МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ   
ІМЕНІ ВАСИЛЯ СТЕФАНИКА

Кафедра комп’ютерних наук та інформаційних систем

**Лабораторна робота №4**

з курсу «Платформи корпоративних інформаційних систем»

на тему:

«Використання Spring Data JPA.»

**Виконав:**

студент групи КН-41

Сенів П.В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис)

**Перевірив:**

викладач кафедри КНІС

к.т.н., доцент Горєлов В.О. «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оцінка, підпис)

м. Івано-Франківськ

2021

Мета: сформувати теоретичний базис щодо використання Spring Data JPA у Spring Boot проекті. Набути практичних навичок написання класів контролера, сервісу, репозиторію з підтримкою JPA.

Завдання: розробити проект з репозиторієм із підтримкою СУБД для реалізації каталогу домашньої бібліотеки. Робота продовжує тематику лабораторної роботи No2. Для реалізації скористайтеся JPA.

Хід роботи

1. Напишіть клас, що описує видання у каталозі домашньої бібліотеки. Поля класу: а) id;

б) назва книги;

в) автор;

г) рік видання;

2. Розробіть інтерфейс, котрий міститиме необхідні операції над даними. Перелік операцій:

а) додавання;

б) перегляд за id;

в) перегляд повного переліку;

г) видалення за id;

Передбачте методи, реалізація котрих базуватиметься на ключових словах JPA та методи на основі анотацій з JPQL.

3. Розробіть репозиторій на основі інтерфейсу (п. 2). Реалізуйте його методи. Репозиторій повинен використовувати сховище на основі СУБД.

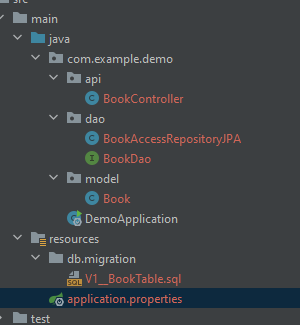
4. Побудуйте сервіс, що використовує отриманий у п. 3 репозиторій (або скористайтеся розробленим у лабораторній роботі No2).

5. Розробіть контролер для обробки запитів користувача (або скористайтеся розробленим у лабораторній роботі No2).

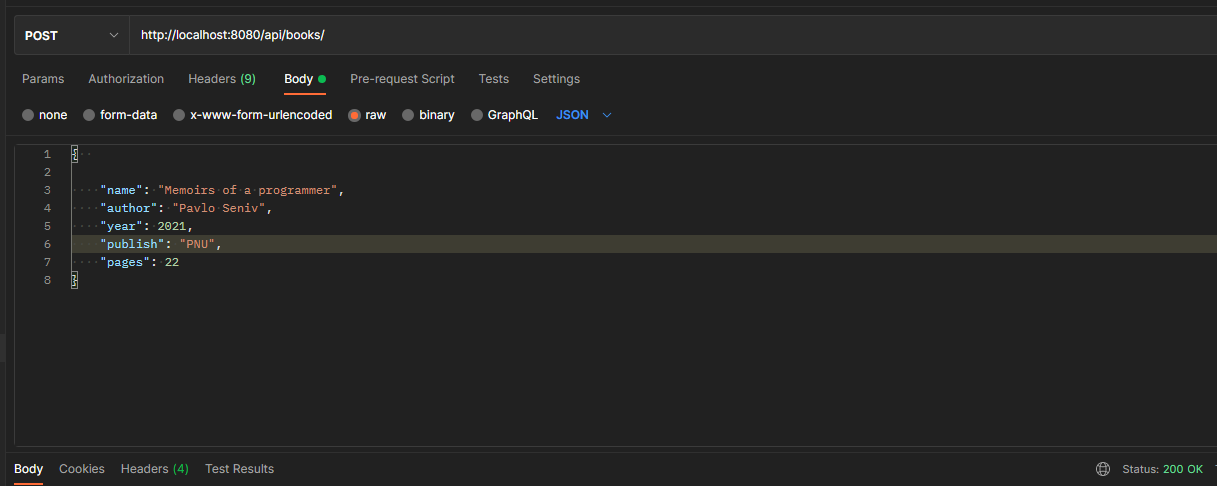
6. Продемонструйте розроблений проект у дії. Скористайтеся для надсилання запитів до сервера API-клієнтом (Postman, Insomnia тощо). Формат тіла HTTP-запитів та відповідей сервера — JSON.

7. Оформте звіт.

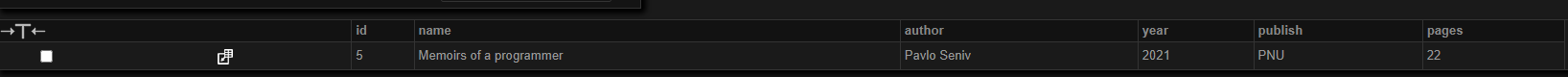
**Результати виконання роботи**



**Рисунок 1. Структура проекту.**



**Рисунок 2. Запити та приклад запиту.**



**Рисунок 3. Таблиця book.**

**Файл BookController.java**

package com.example.demo.api;  
  
import com.example.demo.dao.BookDao;  
import com.example.demo.model.Book;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.ArrayList;  
import java.util.List;  
  
@RequestMapping("/api/books")  
@RestController  
public class BookController {  
 private final BookDao BookRepository;  
  
 @Autowired  
 public BookController(BookDao BookRepository) {  
 this.BookRepository = BookRepository;  
 }  
  
 @PostMapping  
 public void add(@RequestBody Book book) {  
 BookRepository.save(book);  
 }  
  
 @PutMapping(path = "{id}")  
 public void modifyById(@PathVariable int id, @RequestBody Book book) {  
 book.setId(id);  
 BookRepository.save(book);  
 }  
  
 @DeleteMapping  
 public void deleteAll() {  
 BookRepository.deleteAll();  
 }  
  
 @DeleteMapping(path = "id/{id}")  
 public void deleteById(@PathVariable("id") int id) {  
 BookRepository.deleteById(id);  
 }  
  
 @GetMapping  
 public List<Book> getAll() {  
 var iter = BookRepository.findAll().iterator();  
 List<Book> list = new ArrayList<>();  
 while (iter.hasNext()) list.add(iter.next());  
 return list;  
 }  
  
 @GetMapping(path = "id/{id}")  
 public Book getById(@PathVariable("id") int id) {  
 return BookRepository.findById(id).get();  
 }  
  
 @GetMapping(path = "author/{author}")  
 public List<Book> getByAuthor(@PathVariable("author") String author) {  
 return BookRepository.findByAuthor(author).get();  
 }  
  
 @GetMapping(path = "publish/{publish}")  
 public List<Book> getByPublish(@PathVariable("publish") String publish) {  
 return BookRepository.findByPublish(publish).get();  
 }  
  
}

**Файл Book.java**

package com.example.demo.model;  
  
import com.fasterxml.jackson.annotation.JsonProperty;  
  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
  
@Entity  
public class Book {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 private Integer id;  
 private Integer year;  
 private Integer pages;  
 private String name;  
 private String author;  
 private String publish;  
  
 public Book(  
 @JsonProperty("id") int id,  
 @JsonProperty("year") int year,  
 @JsonProperty("pages") int pages,  
 @JsonProperty("name") String name,  
 @JsonProperty("author") String author,  
 @JsonProperty("publish") String publish  
 ) {  
 this.id = 0;  
 this.year = year;  
 this.pages = pages;  
 this.name = name;  
 this.author = author;  
 this.publish = publish;  
 }  
  
 public Book() {  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public int getYear() {  
 return year;  
 }  
  
 public int getPages() {  
 return pages;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public String getAuthor() {  
 return author;  
 }  
  
 public String getPublish() {  
 return publish;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public void setYear(int year) {  
 this.year = year;  
 }  
  
 public void setPages(int pages) {  
 this.pages = pages;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
  
 public void setPublish(String publish) {  
 this.publish = publish;  
 }  
  
}

**Висновок:** на лабораторній роботі було розроблено проект з репозиторієм із підтримкою СУБД для реалізації каталогу домашньої бібліотеки. Робота продовжує тематику лабораторної роботи No2. Для реалізації скористався JPA. Я закріпив навички користування API-клієнтом Postman. Надсилав запити до сервера API-клієнтом Postman. Також із зміною даних за допомогою запитів, змінювалися дані в таблиці book у СУБД.