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

**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(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("Book saved: " + title);

}

}

**Main.java:**

**package** com.library;

**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");

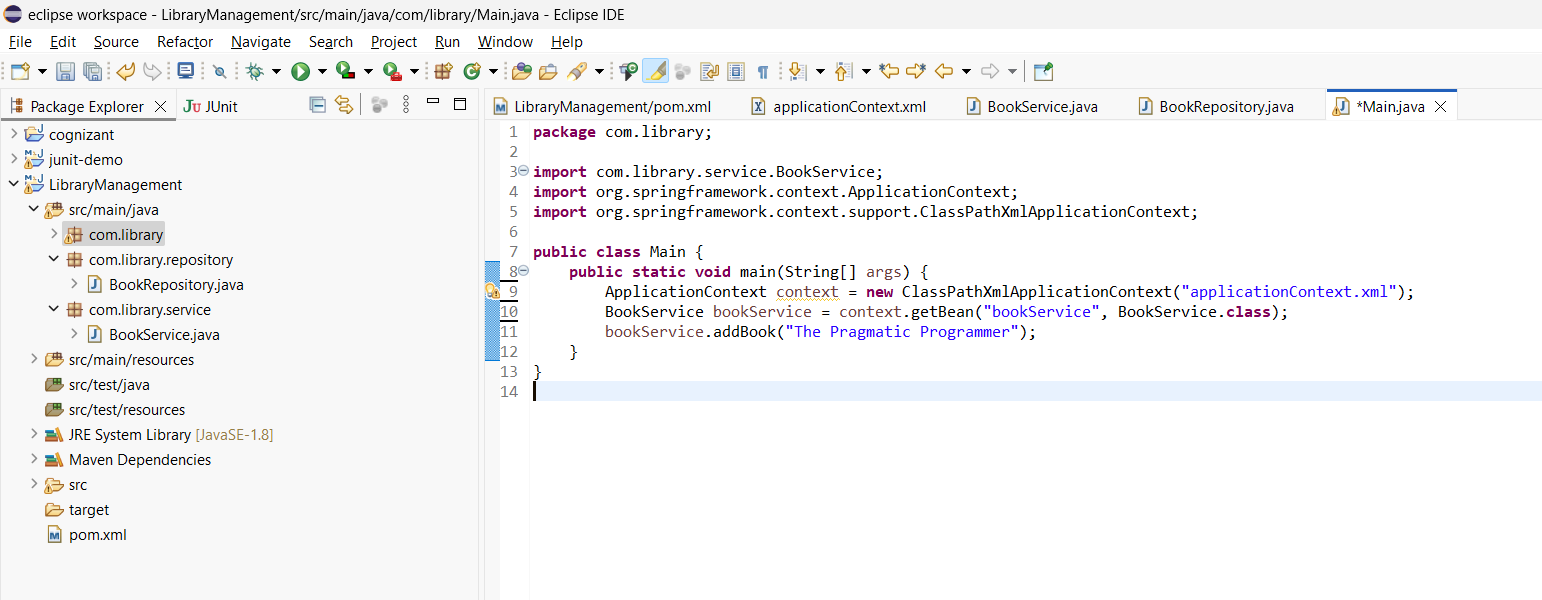
BookService bookService = context.getBean("bookService", BookService.**class**);

bookService.addBook("The Pragmatic Programmer");

}

}

**Output:**



**Exercise 2: Implementing Dependency Injection**

**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(String title) {

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

bookRepository.saveBook(title);

}

}

**LibraryManagementApplication.java:**

**package** com.library;

**import** com.library.service.BookService;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** LibraryManagementApplication {

**public** **static** **void** main(String[] args) {

**try** (ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("applicationContext.xml")) {

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

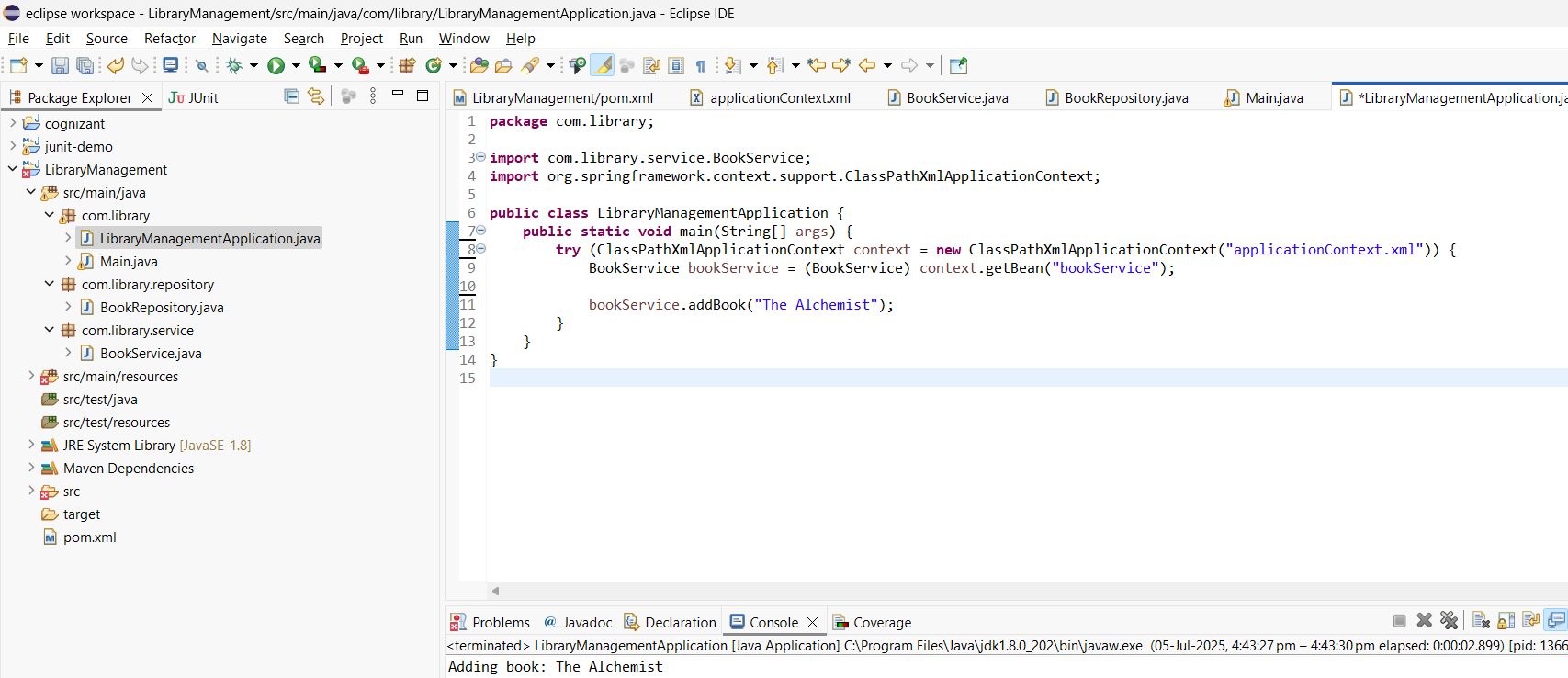
bookService.addBook("The Alchemist");

}

}

}

**Output:**

****

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

**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>com.library</groupId>

<artifactId>LibraryManagement</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.33</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.33</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.33</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>

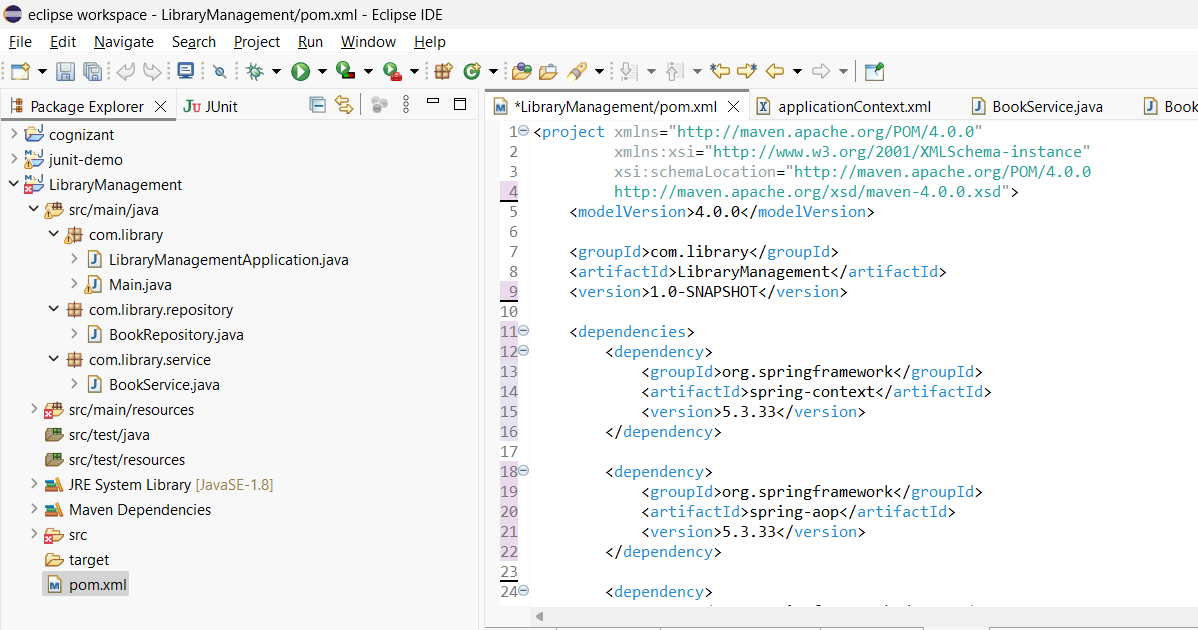
</plugin>

</plugins>

</build>

</project>

**OUTPUT:**



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

**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>

**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(String bookName) {

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

bookRepository.save(bookName);

}

}

**BookRepository.java:**

package com.library.repository;

public class BookRepository {

public void save(String bookName) {

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

}

}

**Main.java:**

**package** com.library;

**import** com.library.service.BookService;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

**try** (ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("applicationContext.xml")) {

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

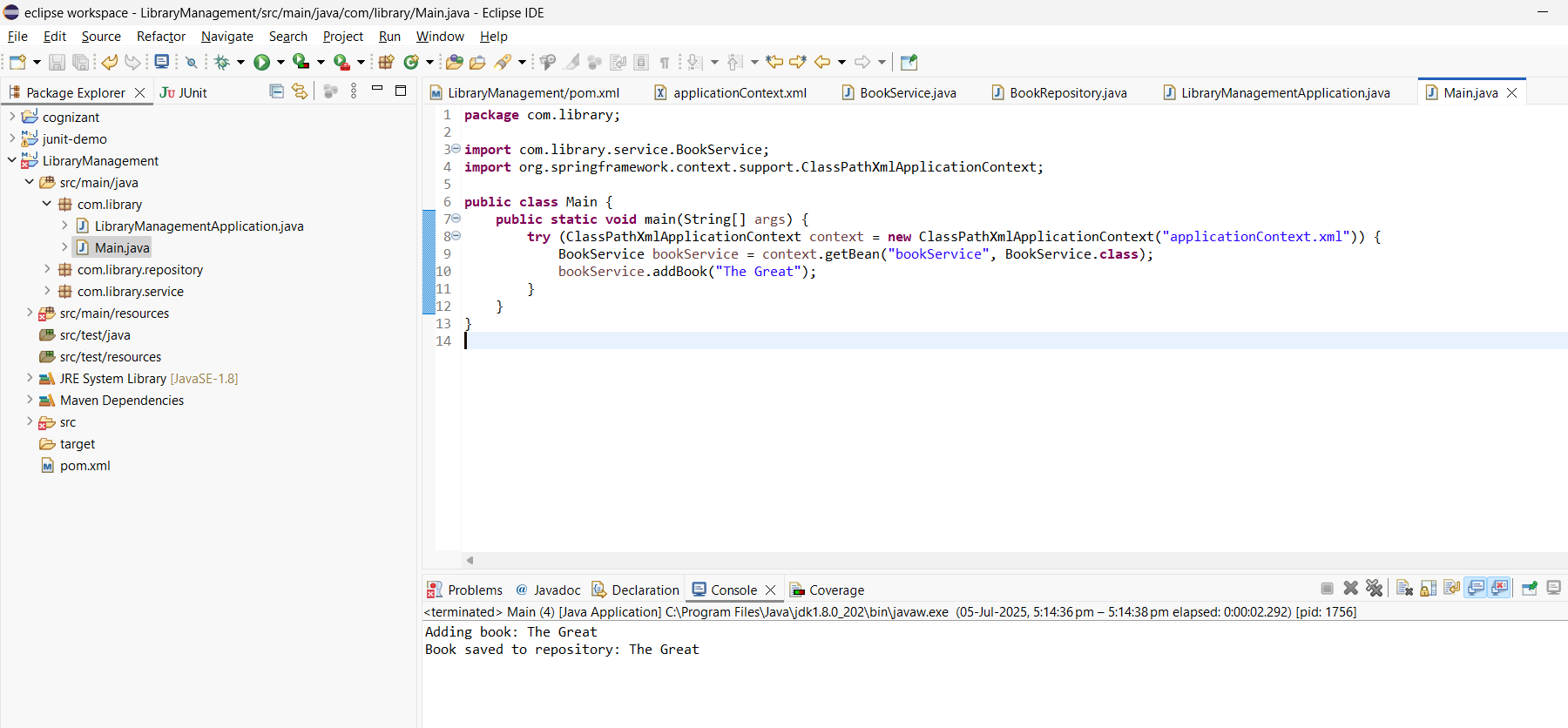
bookService.addBook("The Great Gatsby");

}

}

}

**OUTPUT:**



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

**BookService.java:**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private String libraryName;

private BookRepository bookRepository;

public BookService(String libraryName) {

this.libraryName = libraryName;

}

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String bookName) {

System.out.println("Library: " + libraryName);

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

bookRepository.save(bookName);

}

}

**Main.java:**

**package** com.library;

**import** com.library.service.BookService;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

**try** (ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("applicationContext.xml")) {

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

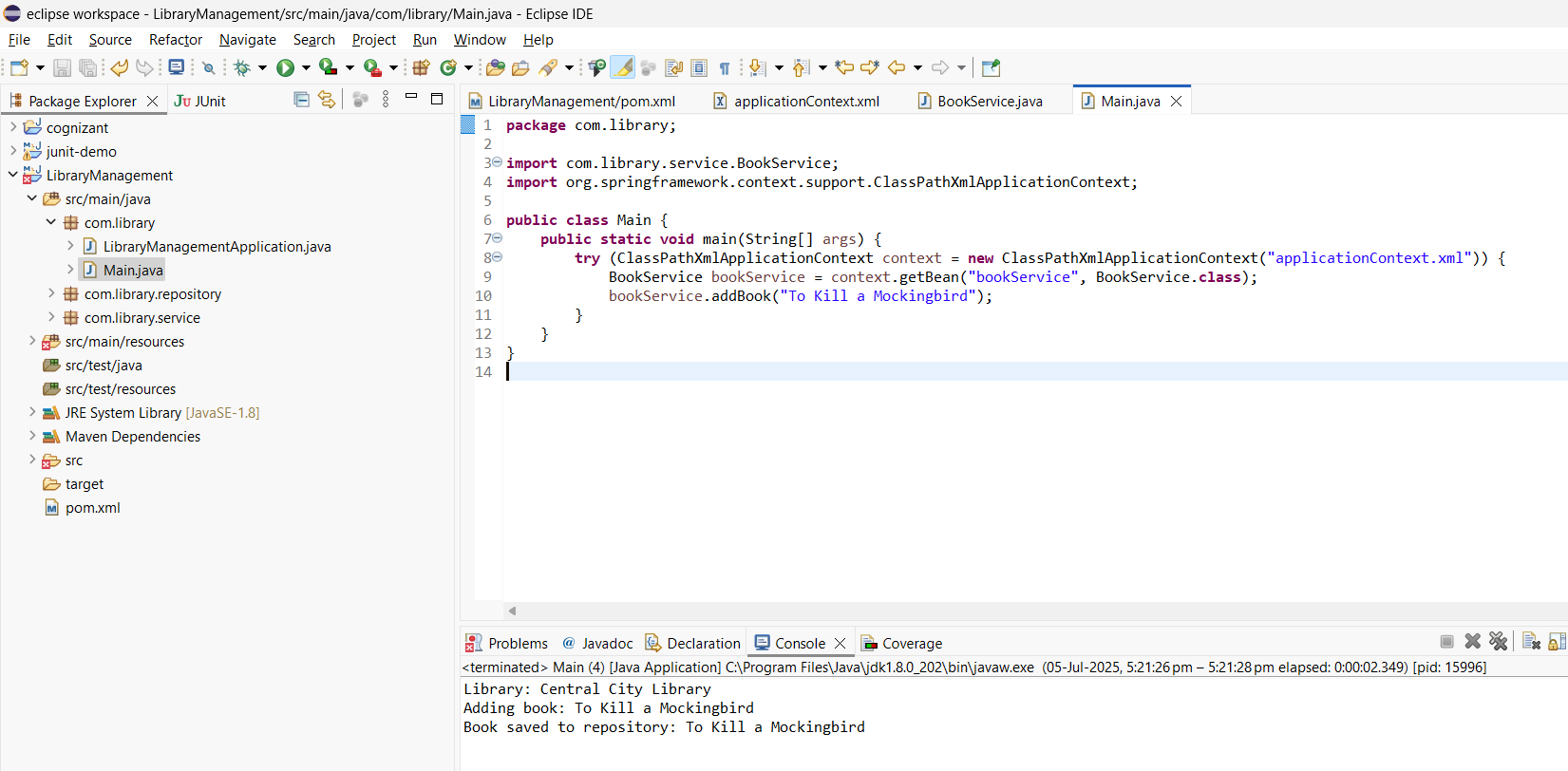
bookService.addBook("To Kill a Mockingbird");

}

}

}

Output:



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

**Book.java:**

**package** com.example.librarymanagement.model;

**import** jakarta.persistence.\*;

@Entity

**public** **class** Book {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**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** **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; }

}

**LibraryManagementApplication.java:**

**package** com.example.librarymanagement;

**import** com.example.librarymanagement.model.Book;

**import** com.example.librarymanagement.repository.BookRepository;

**import** org.springframework.boot.CommandLineRunner;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.annotation.Bean;

@SpringBootApplication

**public** **class** LibraryManagementApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(LibraryManagementApplication.**class**, args);

}

@Bean

CommandLineRunner runner(BookRepository repo) {

**return** args -> {

repo.save(**new** Book("Clean Code", "Robert "));

repo.save(**new** Book("Effective Java", "Joshua "));

};

}

}

**BookRepository.java:**

**package** com.example.librarymanagement.repository;

**import** com.example.librarymanagement.model.Book;

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

**public** **interface** BookRepository **extends** JpaRepository<Book, Long> {

}

**BookController.java:**

**package** com.example.librarymanagement.controller;

**import** com.example.librarymanagement.model.Book;

**import** com.example.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;

@GetMapping

**public** List<Book> getAllBooks() {

**return** bookRepository.findAll();

}

@PostMapping

**public** Book addBook(@RequestBody Book book) {

**return** bookRepository.save(book);

}

@GetMapping("/{id}")

**public** Book getBookById(@PathVariable Long id) {

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

}

@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);

}).orElse(**null**);

}

@DeleteMapping("/{id}")

**public** **void** deleteBook(@PathVariable Long id) {

bookRepository.deleteById(id);

}

}

