**Exercise 4: Spring Data JPA Quick Example**

**Book.java**

package com.library.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 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.model.Book;

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

public interface BookRepository extends JpaRepository<Book, Long> {

Book findByTitle(String title);

}

**BookService.java**

package com.library.service;

import com.library.model.Book;

import com.library.repository.BookRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class BookService {

@Autowired

private BookRepository bookRepository;

public Book save(Book book) {

return bookRepository.save(book);

}

public List<Book> getAll() {

return bookRepository.findAll();

}

}

**LibraryManagementApplication.java**

package com.library;

import com.library.model.Book;

import com.library.service.BookService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class LibrarymanagementApplication implements CommandLineRunner {

@Autowired

private BookService bookService;

public static void main(String[] args) {

SpringApplication.run(LibrarymanagementApplication.class, args);

}

@Override

public void run(String... args) {

bookService.save(new Book("Clean Code", "Robert C. Martin"));

bookService.save(new Book("Effective Java", "Joshua Bloch"));

bookService.getAll().forEach(book ->

System.out.println(book.getTitle() + " by " + book.getAuthor()));

}

}

**application.properties**

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

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=update

spring.h2.console.enabled=true

**Book REST API – BookController.java**

package com.library.librarymanagement.controller;

import com.library.librarymanagement.model.Book;

import com.library.librarymanagement.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);

}

}

**Output:**



