EXERCISE 1:

BookRepository.java

package com.library.repository;

public class BookRepository {

    public void saveBook() {

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

    }

}

BookService.java

package com.library.service;

public class BookService {

    public void addBook() {

        System.out.println("Book added using BookService.");

    }

}

LibrarymanagementApplication.java

package com.library.librarymanagement;

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 bookService = context.getBean(BookService.class);

        bookService.addBook();

    }

}

Pom.xml

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

</dependencies>

applicationContext.xml

<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

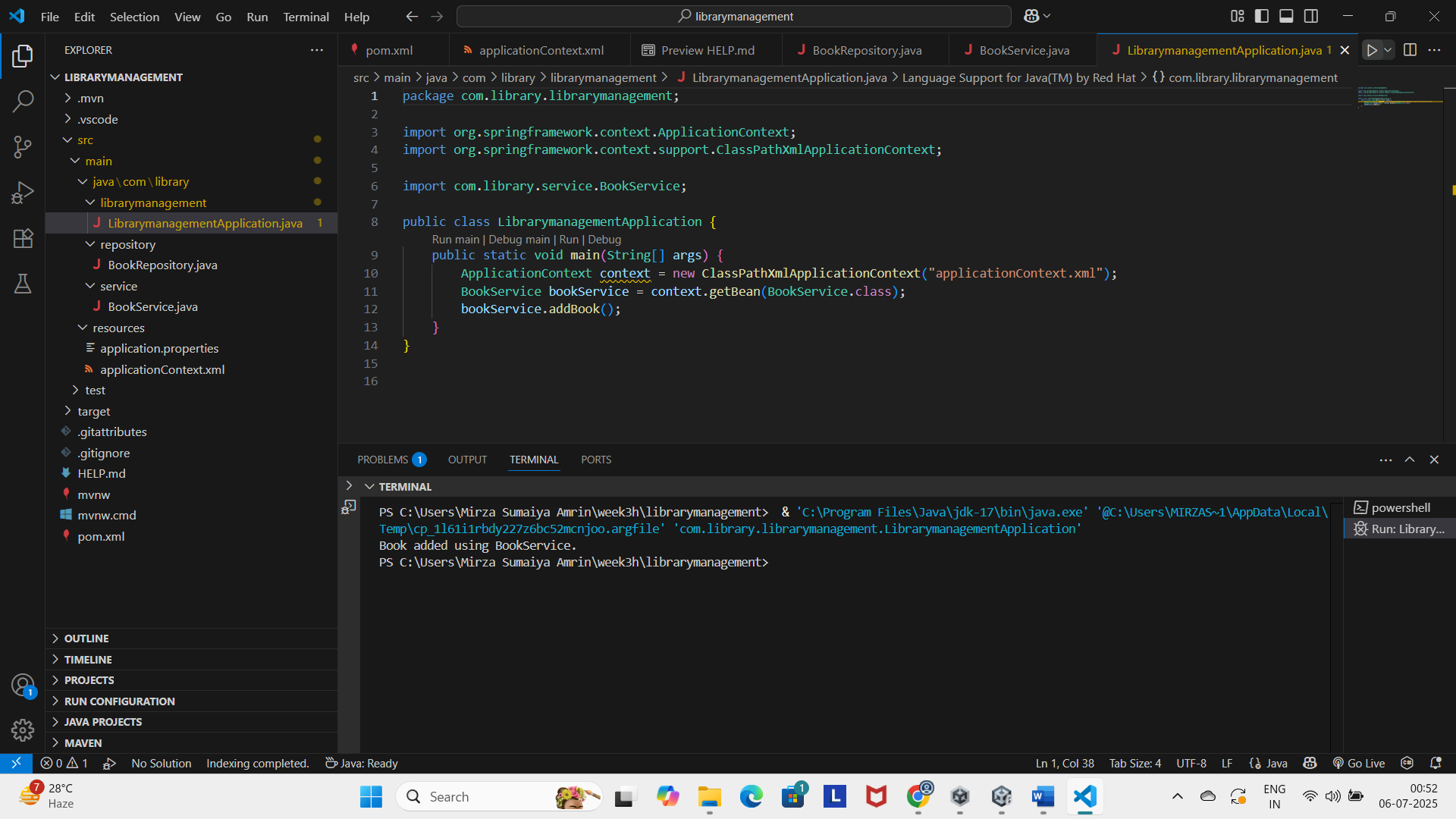
        http://www.springframework.org/schema/beans/spring-beans.xsd">

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

    <bean id="bookService" class="com.library.service.BookService"/>

</beans>

Output:



Exercise 2:

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    // Setter method for Dependency Injection

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    public void addBook() {

        System.out.println("Book added using BookService.");

        bookRepository.saveBook();  // Call method from BookRepository

    }

}

BookRepository.java

package com.library.repository;

public class BookRepository {

    public void saveBook() {

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

    }

}

LibrarymanagementApplication.java

package com.library.librarymanagement;

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 bookService = (BookService) context.getBean("bookService");

        // Call method that uses injected BookRepository

        bookService.addBook();

    }

}

applicationContext.xml

<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">

        <!-- Setter Injection -->

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

    </bean>

</beans>

Output:

A screenshot of a computer screen

AI-generated content may be incorrect.

Exercise 3:

BookService.java

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("Book added using BookService.");

        bookRepository.saveBook();

    }

}

BookRepository.java

package com.library.repository;

public class BookRepository {

    public void saveBook() {

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

    }

}

LoggingAspect.java

package com.library.aspect;

import org.aspectj.lang.ProceedingJoinPoint;

public class LoggingAspect {

    public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

        long start = System.currentTimeMillis();

        // Proceed with the actual method call

        Object result = joinPoint.proceed();

        long timeTaken = System.currentTimeMillis() - start;

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

        return result;

    }

}

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">

    <!-- Beans -->

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

    <bean id="bookService" class="com.library.service.BookService">

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

    </bean>

    <!-- Aspect Bean -->

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

    <!-- Enable AOP -->

    <aop:config>

        <aop:aspect ref="loggingAspect">

            <!-- Around advice for all methods in BookService -->

            <aop:around method="logExecutionTime"

                        pointcut="execution(\* com.library.service.\*.\*(..))"/>

        </aop:aspect>

    </aop:config>

</beans>

LibrarymanagementApplication.java

package com.library.librarymanagement;

import com.library.service.BookService;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.EnableAspectJAutoProxy;

import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication

@EnableAspectJAutoProxy

public class LibrarymanagementApplication {

    public static void main(String[] args) {

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

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

        bookService.addBook();

    }

}

Pom.xml

<dependency>

    <groupId>org.springframework</groupId>

    <artifactId>spring-aop</artifactId>

</dependency>

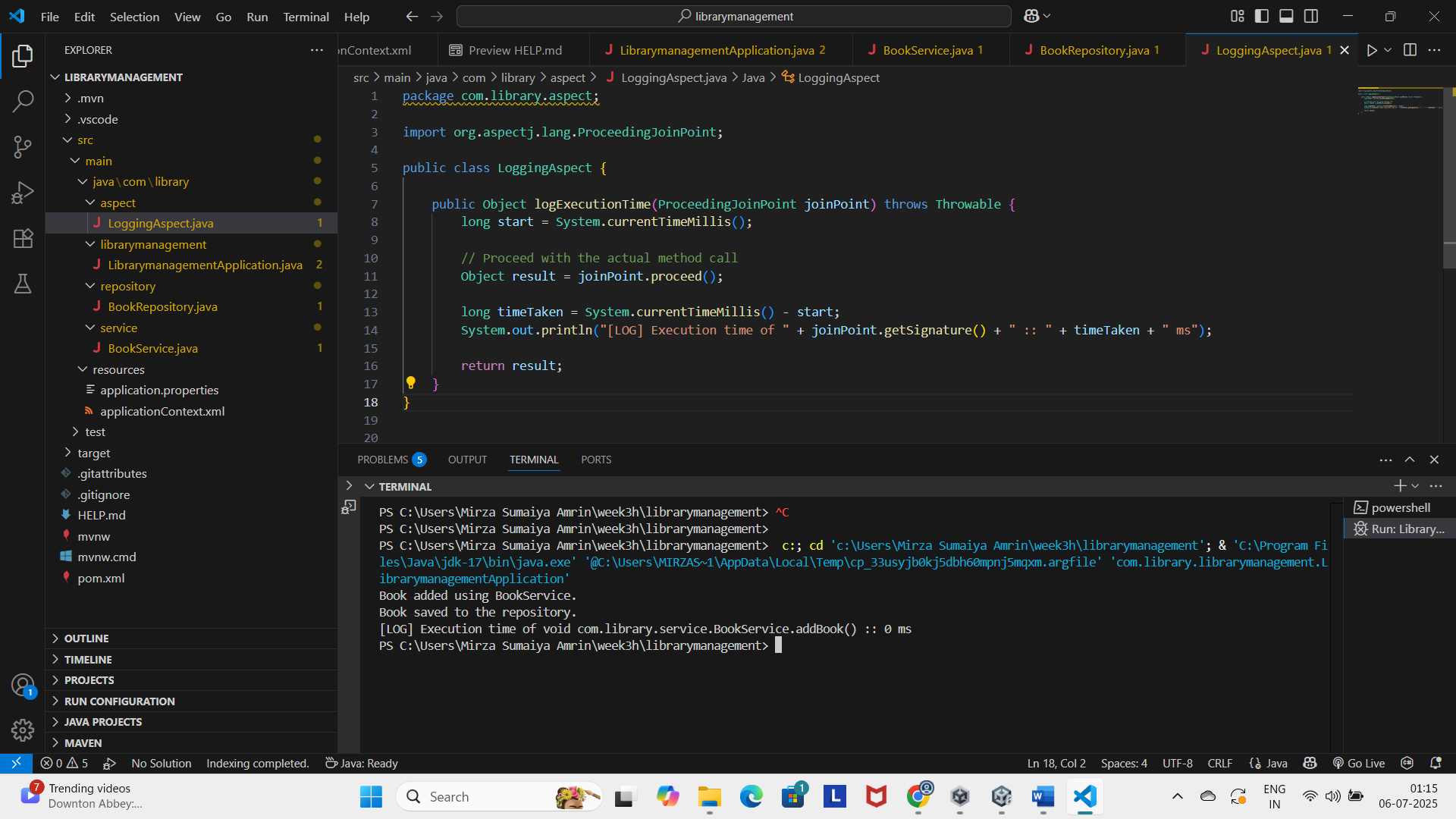
<dependency>

    <groupId>org.aspectj</groupId>

    <artifactId>aspectjweaver</artifactId>

</dependency>

Output:



Exercise 4:

Same project as Exercise 3, Update pom.xml to build tests

<dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-webmvc</artifactId>

</dependency>

<dependency>

            <groupId>junit</groupId>

            <artifactId>junit</artifactId>

            <scope>test</scope>

</dependency>

<dependency>

            <groupId>javax.servlet</groupId>

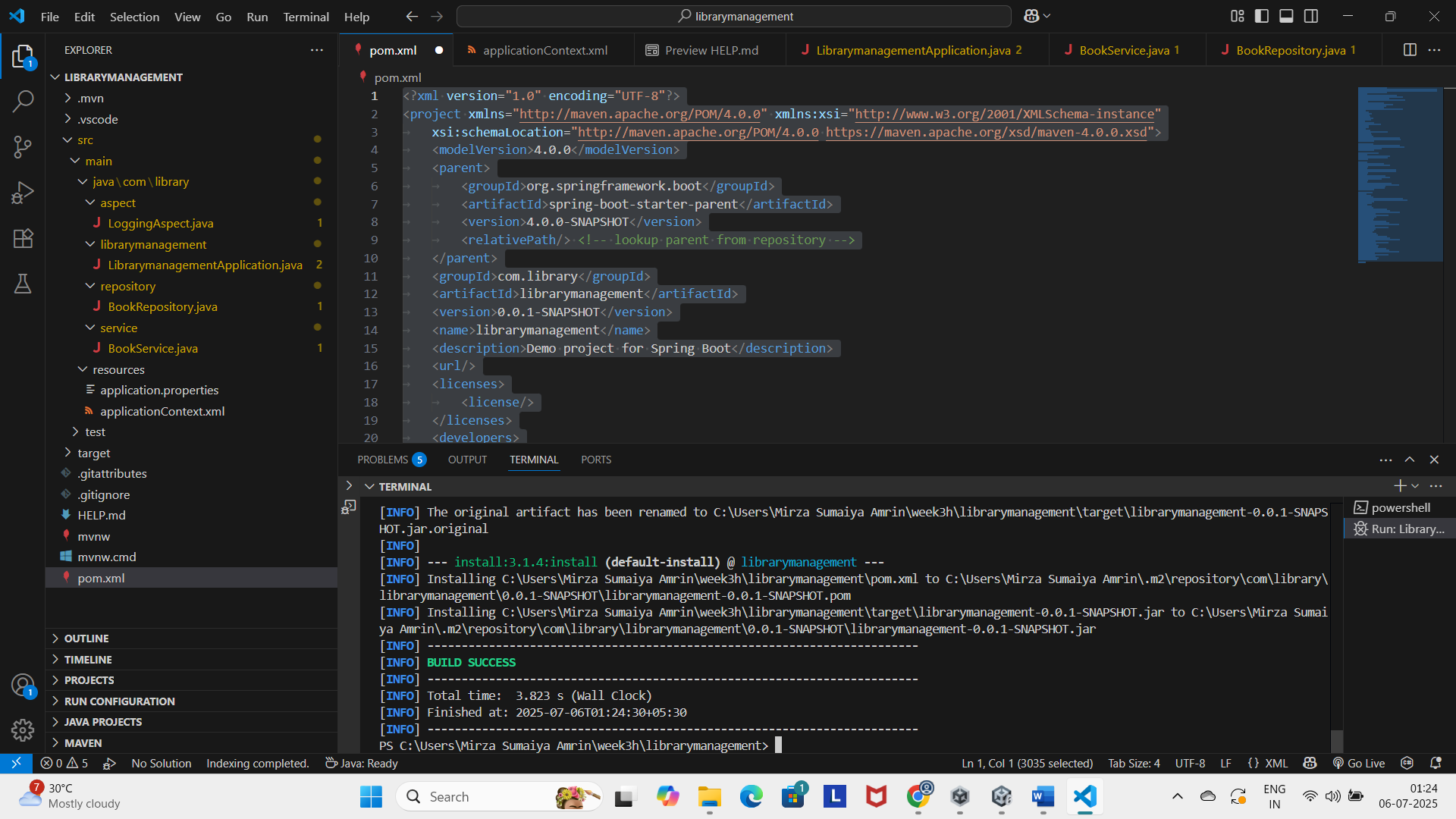
            <artifactId>javax.servlet-api</artifactId>

            <version>4.0.1</version>

            <scope>provided</scope>

        </dependency>

Output:



Exercise 5:

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    // Setter for setter injection

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    public void addBook(String title) {

        System.out.println("Adding book: " + title);

        bookRepository.saveBook(title);

    }

}

BookRepository.java

package com.library.repository;

public class BookRepository {

    public void saveBook(String title) {

        System.out.println("Saving book: " + title);

    }

}

LibrarymanagementApplication.java

package com.library.librarymanagement;

import com.library.service.BookService;

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);

        bookService.addBook("Java Fundamentals");

    }

}

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">

    <!-- BookRepository bean -->

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

    <!-- BookService bean with setter injection -->

    <bean id="bookService" class="com.library.service.BookService">

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

    </bean>

</beans>

A screenshot of a computer screen

AI-generated content may be incorrect.Output:

Exercise 6:

BookService.java

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  // Setter or field injection

    private BookRepository bookRepository;

    public void addBook(String title) {

        System.out.println("Adding book: " + title);

        bookRepository.saveBook(title);

    }

}

BookRepository.java

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

    public void saveBook(String title) {

        System.out.println("Saving book: " + title);

    }

}

LibararymanagementApplication.java

package com.library.librarymanagement;

import com.library.service.BookService;

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("Spring with Annotations");

    }

}

applicationContext.xml

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

<beans xmlns="http://www.springframework.org/schema/beans"

       xmlns:context="http://www.springframework.org/schema/context"

       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

           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

    <!-- Enable annotation-based component scanning -->

    <context:component-scan base-package="com.library" />

</beans>

Output:

A screenshot of a computer screen

AI-generated content may be incorrect.

Exercise 7:

BookRepository.java

package com.library.repository;

public class BookRepository {

    public void saveBook(String title) {

        System.out.println("Saving book: " + title);

    }

}

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    private String serviceName;

    // Constructor injection

    public BookService(String serviceName) {

        this.serviceName = serviceName;

    }

    // Setter injection

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    public void addBook(String title) {

        System.out.println("[" + serviceName + "] Adding book: " + title);

        bookRepository.saveBook(title);

    }

}

LibrarymanagementApplication.java

package com.library.librarymanagement;

import com.library.service.BookService;

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("Effective Java");

    }

}

applicationContext.xml

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

<beans xmlns="http://www.springframework.org/schema/beans"

       xmlns:context="http://www.springframework.org/schema/context"

       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

           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

    <!-- Enable annotation scanning -->

    <context:component-scan base-package="com.library" />

    <!-- BookRepository bean -->

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

    <!-- BookService bean using constructor and setter injection -->

    <bean id="bookService" class="com.library.service.BookService">

        <constructor-arg value="LibraryService" />

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

    </bean>

</beans>

Output:

A screenshot of a computer screen

AI-generated content may be incorrect.

Exercise 8:

BookRepository.java

package com.library.repository;

public class BookRepository {

    public void saveBook(String title) {

        System.out.println("Saving book: " + title);

    }

}

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    private String serviceName;

    // Constructor injection

    public BookService(String serviceName) {

        this.serviceName = serviceName;

    }

    // Setter injection

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    public void addBook(String title) {

        System.out.println("[" + serviceName + "] Adding book: " + title);

        bookRepository.saveBook(title);

    }

}

LibrarymanagemetApplication.java

package com.library.librarymanagement;

import com.library.service.BookService;

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("Effective Java");

    }

}

applicationContext.xml

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

<beans xmlns="http://www.springframework.org/schema/beans"

       xmlns:context="http://www.springframework.org/schema/context"

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

       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

         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">

    <!-- Enable annotation scanning -->

    <context:component-scan base-package="com.library" />

    <!-- AOP auto-proxy support -->

    <aop:aspectj-autoproxy />

    <!-- Beans -->

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

    <bean id="bookService" class="com.library.service.BookService">

        <constructor-arg value="LibraryService" />

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

    </bean>

    <!-- Aspect bean -->

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

</beans>

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.BookService.\*(..))")

    public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

        long start = System.currentTimeMillis();

        Object result = joinPoint.proceed();  // Actual method execution

        long end = System.currentTimeMillis();

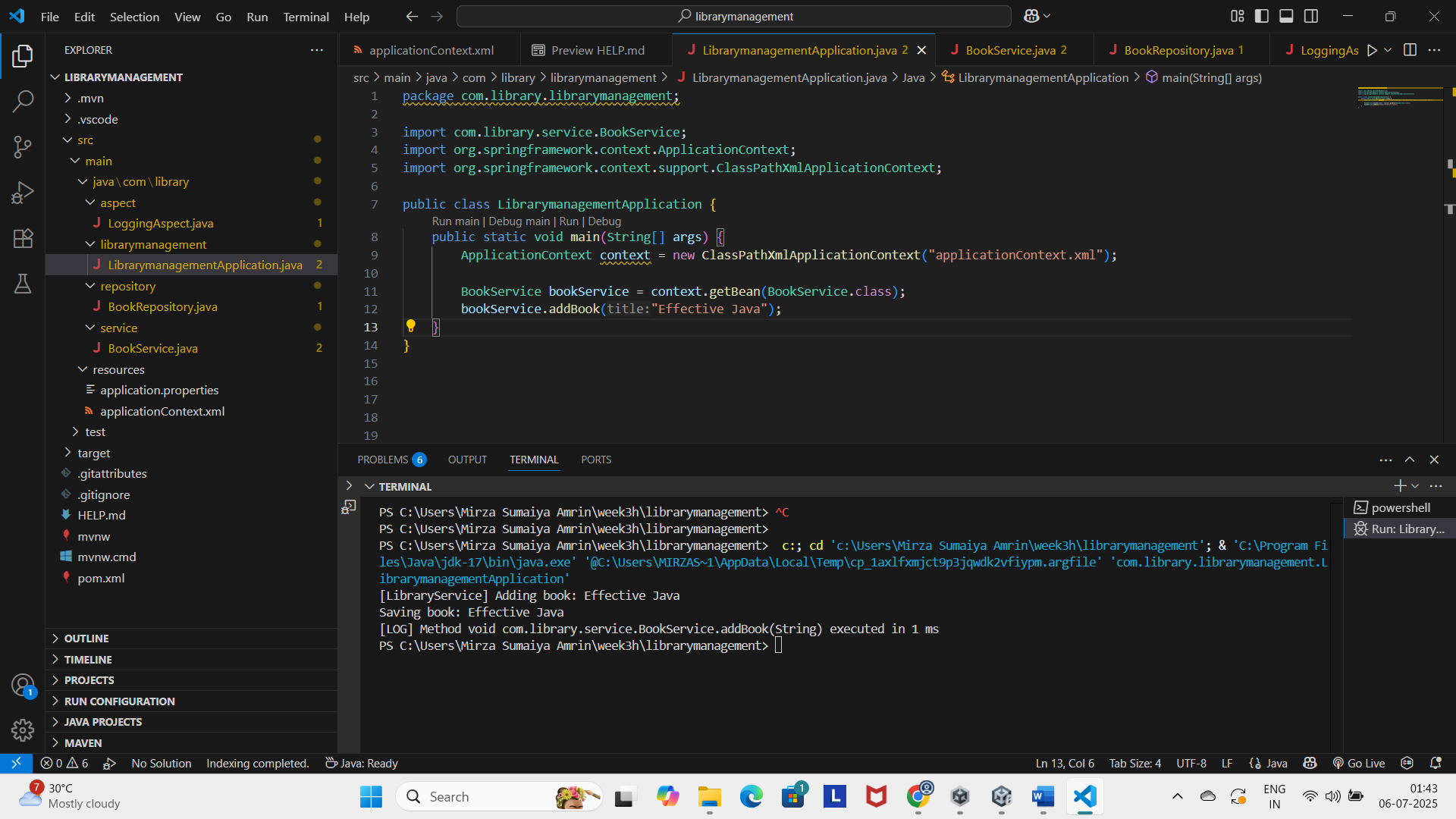
        System.out.println("[LOG] Method " + joinPoint.getSignature() + " executed in " + (end - start) + " ms");

        return result;

    }

}

Output:



Exercise 9:

BookRepository.java

package com.library.librarymanagement.repository;

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

import org.springframework.stereotype.Repository;

import com.library.librarymanagement.model.Book;

@Repository

public interface BookRepository extends JpaRepository<Book, Long> {

}

BookService.java

package com.library.librarymanagement.service;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.library.librarymanagement.model.Book;

import com.library.librarymanagement.repository.BookRepository;

@Service

public class BookService {

    private BookRepository bookRepository;

    private String serviceName;

    // No-arg constructor: setting a default service name.

    public BookService() {

        this.serviceName = "LibraryService";

    }

    // Setter injection for BookRepository

    @Autowired

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    /\*\*

     \* Adds a new Book with the given title.

     \* For simplicity, this method assigns a default author ("Unknown").

     \* You could overload this method to accept an author as an additional parameter.

     \*

     \* @param title the title of the book

     \* @return the saved Book entity

     \*/

    public Book addBook(String title) {

        System.out.println("[" + serviceName + "] Adding book: " + title);

        // Create a new Book entity and set its properties.

        Book book = new Book();

        book.setTitle(title);

        book.setAuthor("Unknown"); // Default author; adjust as needed.

        // Save the Book using the JPA repository's save() method.

        return bookRepository.save(book);

    }

}

BookController.java

package com.library.librarymanagement.controller;

import com.library.librarymanagement.model.Book;

import com.library.librarymanagement.repository.BookRepository;

import org.springframework.beans.factory.annotation.Autowired;

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

import java.util.List;

@RestController

@RequestMapping("/books")

public class BookController {

    @Autowired

    private BookRepository bookRepository;

    // POST /books

    @PostMapping

    public Book saveBook(@RequestBody Book book) {

        return bookRepository.save(book);

    }

    // GET /books

    @GetMapping

    public List<Book> getAllBooks() {

        return bookRepository.findAll();

    }

}

LibrarymanagementApplication.java

package com.library.librarymanagement;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan(basePackages = {"com.library"})

public class LibrarymanagementApplication {

    public static void main(String[] args) {

        SpringApplication.run(LibrarymanagementApplication.class, args);

    }

}

applicationContext.xml

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

<beans xmlns="http://www.springframework.org/schema/beans"

       xmlns:context="http://www.springframework.org/schema/context"

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

       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

         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">

    <!-- Enable annotation scanning -->

    <context:component-scan base-package="com.library" />

    <!-- AOP auto-proxy support -->

    <aop:aspectj-autoproxy />

    <!-- Beans -->

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

    <bean id="bookService" class="com.library.service.BookService">

        <constructor-arg value="LibraryService" />

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

    </bean>

    <!-- Aspect bean -->

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

</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.hibernate.ddl-auto=update

spring.jpa.show-sql=true

# Optional: Enable H2 console

spring.h2.console.enabled=true

LoggingAspect.java

package com.library.librarymanagement.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.BookService.\*(..))")

    public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

        long start = System.currentTimeMillis();

        Object result = joinPoint.proceed();  // Actual method execution

        long end = System.currentTimeMillis();

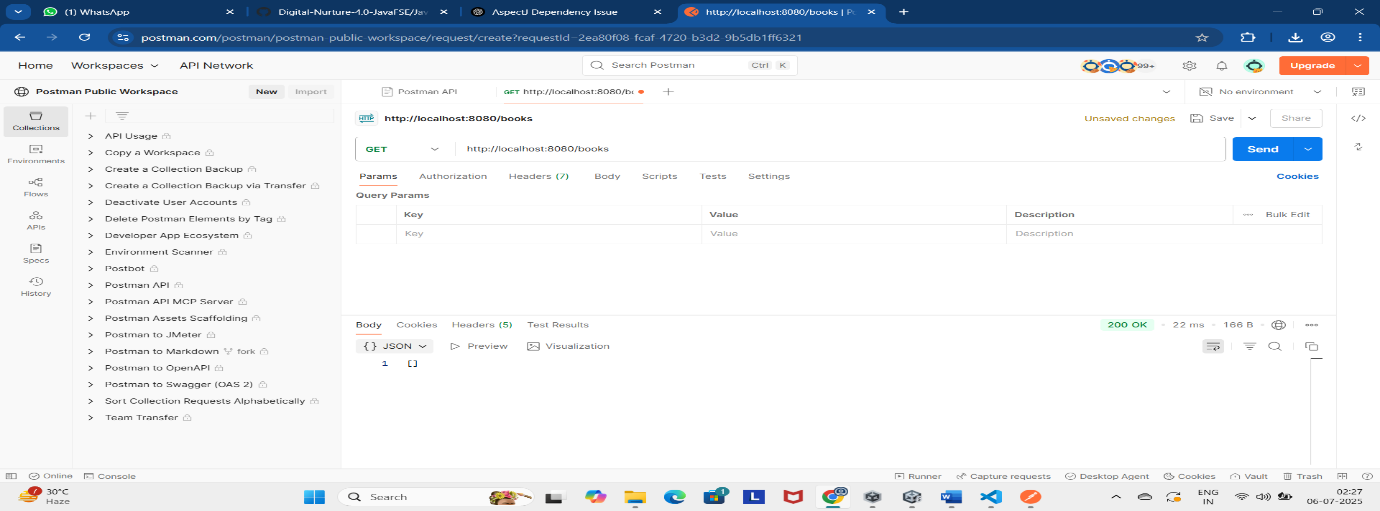
        System.out.println("[LOG] Method " + joinPoint.getSignature() + " executed in " + (end - start) + " ms");

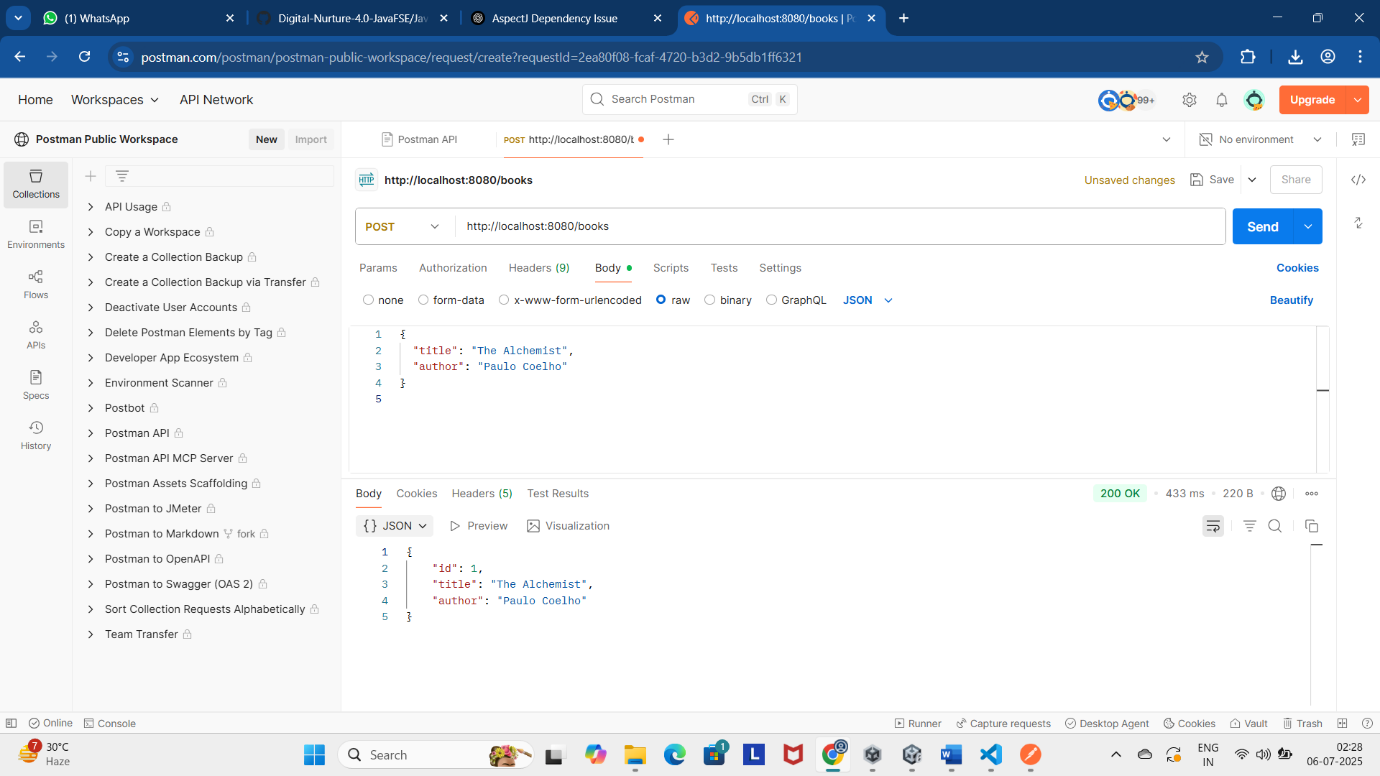
        return result;

    }

}

Output:





A screenshot of a computer

AI-generated content may be incorrect.