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

package com.library.repository;

public class BookRepository {

public void saveBook() {

System.out.println("Book saved to repository.");

}

}

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook() {

System.out.println("Service: Adding book");

bookRepository.saveBook();

}

}

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

public static void main(String[] args) {

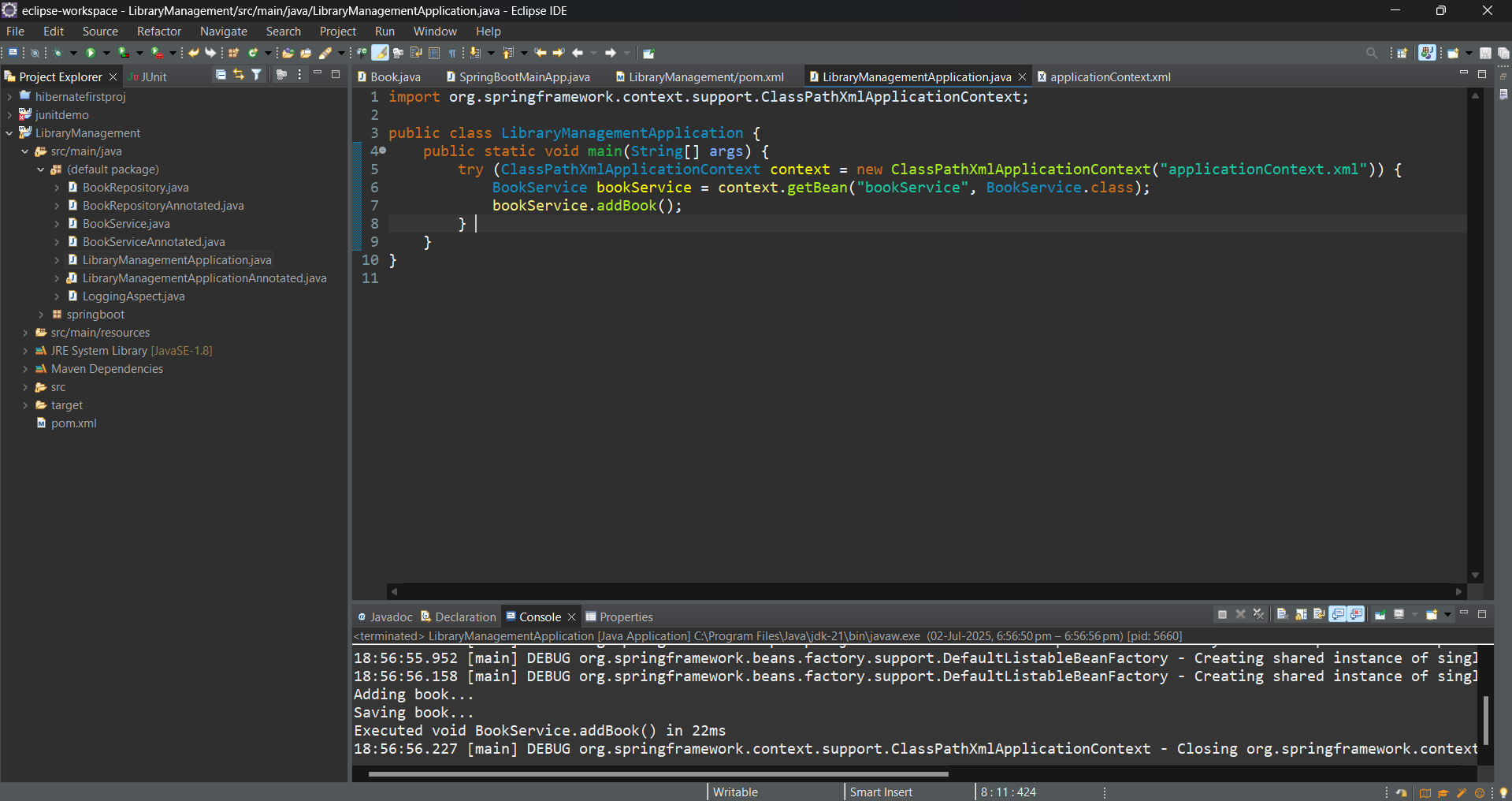
ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

BookService service = context.getBean(BookService.class);

service.addBook();

}

}



**Exercise 2: Implementing Dependency Injection**

**BookService.java**

public class BookService {

private BookRepository bookRepository;

// Constructor Injection

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook() {

System.out.println("Service: Adding book...");

bookRepository.saveBook();

}

}

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

**LoggingAspect.java**

package com.library.aspect;

import org.aspectj.lang.ProceedingJoinPoint;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

@Aspect

public class LoggingAspect {

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

public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

long start = System.currentTimeMillis();

Object result = joinPoint.proceed();

long timeTaken = System.currentTimeMillis() - start;

System.out.println("Execution time of " + joinPoint.getSignature() + ": " + timeTaken + "ms");

return result;

}

}

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

<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>LibraryManagement</groupId>

<artifactId>LibraryManagement</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<!-- Spring Core and Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

<!-- Spring AOP -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.32</version>

</dependency>

<!-- AspectJ Weaver -->

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.9.21</version>

</dependency>

<!-- Optional: Spring WebMVC if needed in future -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.32</version>

</dependency>

</dependencies>

<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>

</project>

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

<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="BookRepository"/>

<bean id="bookService" class="BookService">

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

</bean>

</beans>

**Exercise 6: Configuring Beans with Annotations**

<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">

<!-- Enable annotation-based configuration -->

<context:annotation-config/>

<context:component-scan base-package="."/> <!-- Scan default package -->

</beans>

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

<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">

<!-- Constructor injection for BookService -->

<bean id="bookRepository" class="BookRepository"/>

<bean id="bookService" class="BookService">

<constructor-arg ref="bookRepository"/>

</bean>

</beans>

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

<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="BookRepository"/>

<bean id="bookService" class="BookService">

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

</bean>

<bean id="loggingAspect" class="LoggingAspect"/>

<!-- Enable AOP -->

<aop:config>

<aop:aspect ref="loggingAspect">

<aop:pointcut id="allServiceMethods" expression="execution(\* BookService.\*(..))"/>

<aop:before method="logBefore" pointcut-ref="allServiceMethods"/>

<aop:after method="logAfter" pointcut-ref="allServiceMethods"/>

</aop:aspect>

</aop:config>

</beans>

**LoggingAspect.java**

public class LoggingAspect {

public void logBefore() {

System.out.println(“ Before method execution...");

}

public void logAfter() {

System.out.println(" After method execution.");

}

}

**BookService.java**

public class BookService {

private BookRepository bookRepository;

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook() {

System.out.println(" BookService: Adding a book...");

bookRepository.saveBook();

}

}

**BookRepository.java**

public class BookRepository {

public void saveBook() {

System.out.println(" BookRepository: Book saved to DB.");

}

}

**LibraryManagementApplication.java**

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.class);

bookService.addBook();

}

}

**Exercise 9: Creating a Spring Boot Application**

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>LibraryManagement</groupId>

<artifactId>LibraryManagement</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<!-- Spring Core: ApplicationContext, Beans, etc. -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

<!-- Spring AOP: Aspect-Oriented Programming -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.32</version>

</dependency>

<!-- AspectJ Weaver: Required for Spring AOP runtime -->

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.9.21</version>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Compiler plugin for Java 8 -->

<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>

</project>

**ApplicationContext.xml**

<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">

<!-- BookRepository Bean -->

<bean id="bookRepository" class="BookRepository"/>

<!-- BookService Bean with Setter Injection -->

<bean id="bookService" class="BookService">

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

</bean>

<!-- Logging Aspect Bean -->

<bean class="LoggingAspect"/>

<!-- Enable AOP with AspectJ -->

<aop:aspectj-autoproxy/>

</beans>

**application.properties**

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

**Book.java**

package com.library.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Book {

@Id

private int id;

private String title;

// Getters & Setters

}

**BookRepository.java**

package com.library.repository;

import com.library.model.Book;

import org.springframework.data.jpa.repository.JpaRepository;

public interface BookRepository extends JpaRepository<Book, Integer> {}

**BookController.java**

package com.library.controller;

import com.library.model.Book;

import com.library.repository.BookRepository;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/books")

public class BookController {

private final BookRepository repo;

public BookController(BookRepository repo) {

this.repo = repo;

}

@PostMapping

public Book addBook(@RequestBody Book book) {

return repo.save(book);

}

@GetMapping

public List<Book> getBooks() {

return repo.findAll();

}

}

**LibraryManagementApplication.java**

package com.library;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class LibraryManagementApplication {

public static void main(String[] args) {

SpringApplication.run(LibraryManagementApplication.class, args);

}

}

