**HANDSON EXERCISES - WEEK 3**

**Skill : Spring Core and Maven**

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

**CODE :**

**src/main/resources : applicationContent.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>

**com.library.repository : BookRepository.java -**

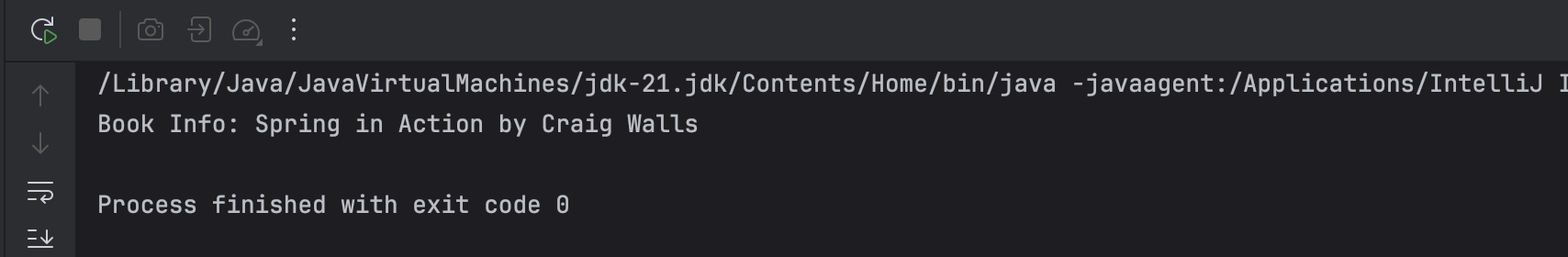
package com.library.repository;  
  
public class BookRepository {  
 public String getBookInfo() {  
 return "Spring in Action by Craig Walls";  
 }  
}

**com.library.service : 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 displayBook() {  
 System.out.println("Book Info: " + bookRepository.getBookInfo());  
 }  
}

**src/main/java : Main.java -**

import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
 bookService.displayBook();  
 }  
}

**OUTPUT :**  


**Exercise 2 : Implementing Dependency Injection**

**CODE :**

**src/main/resources : applicationContent.xml -**

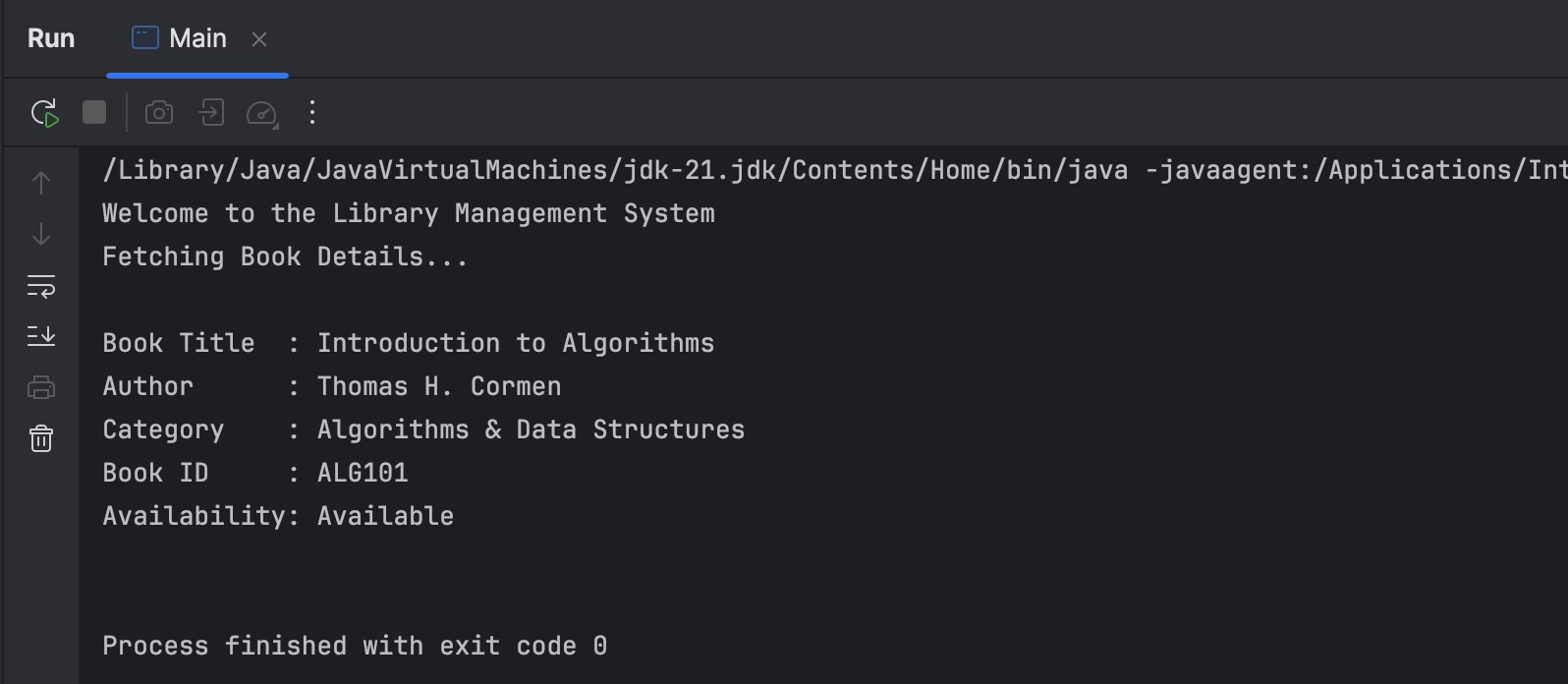
<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="  
 http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
 <bean id="bookRepository" class="com.library.repository.BookRepository"/>  
  
 <bean id="bookService" class="com.library.service.BookService">  
 <property name="bookRepository" ref="bookRepository"/>  
 </bean>  
  
</beans>

**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 displayBook() {  
 System.out.println("Welcome to the Library Management System");  
 System.out.println("Fetching Book Details...\n");  
 System.out.println(bookRepository.getBookInfo());  
 }  
}

**BookRepository.java :**

package com.library.repository;  
  
public class BookRepository {  
 public String getBookInfo() {  
 return """  
 Book Title : Introduction to Algorithms  
 Author : Thomas H. Cormen  
 Category : Algorithms & Data Structures  
 Book ID : ALG101  
 Availability: Available  
 """;  
 }  
}

**OUTPUT :**

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

**CODE:**

**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>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>21</maven.compiler.source>  
 <maven.compiler.target>21</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
  
 <dependencies>  
  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.38</version>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aop</artifactId>  
 <version>5.3.38</version>  
 </dependency>  
  
 <dependency>  
 <groupId>org.aspectj</groupId>  
 <artifactId>aspectjweaver</artifactId>  
 <version>1.9.21</version>  
 </dependency>  
  
 </dependencies>  
  
</project>

**com.library.aspect : 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(); // proceed with the original method  
  
 long end = System.currentTimeMillis();  
 System.out.println("Execution time of " + joinPoint.getSignature().getName() + ": " + (end - start) + " ms");  
  
 return result;  
 }  
}

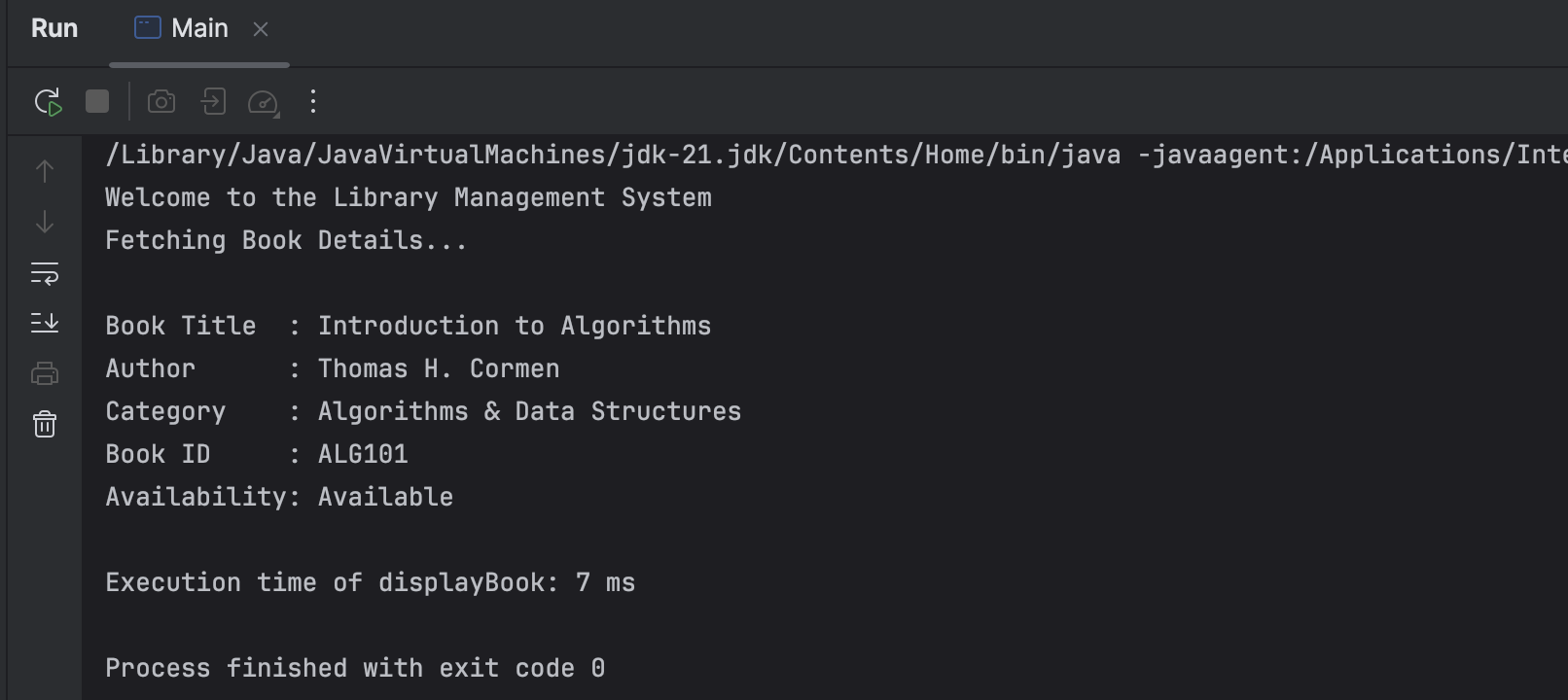
**applicationContext.xml :**

<?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  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/aop  
 http://www.springframework.org/schema/aop/spring-aop.xsd">  
  
 <aop:aspectj-autoproxy />  
  
 <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"/>  
</beans>

**Main.java :**

import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
 bookService.displayBook();

}

**OUTPUT :**

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

**CODE :**

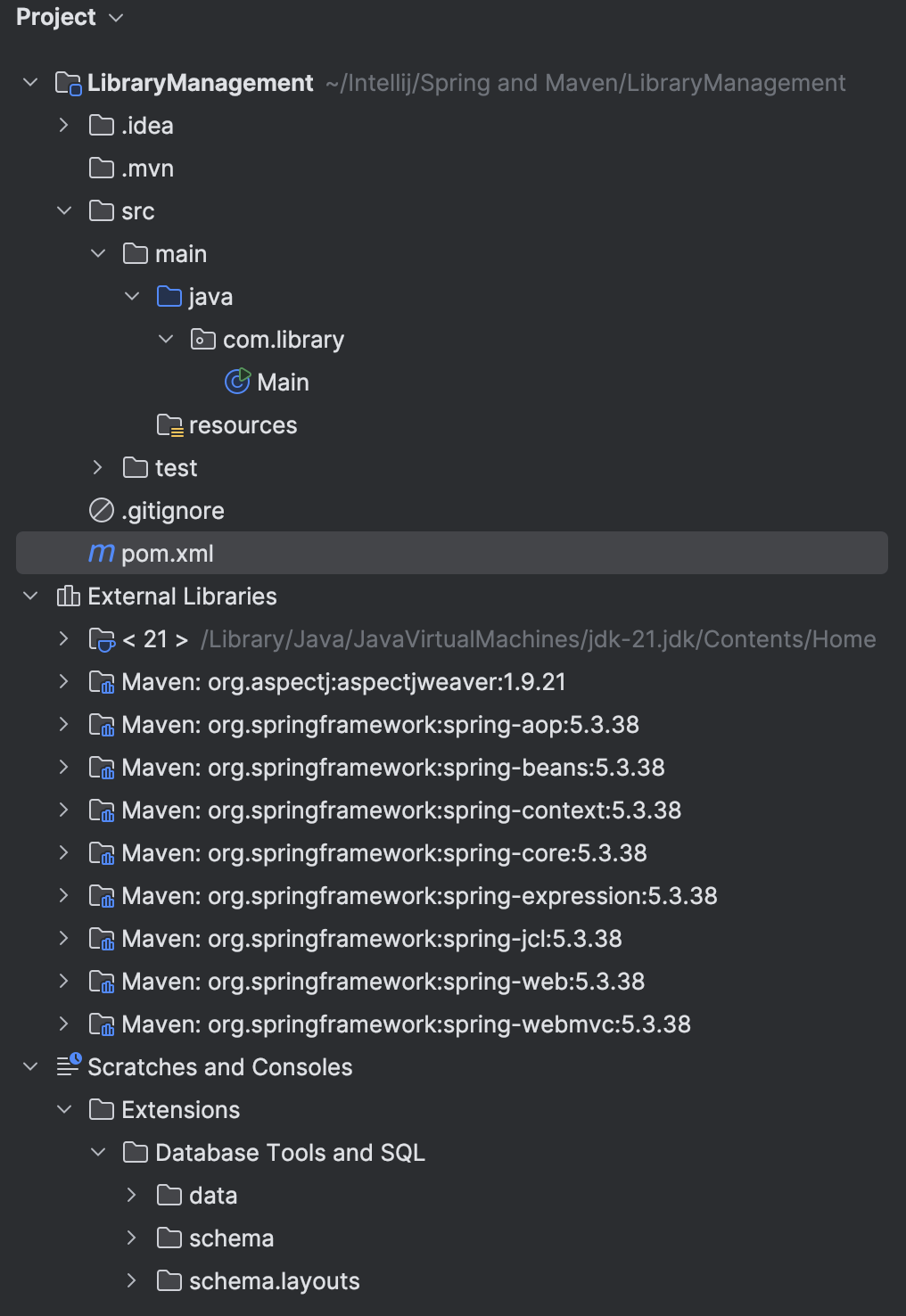
**pom.xml :**

<?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>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>1.8</maven.compiler.source>  
 <maven.compiler.target>1.8</maven.compiler.target>

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

<dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>5.3.38</version>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aop</artifactId>  
 <version>5.3.38</version>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>5.3.38</version>  
 </dependency>  
  
 <dependency>  
 <groupId>org.aspectj</groupId>  
 <artifactId>aspectjweaver</artifactId>  
 <version>1.9.21</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 :**



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

**CODE:**

**applicationContent.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>

**BookRepository.java :**

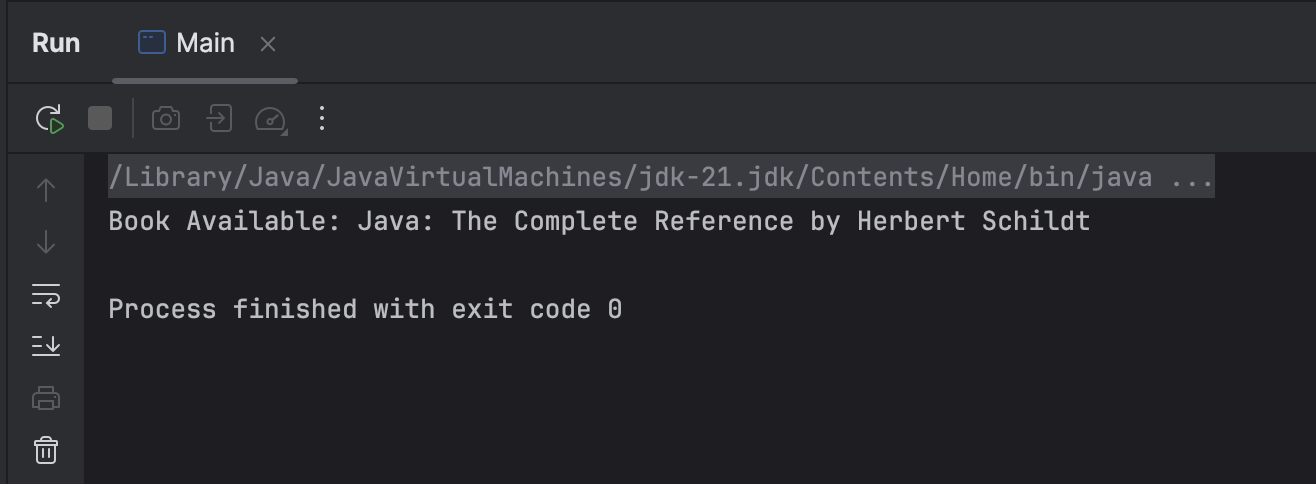
package com.library.repository;  
  
public class BookRepository {  
 public String getBookTitle() {  
 return "Java: The Complete Reference by Herbert Schildt";  
 }  
}

**BookService.java :**

package com.library.service;  
  
import com.library.repository.BookRepository;  
  
public class BookService {  
 private BookRepository bookRepository;  
  
 // setter for Spring to inject  
 public void setBookRepository(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void displayBook() {  
 System.out.println("Book Available: " + bookRepository.getBookTitle());  
 }  
}

**Main.java :**

import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
 bookService.displayBook();  
 }  
}

**OUTPUT :**

**Exercise 6 : Configuring Beans with Annotations**

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

**BookRepository.java :**

package com.library.repository;  
  
import org.springframework.stereotype.Repository;  
  
@Repository   
public class BookRepository {  
 public String getBookTitle() {  
 return "Java: The Complete Reference by Herbert Schildt";  
 }  
}

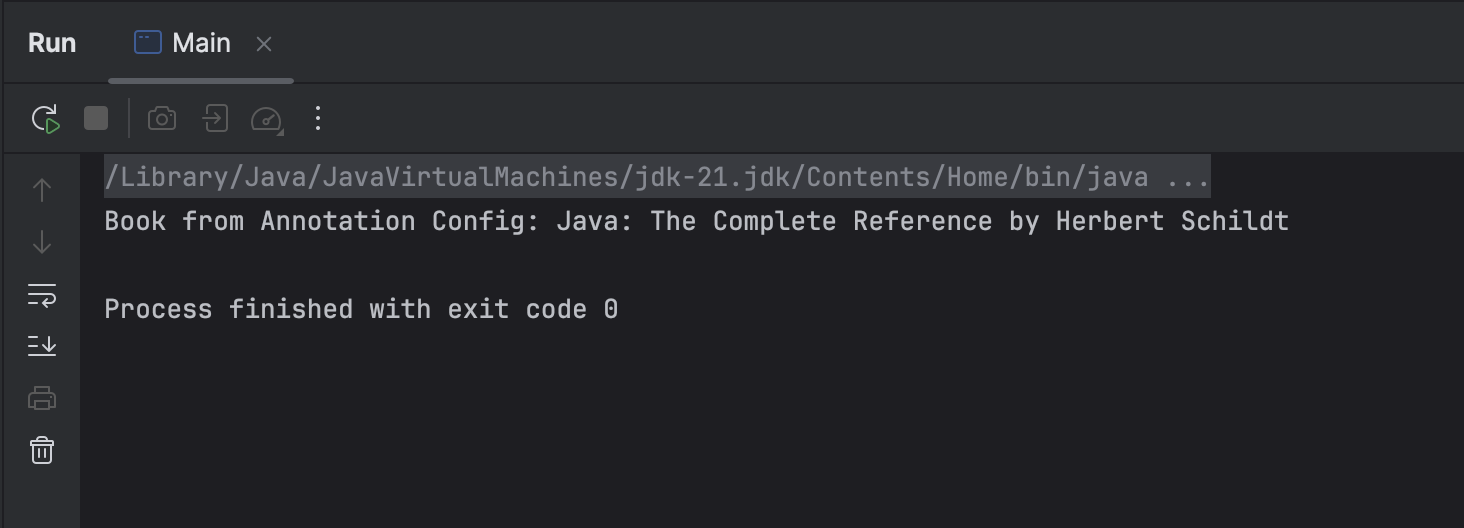
**BookService.java :**

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 {  
  
 private BookRepository bookRepository;  
  
 @Autowired  
 public BookService(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void displayBook() {  
 System.*out*.println("Book from Annotation Config: " + bookRepository.getBookTitle());  
 }  
}

**Main.java :**

import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean(BookService.class);  
 bookService.displayBook();  
 }  
}

**OUTPUT :**



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

**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"  
 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">  
 <constructor-arg ref="bookRepository"/>  
 <property name="libraryName" value="Public Library"/>  
 </bean>  
  
</beans>

**BookService.java :**

package com.library.service;  
  
import com.library.repository.BookRepository;  
  
public class BookService {  
 private BookRepository bookRepository;  
 private String libraryName;  
  
 public BookService(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void setLibraryName(String libraryName) {  
 this.libraryName = libraryName;  
 }  
  
 public void displayBook() {

System.*out*.println("Library Name: " + libraryName);  
 System.*out*.println("Book Title : " + bookRepository.getBookTitle());  
 }

}

**BookRepository.java :**

package com.library.repository;  
  
public class BookRepository {

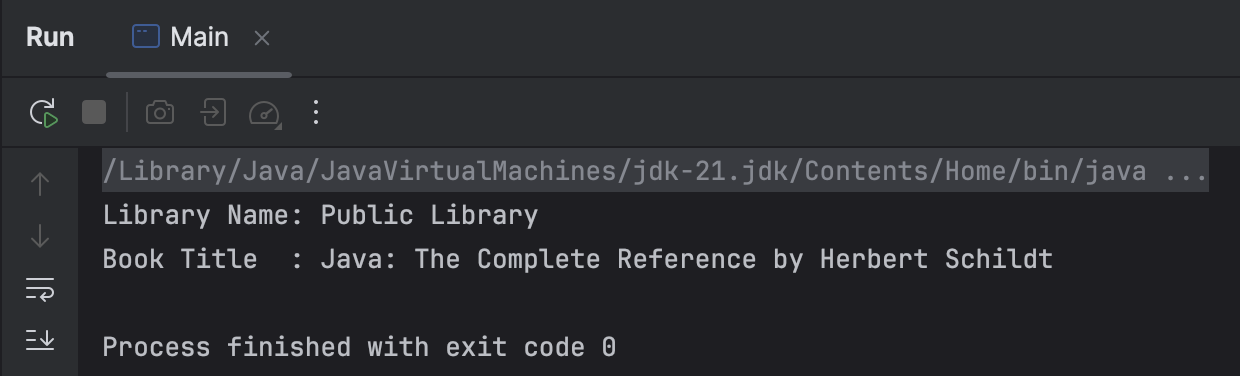
public String getBookTitle() {  
 return "Java: The Complete Reference by Herbert Schildt";  
 }

}

**Main.java :**

import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean("bookService", BookService.class);  
 bookService.displayBook();  
 }  
}

**OUTPUT:**



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

**CODE:**

**com.library.aspect - LoggingAspect.java :**

package com.library.aspect;  
  
import org.aspectj.lang.JoinPoint;  
import org.aspectj.lang.annotation.After;  
import org.aspectj.lang.annotation.Aspect;  
import org.aspectj.lang.annotation.Before;  
  
@Aspect  
public class LoggingAspect {  
  
 @Before("execution(\* com.library.service.\*.\*(..))")  
 public void beforeMethod(JoinPoint joinPoint) {  
 System.out.println("[BEFORE] Executing: " + joinPoint.getSignature().getName());  
 }  
  
 @After("execution(\* com.library.service.\*.\*(..))")  
 public void afterMethod(JoinPoint joinPoint) {  
 System.out.println("[AFTER] Completed: " + joinPoint.getSignature().getName());  
 }  
}

**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">  
  
 <context:component-scan base-package="com.library"/>  
  
 <aop:aspectj-autoproxy/>  
  
 <bean id="loggingAspect" class="com.library.aspect.LoggingAspect"/>  
  
</beans>

**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 {  
  
 private BookRepository bookRepository;  
  
 @Autowired  
 public BookService(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 public void displayBook() {  
 System.out.println("Book: " + bookRepository.getBookTitle());  
 }  
}

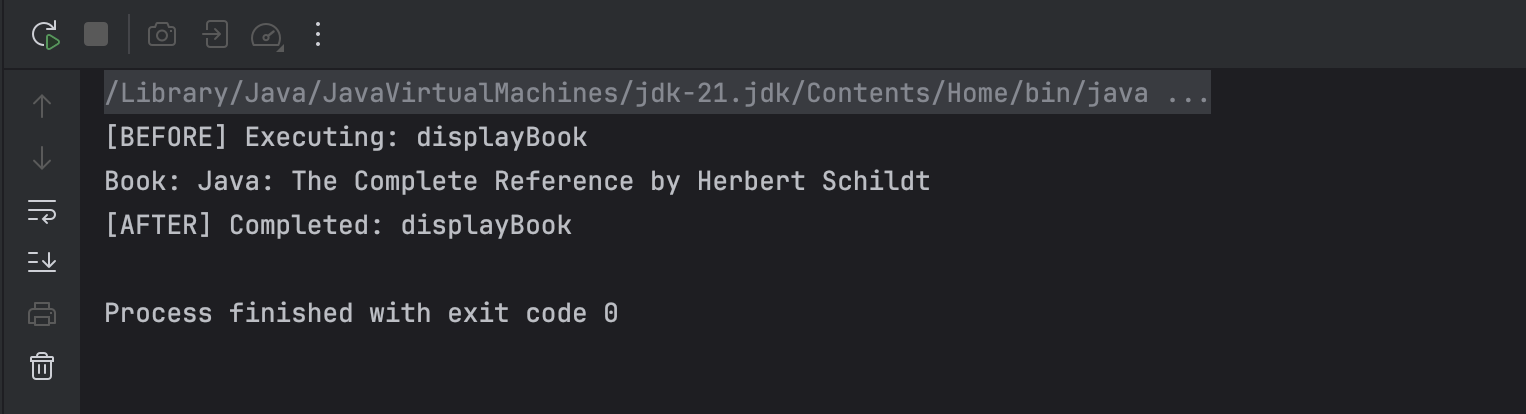
**BookRepository.java :**

package com.library.repository;  
  
import org.springframework.stereotype.Repository;  
  
@Repository  
public class BookRepository {  
 public String getBookTitle() {  
 return "Java: The Complete Reference by Herbert Schildt";  
 }  
}

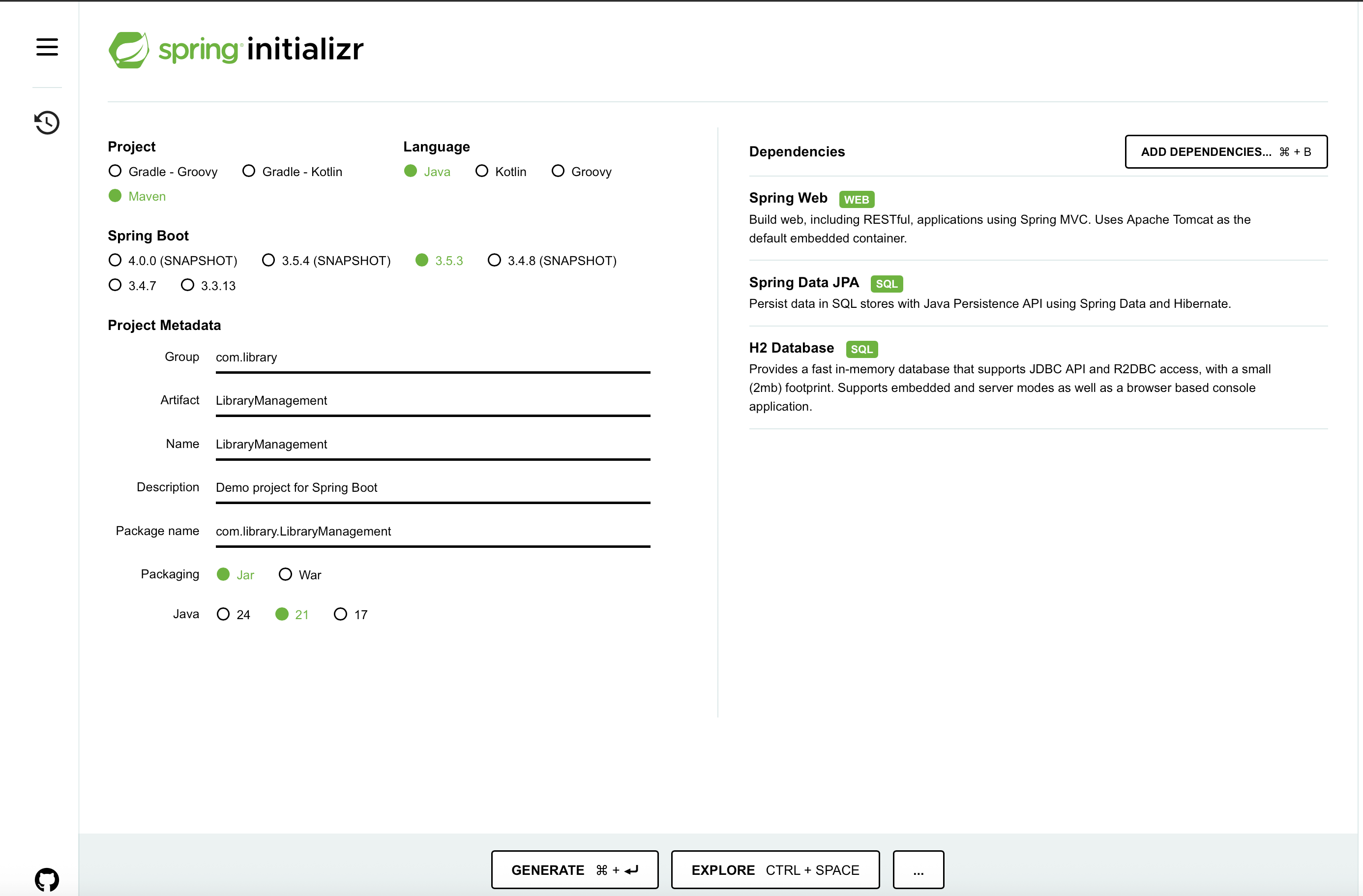
**Main.java :**

import com.library.service.BookService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Main {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
  
 BookService bookService = context.getBean(BookService.class);  
 bookService.displayBook();  
 }  
}

**OUTPUT:**



**Exercise 9 : Implementing Basic AOP with Spring**



**application.properties :**

spring.datasource.url=jdbc:h2:mem:librarydb  
spring.datasource.driverClassName=org.h2.Driver  
spring.datasource.username=sa  
spring.datasource.password=  
  
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect  
spring.jpa.hibernate.ddl-auto=update  
  
spring.h2.console.enabled=true  
spring.h2.console.path=/h2-console

**com.library.model : Book.java -**

package com.library.model;  
  
import jakarta.persistence.Entity;  
import jakarta.persistence.GeneratedValue;  
import jakarta.persistence.Id;  
  
@Entity  
public class Book {  
 @Id  
 @GeneratedValue  
 private Long id;  
  
 private String title;  
 private String author;  
  
 public Book() {}  
 public Book(String title, String author) {  
 this.title = title;  
 this.author = author;  
 }  
  
 public Long getId() { return id; }  
 public String getTitle() { return title; }  
 public String getAuthor() { return author; }  
  
 public void setId(Long id) { this.id = id; }  
 public void setTitle(String title) { this.title = title; }  
 public void setAuthor(String author) { this.author = author; }  
}

**com.library.repository : BookRepository.java -**

package com.library.repository;  
  
import com.library.model.Book;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface BookRepository extends JpaRepository<Book, Long> {  
}

**com.library.controller : BookController.java -**

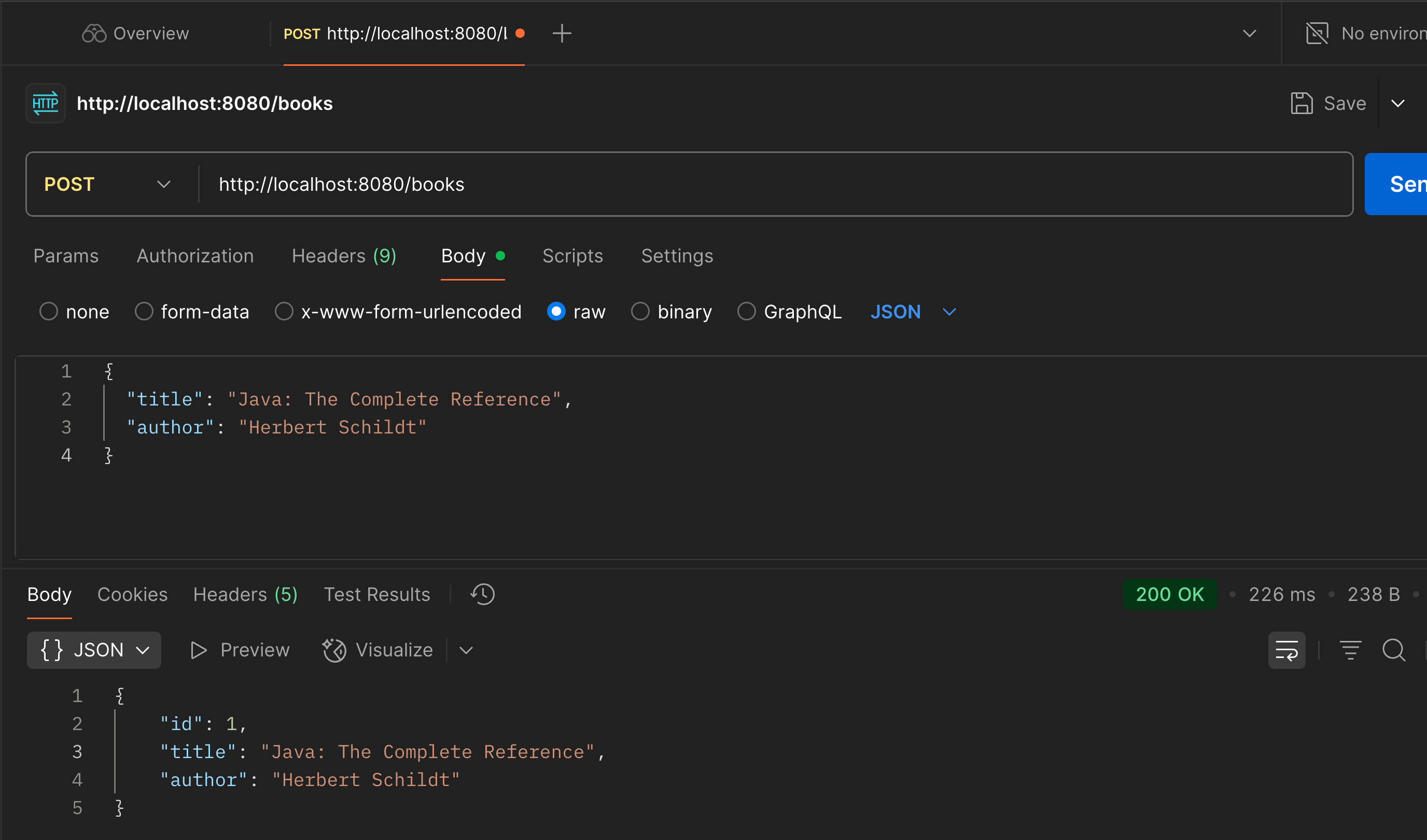
package com.library.controller;  
  
import com.library.model.Book;  
import com.library.repository.BookRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
@RestController  
@RequestMapping("/books")  
public class BookController {  
  
 @Autowired  
 private BookRepository bookRepository;  
  
 @PostMapping  
 public Book addBook(@RequestBody Book book) {  
 return bookRepository.save(book);  
 }  
  
 @PutMapping("/{id}")  
 public Book updateBook(@PathVariable Long id, @RequestBody Book updatedBook) {  
 return bookRepository.findById(id)  
 .map(book -> {  
 book.setTitle(updatedBook.getTitle());  
 book.setAuthor(updatedBook.getAuthor());  
 return bookRepository.save(book);  
 })  
 .orElseThrow(() -> new RuntimeException("Book not found with id: " + id));  
 }  
  
 @GetMapping  
 public List<Book> getAllBooks() {  
 return bookRepository.findAll();  
 }  
  
 @GetMapping("/{id}")  
 public Book getBook(@PathVariable Long id) {  
 return bookRepository.findById(id).orElse(null);  
 }  
  
 @DeleteMapping("/{id}")  
 public void deleteBook(@PathVariable Long id) {  
 bookRepository.deleteById(id);  
 }  
}

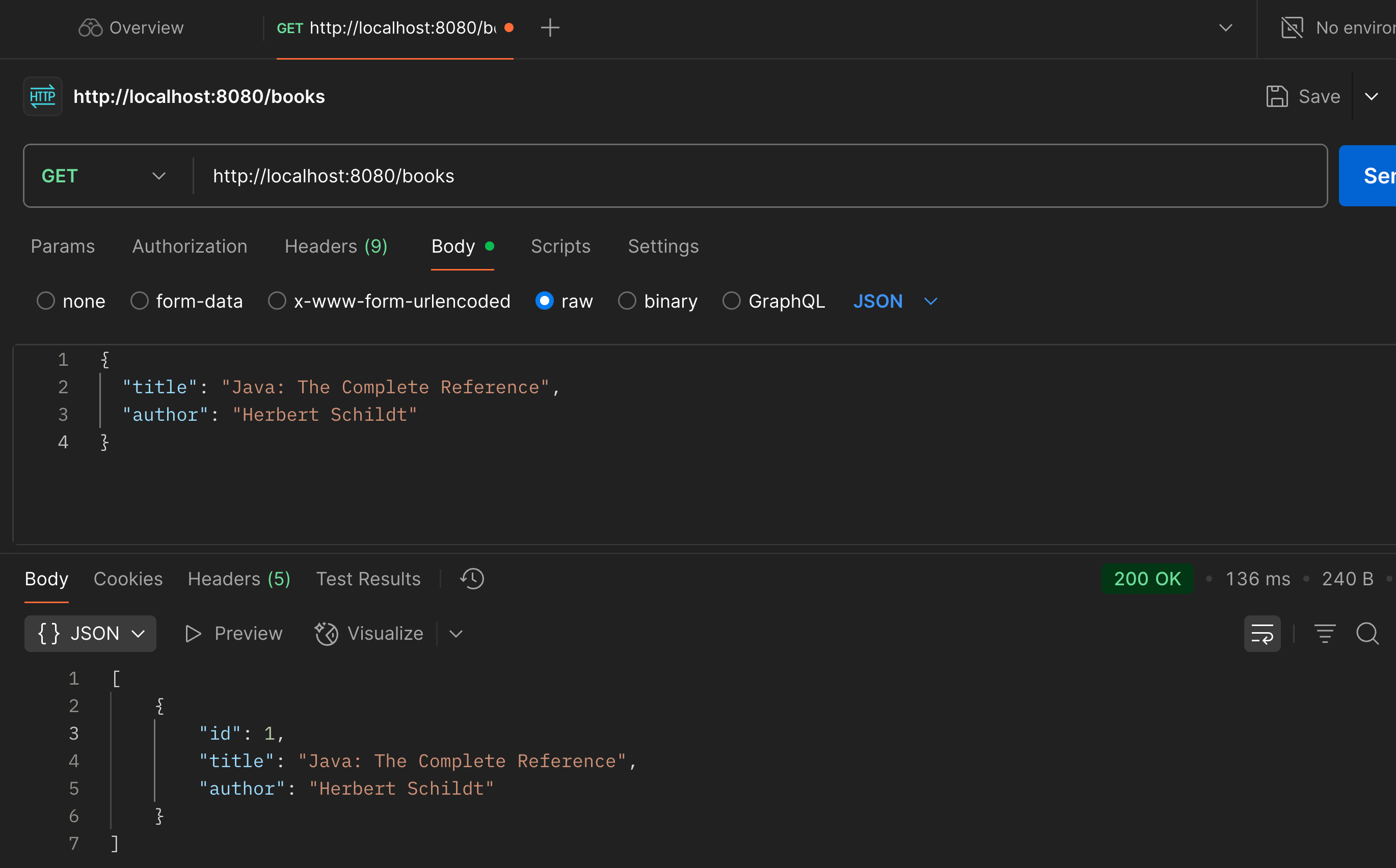
**LibraryManagementApplication.java :**

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);  
 }  
}

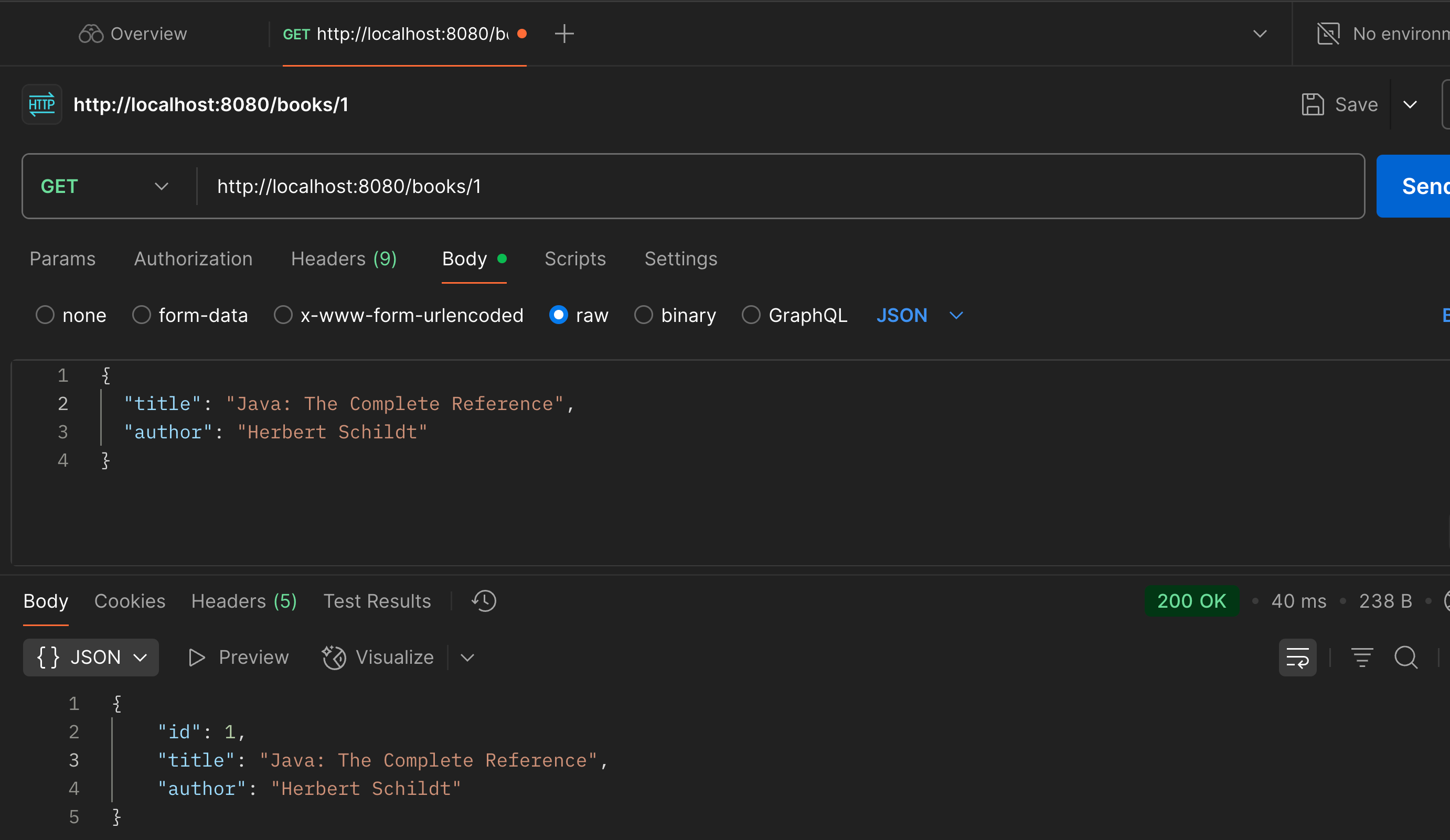
**OUTPUT : ( used Postman for testing )**

**POST Request :**

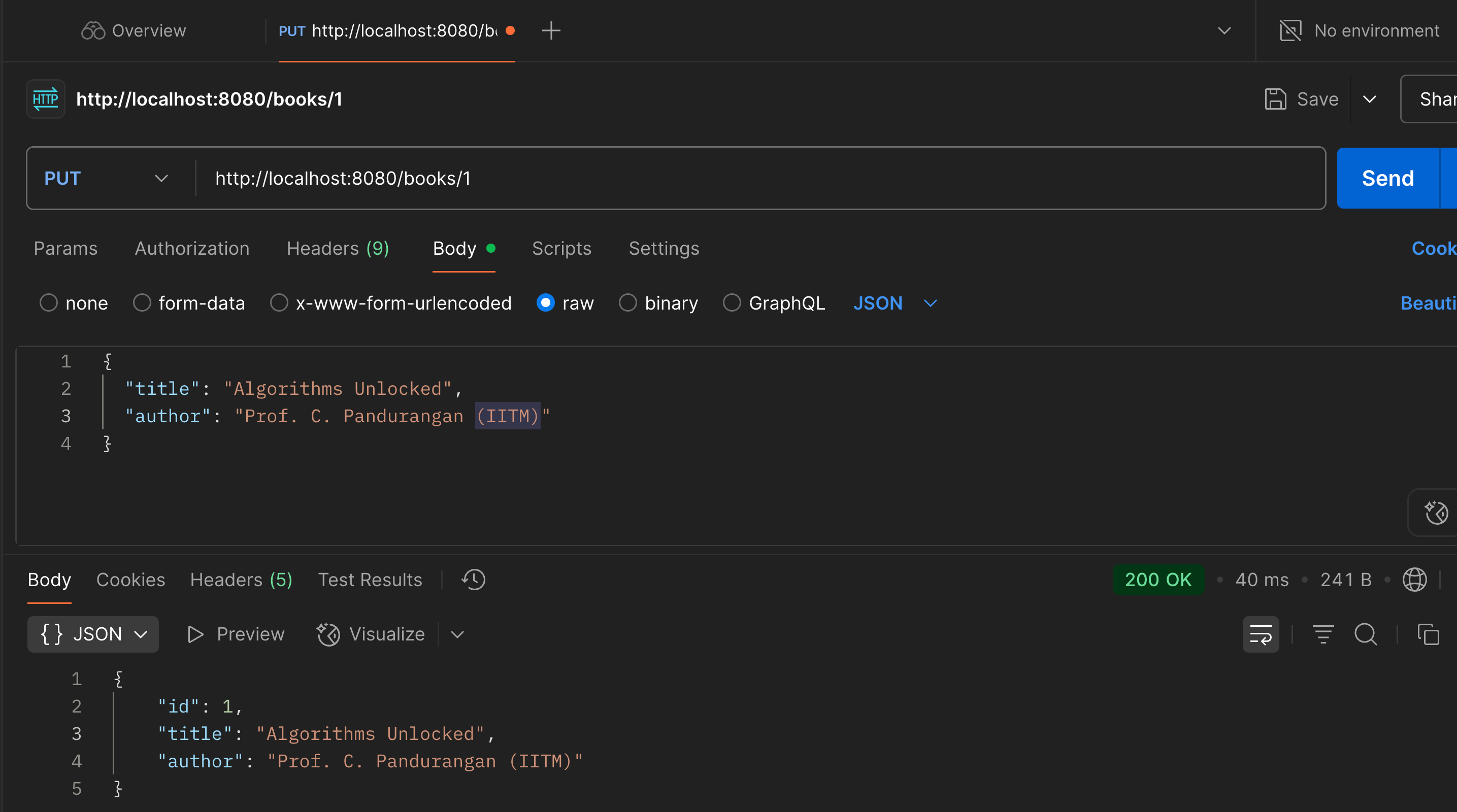


**GET All Request :**

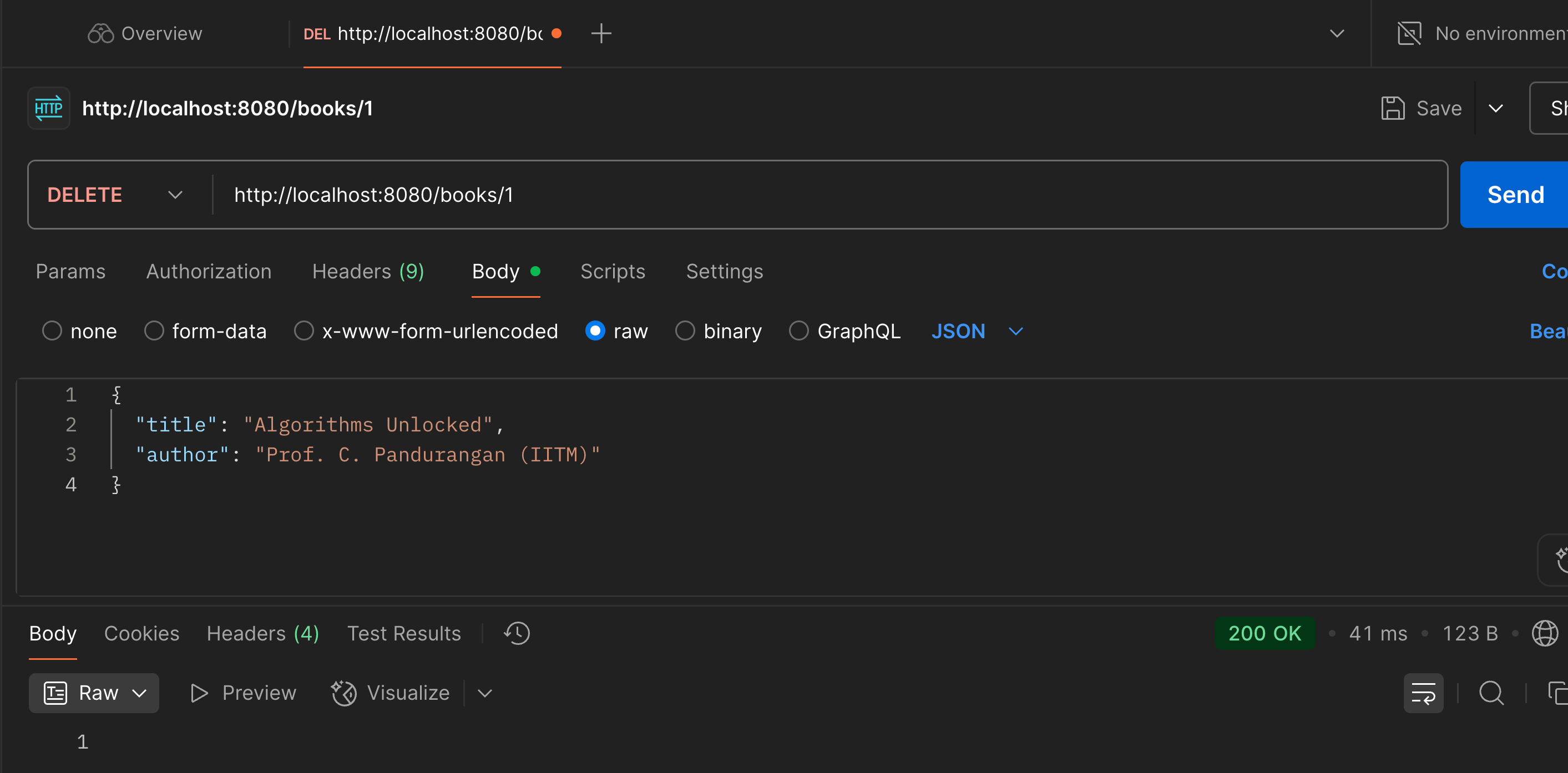
**GET by ID :**



**PUT Request :**



**DELETE Request :**



**Console Output in IntelliJ :**

