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

# **Java Files**

## **LibraryManagementApplication.java**

package com.example.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);  
 }  
}

## **Book.java**

package com.example.Library.entity;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Book {  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private Long id;  
  
 private String title;  
 private String author;  
 private String isbn;  
  
 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; }  
 public String getIsbn() { return isbn; }  
 public void setIsbn(String isbn) { this.isbn = isbn; }  
}

## **BookRepository.java**

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

## **BookController.java**

package com.example.Library.controller;  
  
import com.example.Library.entity.Book;  
import com.example.Library.repository.BookRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
import java.util.Optional;  
  
@RestController  
@RequestMapping("/books")  
public class BookController {  
  
 @Autowired  
 private BookRepository bookRepository;  
  
 @PostMapping  
 public Book addBook(@RequestBody Book book) {  
 return bookRepository.save(book);  
 }  
  
 @GetMapping  
 public List<Book> getAllBooks() {  
 return bookRepository.findAll();  
 }  
  
 @GetMapping("/{id}")  
 public Book getBookById(@PathVariable Long id) {  
 Optional<Book> book = bookRepository.findById(id);  
 return book.orElse(null);  
 }  
  
 @PutMapping("/{id}")  
 public Book updateBook(@PathVariable Long id, @RequestBody Book bookDetails) {  
 Optional<Book> optionalBook = bookRepository.findById(id);  
 if (optionalBook.isPresent()) {  
 Book book = optionalBook.get();  
 book.setTitle(bookDetails.getTitle());  
 book.setAuthor(bookDetails.getAuthor());  
 book.setIsbn(bookDetails.getIsbn());  
 return bookRepository.save(book);  
 } else {  
 return null;  
 }  
 }  
  
 @DeleteMapping("/{id}")  
 public void deleteBook(@PathVariable Long id) {  
 bookRepository.deleteById(id);  
 }  
}

# **Configuration Files**

## **application.properties**

# H2 in-memory database config  
spring.datasource.url=jdbc:h2:mem:librarydb  
spring.datasource.driverClassName=org.h2.Driver  
spring.datasource.username=sa  
spring.datasource.password=  
  
# JPA and Hibernate config  
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect  
spring.jpa.show-sql=true  
spring.jpa.hibernate.ddl-auto=update  
  
# H2 console (optional)  
spring.h2.console.enabled=true  
spring.h2.console.path=/h2-console  
  
# Optional: change port if needed  
server.port=8080

**Output:**



