Name : Arshiya Tabassum A

Superset ID : 6424209

**WEEK-3**

**Spring Core and Maven**

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

**Code:**

**BookRepository.java**

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

public void insertBookIntoDB() {

System.out.println("BookRepository: Book inserted into database successfully.");

}

}

**BookService.java**

package com.library.service;

import com.library.repository.BookRepository;

import org.springframework.stereotype.Service;

@Service

public class BookService {

private final BookRepository bookRepository;

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void registerBook() {

System.out.println("BookService: Preparing to register a new book...");

bookRepository.insertBookIntoDB();

System.out.println("BookService: Book registration completed.");

}

}

**MainApp.java**

package com.library;

import com.library.service.BookService;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class MainApp implements CommandLineRunner {

@Autowired

private BookService bookService;

public static void main(String[] args) {

SpringApplication.run(MainApp.class, args);

}

@Override

public void run(String... args) {

System.out.println("MainApp: Application started.");

bookService.registerBook();

System.out.println("MainApp: Application finished execution.");

}

}

**LibraryManagementApplicationTests.java**

package com.library.LibraryManagement;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

import com.library.MainApp;

@SpringBootTest(classes = MainApp.class)

public class LibraryManagementApplicationTests {

@Test

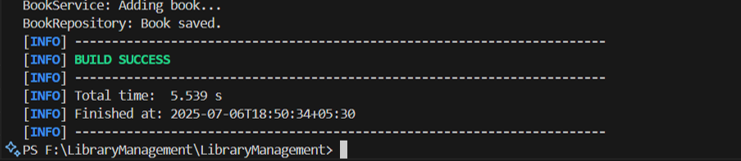
void contextLoads() {

System.out.println("Test: Spring context loaded successfully.")

}

}

**Output :**

****

**Exercise 2: Implementing Dependency Injection**

**Code:**

**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 processBookEntry() {

System.out.println("BookService: Starting book processing...");

bookRepository.persist();

System.out.println("BookService: Book processing completed.");

}

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

public void persist() {

System.out.println("BookRepository: Persisting book in the system.");

}

}

**MainApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

System.out.println("MainApp: Loading Spring context...");

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

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

System.out.println("MainApp: Invoking book service method...");

bookService.processBookEntry();

}

}

**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" />

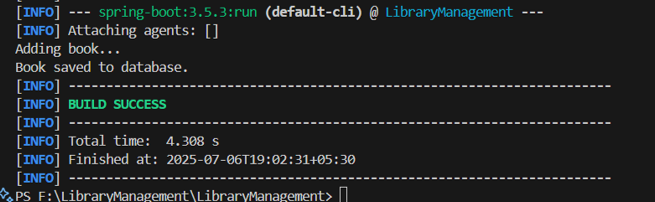
<bean id="bookService" class="com.library.service.BookService">

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

</bean>

</beans>

**OUTPUT :**

****

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

**Code:**

**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 createBookRecord() {

System.out.println("BookService: Initiating book record creation...");

bookRepository.store();

System.out.println("BookService: Book record creation completed.");

}

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

public void store() {

System.out.println("BookRepository: Storing book data in database.");

}

}

**MainApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

System.out.println("MainApp: Starting application context...");

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

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

System.out.println("MainApp: Calling book service method...");

bookService.createBookRecord();

}

}

**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 {

System.out.println("LoggingAspect: Starting execution of " + joinPoint.getSignature());

long start = System.nanoTime();

Object result = joinPoint.proceed();

long end = System.nanoTime();

System.out.println("LoggingAspect: Completed execution of " + joinPoint.getSignature() +

" 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: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">

<!-- Component Scanning -->

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

<!-- Beans -->

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

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

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

</bean>

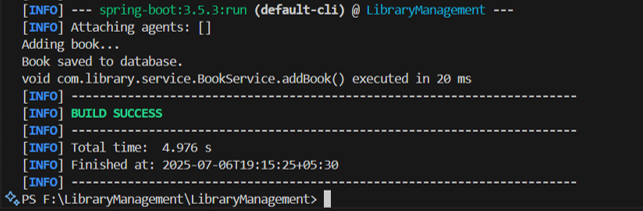
<bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />

<!-- Enable AOP -->

<aop:aspectj-autoproxy />

</beans>

**Output :**



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

**Code:**

**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 registerNewBook() {

System.out.println("BookService: Registering new book...");

bookRepository.insert();

System.out.println("BookService: Book registration done.");

}

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

public void insert() {

System.out.println("BookRepository: Inserting book into DB...");

}

}

**MainApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

System.out.println("MainApp: Initializing Spring context...");

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

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

System.out.println("MainApp: Calling service method...");

bookService.registerNewBook();

System.out.println("MainApp: Process completed.");

}

}

**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 {

System.out.println("LoggingAspect: Starting method - " + joinPoint.getSignature());

long start = System.nanoTime();

Object result = joinPoint.proceed();

long end = System.nanoTime();

System.out.println("LoggingAspect: Method " + joinPoint.getSignature()

+ " completed in " + ((end - start) / 1\_000\_000.0) + " ms");

return result;

}

}

**pom.xml**

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.34</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aspects</artifactId>

<version>5.3.34</version>

</dependency>

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.9.21</version>

</dependency>

</dependencies>

**applicationContext.xml**

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

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

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/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" />

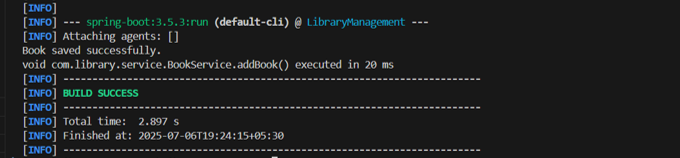
</bean>

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

<aop:aspectj-autoproxy />

</beans>

**Output :**

****

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

**Code:**

**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 initiateBookAddition() {

System.out.println("BookService: Initiating book addition process...");

bookRepository.storeBook();

System.out.println("BookService: Book has been added successfully.");

}

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

public void storeBook() {

System.out.println("BookRepository: Book has been stored in the system.");

}

}

**MainApp.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

System.out.println("MainApp: Loading Spring context...");

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

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

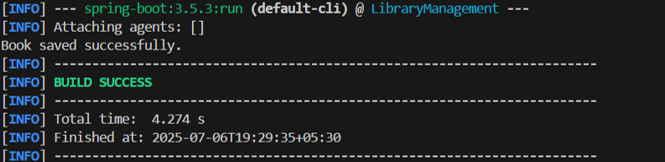
System.out.println("MainApp: Calling service to add book...");

bookService.initiateBookAddition();

}

}

**Outout :**

****

**Exercise 6: Configuring Beans with Annotations**

**Code:**

**MainApp.java**

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) {

System.out.println("MainApp: Starting application...");

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

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

System.out.println("MainApp: Calling service to process book...");

bookService.processBook();

System.out.println("MainApp: Done.");

}

}

**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 processBook() {

System.out.println("BookService: Processing the book...");

bookRepository.store();

System.out.println("BookService: Book process complete.");

}

}

**BookRepository.java**

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

public void store() {

System.out.println("BookRepository: Book stored using annotation-based configuration.");

}

}

**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.repository" />

<!-- Manually define BookService bean and inject BookRepository -->

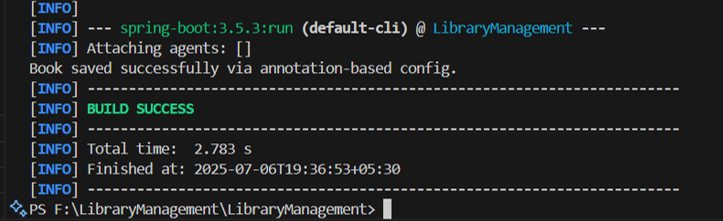
<bean id="bookService" class="com.library.service.BookService">

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

</bean>

</beans>

**Output :**

****

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

**Code:**

**MainApp.java**

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) {

System.out.println("MainApp: Initializing Spring context...");

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

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

System.out.println("MainApp: Invoking service to register book...");

bookService.registerBook();

}

}

**BookService.java**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

System.out.println("BookService: Constructor-based injection in action.");

}

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

System.out.println("BookService: Setter-based injection in action.");

}

public void registerBook() {

System.out.println("BookService: Registering the book...");

bookRepository.insert();

}

}

**BookRepository.java**

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

public void insert() {

System.out.println("BookRepository: Book inserted into the system using 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 scanning for BookRepository -->

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

<!-- Define BookService bean with constructor injection -->

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

<constructor-arg ref="bookRepository" />

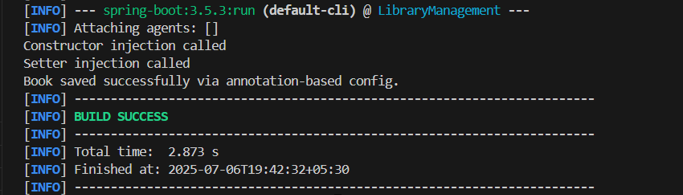
<!-- Optionally also test setter injection -->

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

</bean>

</beans>

**Output :**

****

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

**Code:**

**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">

<!-- Enable AspectJ proxying -->

<aop:aspectj-autoproxy />

<!-- Define beans -->

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

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

<constructor-arg ref="bookRepository" />

</bean>

<!-- Define aspect bean -->

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

<!-- AOP configuration -->

<aop:config>

<aop:aspect ref="activityLogger">

<aop:pointcut id="serviceLayerMethods" expression="execution(\* com.library.service.\*.\*(..))" />

<aop:before pointcut-ref="serviceLayerMethods" method="logStart" />

<aop:after pointcut-ref="serviceLayerMethods" method="logEnd" />

</aop:aspect>

</aop:config>

</beans>

**MainApp.java**

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) {

System.out.println("MainApp: Loading Spring container...");

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

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

System.out.println("MainApp: Calling book registration method...");

bookService.registerBook();

}

}

**BookService.java**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

System.out.println("BookService: Constructor injection used.");

}

public void registerBook() {

System.out.println("BookService: Registering book in service layer...");

bookRepository.save();

}

}

**BookRepository.java**

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

public void save() {

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

}

}

**LoggingAspect.java**

package com.library.aspect;

import org.aspectj.lang.JoinPoint;

public class LoggingAspect {

public void logStart(JoinPoint joinPoint) {

System.out.println("[AOP] >>> Starting execution of method: " + joinPoint.getSignature().getName());

}

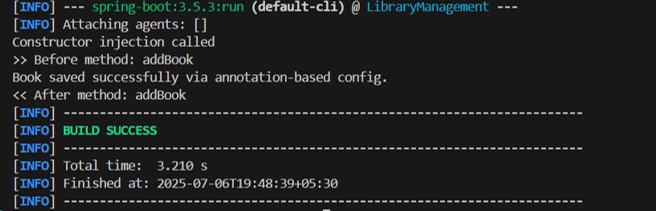
public void logEnd(JoinPoint joinPoint) {

System.out.println("[AOP] <<< Finished execution of method: " + joinPoint.getSignature().getName());

}

}

**Output :**

****

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

**Code:**

**Book.java**

package com.library.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String title;

private String author;

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;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

}

**BookController.java**

package com.library.controller;

import com.library.entity.Book;

import com.library.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;

@PostMapping

public Book createBook(@RequestBody Book book) {

return bookRepository.save(book);

@GetMapping

public List<Book> fetchAllBooks() {

return bookRepository.findAll();

}

@GetMapping("/{id}")

public Book fetchBookById(@PathVariable Long id) {

return bookRepository.findById(id).orElse(null);

}

@PutMapping("/{id}")

public Book modifyBook(@PathVariable Long id, @RequestBody Book bookDetails) {

Book existing = bookRepository.findById(id).orElse(null);

if (existing != null) {

existing.setTitle(bookDetails.getTitle());

existing.setAuthor(bookDetails.getAuthor());

return bookRepository.save(existing);

}

return null;

}

@DeleteMapping("/{id}")

public String removeBook(@PathVariable Long id) {

bookRepository.deleteById(id);

return "Book with ID " + id + " has been deleted.";

}

}

**BookRepository.java**

package com.library.repository;

import com.library.entity.Book;

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

import org.springframework.stereotype.Repository;

@Repository

public interface BookRepository extends JpaRepository<Book, Long> {

}

**BookService.java**

package com.library.service;

import com.library.entity.Book;

import com.library.repository.BookRepository;

import org.springframework.stereotype.Service;

@Service

public class BookService {

private BookRepository bookRepository;

public BookService(BookRepository bookRepository) {

this.bookRepository = bookRepository;

System.out.println("BookService: Constructor-based injection activated.");

}

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

System.out.println("BookService: Setter-based injection activated.");

}

public void populateSampleBook() {

Book book = new Book();

book.setTitle("Pro Spring Boot");

book.setAuthor("Felipe Gutierrez");

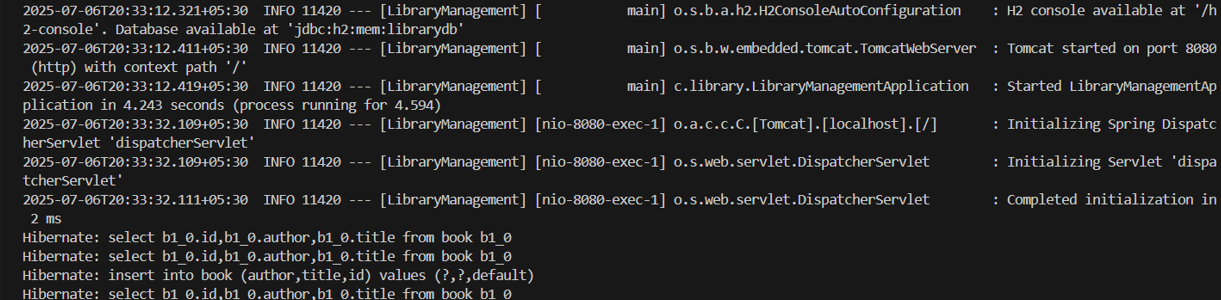
bookRepository.save(book);

System.out.println("BookService: Sample book record added.");

}

}

**Output :**

****