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

**application.properties**

spring.application.name=LibraryManagement  
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  
spring.datasource.driver-class-name=org.h2.Driver

**Book.java**

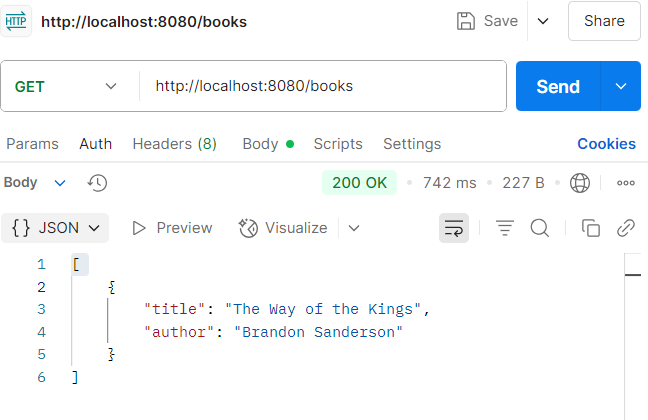
package com.library.entity;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Book {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String title;  
 private String author;

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.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/books")  
public class BookController {  
 private final BookRepository bookRepository;  
  
 public BookController(BookRepository bookRepository) {  
 this.bookRepository = bookRepository;  
 }  
  
 @GetMapping  
 public List<Book> getAllBooks() {  
 return bookRepository.findAll();  
 }  
  
 @PostMapping  
 public Book createBook(@RequestBody Book book) {  
 return bookRepository.save(book);  
 }  
  
 @PutMapping("/{id}")  
 public Book updateBook(@PathVariable Long id, @RequestBody Book bookDetails) {  
 Book book = bookRepository.findById(id).orElseThrow();  
 book.setTitle(bookDetails.getTitle());  
 book.setAuthor(bookDetails.getAuthor());  
 return bookRepository.save(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);  
 }  
}

**Output**

