**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.library</groupId>  
 <artifactId>LibraryManagement</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>LibraryManagement</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <!-- Spring Data JPA -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
  
 <!-- H2 Database -->  
 <dependency>  
 <groupId>com.h2database</groupId>  
 <artifactId>h2</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>com.h2database</groupId>  
 <artifactId>h2</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**Application.properties**

spring.application.name=LibraryManagement  
# H2 database config  
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  
  
# Show SQL  
spring.jpa.show-sql=true  
spring.h2.console.enabled=true  
  
# Auto-create schema  
spring.jpa.hibernate.ddl-auto=update

**Book.java**

package com.library.entity;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Book {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String title;  
 private String author;  
  
 // Constructors  
 public Book() {}  
 public Book(String title, String author) {  
 this.title = title;  
 this.author = author;  
 }  
  
 // Getters and Setters  
 public Long getId() { return 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; }  
}

**BookRepository.java**

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

**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("/api/books")  
public class BookController {  
  
 @Autowired  
 private BookRepository bookRepository;  
  
 // Get all books  
 @GetMapping  
 public List<Book> getAllBooks() {  
 return bookRepository.findAll();  
 }  
  
 // Get book by ID  
 @GetMapping("/{id}")  
 public Book getBookById(@PathVariable Long id) {  
 return bookRepository.findById(id).orElse(null);  
 }  
  
 // Add new book  
 @PostMapping  
 public Book createBook(@RequestBody Book book) {  
 return bookRepository.save(book);  
 }  
  
 // Update 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);  
 })  
 .orElse(null);  
 }  
  
 // Delete book  
 @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);  
 }  
}

**OUTPUTS:**

A screenshot of a computer program

AI-generated content may be incorrect.

//PERFORMING CRUD OPERATIONS USING LOCAL PORT

GET - http://localhost:8080/api/books/1

|  |
| --- |
| PUT - <http://localhost:8080/api/books> |

DELETE - http://localhost:8080/api/books/1

POST - http://localhost:8080/api/books

A black and purple rectangle

AI-generated content may be incorrect.

A black screen with a black background

AI-generated content may be incorrect.

**View H2 Database**

Visit:  
📎 http://localhost:8080/h2-console

**JDBC URL**: jdbc:h2:mem:librarydb

**User**: sa

**Password**: *(leave blank)*A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.