Step 1: Create a Spring Boot Project

Use Spring Initializr to create a new Spring Boot project named LibraryManagement. You can do this by visiting the Spring Initializr website, selecting the desired dependencies, and generating the project.

Step 2: Add Dependencies

In the pom.xml file (if you're using Maven) or build.gradle file (if you're using Gradle), add the following dependencies:

Maven (pom.xml)

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

</dependency>

</dependencies>

Gradle (build.gradle)

dependencies {

implementation 'org.springframework.boot:spring-boot-starter-web'

implementation 'org.springframework.boot:spring-boot-starter-data-jpa'

implementation 'com.h2database:h2'

}

Step 3: Create Application Properties

Create a file named application.properties in the src/main/resources directory and add the following configuration:

spring.datasource.url=jdbc:h2:mem:library

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=create-drop

This configuration sets up an in-memory H2 database and enables Hibernate to create and drop the database schema.

Step 4: Define Entities and Repositories

Create a new package com.library.entity and add the following Book entity:

package com.library.entity;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String title;

private String author;

// getters and setters

}

Create a new package com.library.repository and add the following BookRepository interface:

package com.library.repository;

import com.library.entity.Book;

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

public interface BookRepository extends JpaRepository<Book, Long> {

}

Step 5: Create a REST Controller

Create a new package com.library.controller and add the following BookController class:

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.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController

public class BookController {

private final BookRepository bookRepository;

@Autowired

public BookController(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

@GetMapping("/books")

public List<Book> getAllBooks() {

return bookRepository.findAll();

}

@PostMapping("/books")

public Book createBook(@RequestBody Book book) {

return bookRepository.save(book);

}

}

Step 6: Run the Application

Run the Spring Boot application using the main method in the LibraryManagementApplication class:

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);

}

}