();

}

**LibraryServiceImpl.java**

package com.library.service;

import com.library.model.Book;

import com.library.repository.BookRepository;

public class LibraryServiceImpl implements LibraryService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

@Override

public void listAvailableBooks() {

for (Book book : bookRepository.findAll()) {

if (book.isAvailable()) {

System.out.println(book);

}

}

}

@Override

public void borrowBook(int id, int days) {

for (Book book : bookRepository.findAll()) {

if (book.getId() == id) {

if (book.isAvailable()) {

book.setAvailable(false);

book.setDaysBorrowed(days);

System.out.println("You borrowed: " + book.getTitle());

} else {

System.out.println("Already borrowed.");

}

return;

}

}

System.out.println("Book ID not found.");

}

@Override

public void showBorrowedBooksAndFine() {

for (Book book : bookRepository.findAll()) {

if (!book.isAvailable()) {

int fine = Math.max(0, book.getDaysBorrowed() - 7) \* 5;

System.out.println(book + " | Fine: ₹" + fine);

}

}

}

}

**LoggingAspect.java**

package com.library.aspect;

import org.aspectj.lang.ProceedingJoinPoint;

import org.aspectj.lang.annotation.\*;

import org.springframework.stereotype.Component;

@Aspect

public class LoggingAspect {

@Around("execution(\* com.library.service.LibraryServiceImpl.\*(..))")

public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

long start = System.nanoTime();

Object result = joinPoint.proceed();

long end = System.nanoTime();

System.out.println("Executed " + joinPoint.getSignature().getName() +

" in " + (end - start) / 1\_000\_000 + " ms");

return result;

}

}

**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:aop="http://www.springframework.org/schema/aop"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

https://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/aop

https://www.springframework.org/schema/aop/spring-aop.xsd">

<aop:aspectj-autoproxy />

<bean id="bookRepository" class="com.library.repository.BookRepositoryImpl" />

<bean id="libraryService" class="com.library.service.LibraryServiceImpl">

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

</bean>

<bean class="com.library.aspect.LoggingAspect" />

</beans>

**Pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<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.example</groupId>

<artifactId>EX4\_Maven</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<maven.compiler.source>22</maven.compiler.source>

<maven.compiler.target>22</maven.compiler.target>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.34</version>

</dependency>

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.9.22</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

<version>3.11.0</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

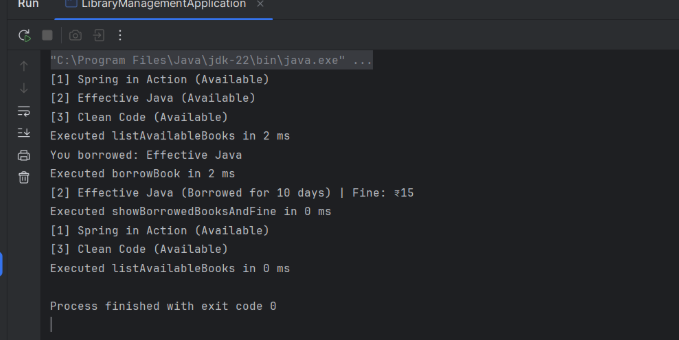
</plugin>

</plugins>

</build>

</project>

**OUTPUT :**

****