МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ   
ІМЕНІ ВАСИЛЯ СТЕФАНИКА

Кафедра комп’ютерних наук та інформаційних систем

**Лабораторна робота №3**

з курсу «Платформи корпоративних інформаційних систем»

на тему:

«Використання Spring Data JPA. Базові поняття.»

**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: набути практичних навичок з використання контролера, сервісу та  репозиторію у Spring Boot.

Завдання: розробити API для проведення регресійного аналізу на боці  сервера. Для цього: налаштувати під’єднання до СУБД, розробити клас моделі  даних, реалізувати базові операції над даними та контролер для обробки HTTP запитів.

Хід роботи

1. Створіть клас моделі даних (масиви x та y, y=f(x)).

2. Реалізуйте інтерфейс, в якому передбачте потрібні операції над даними з  викориcтанням JPA (CrudRepository). Передбачте наступні операції: - додавання послідовностей y(x). Перед додаванням нової послідовності  видаляйте стару;

- отримання збережених послідовностей;

3. Розробіть контролер для обробки HTTP-запитів користувача. Формат  передачі даних – JSON.

4. Реалізуйте можливість додати до бази даних послідовності x та y. За  запитом користувача, котрий містить довільне значення x у діапазоні можливих  значень, що містить таблиця бази даних, сервер отримує збережені послідовності  x та y, знаходить коефіцієнти рівняння регресії та повертає розраховане значення  y.

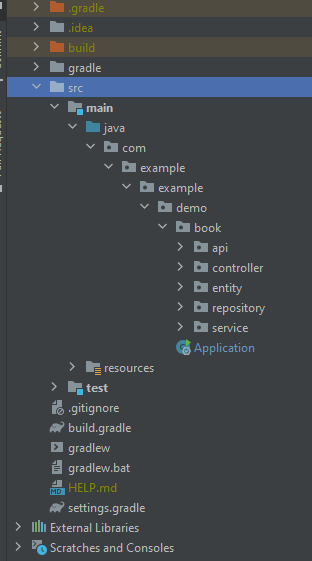
5. Продемонструйте виконану роботу з використанням HTTP-клієнта  (наприклад, Postman).

6. Підготуйте звіт. До звіту додайте архів проекту (zip чи 7z) та відеозапис з  екрану, що демонструє запуск проекту на сервері, надсилання запитів та отримання  відповідей сервера. Для запису відео скористайтеся або вбудованими засобами  операційної системи, або довільним безкоштовним програмним забезпеченням на  власний розсуд.

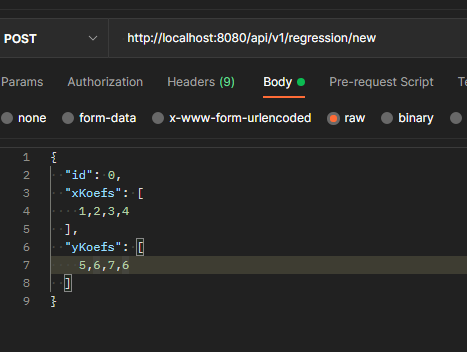
Висновки

Висновки повинні мати практичний характер і не бути анотацією до  лабораторної роботи.

**Результати виконання роботи**



**Рисунок 1. Структура проекту.**



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

**Файл RegressionController.java**

package com.example.example.demo.book.controller;  
  
import com.example.example.demo.book.api.RegressionApi;  
import com.example.example.demo.book.entity.Koefs;  
import com.example.example.demo.book.repository.KoefsRepository;  
import com.example.example.demo.book.service.KoefsService;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.RestController;  
  
import java.util.List;  
import java.util.Optional;  
  
@RestController  
public class RegressionController implements RegressionApi {  
 private final KoefsService koefsService;  
  
 public RegressionController(KoefsService koefsService) {  
 this.koefsService = koefsService;  
 }  
  
 @Override  
 public ResponseEntity<Koefs> findKoefs(Long id) {  
 return ResponseEntity.*ok*(koefsService.findKoefs(id));  
 }  
  
 @Override  
 public ResponseEntity<Koefs> addXYKoefs(Koefs koefs) {  
 return ResponseEntity.*ok*(koefsService.addKoefs(koefs));  
 }  
}

**Файл KoefsService.java**

package com.example.example.demo.book.service;  
  
import com.example.example.demo.book.entity.Koefs;  
import com.example.example.demo.book.entity.KoefsResponse;  
  
import java.util.List;  
  
public interface KoefsService {  
 Koefs addKoefs(Koefs koefs);  
  
 KoefsResponse findKoefs(Long id);  
}

**Файл Koefs.java**

package com.example.example.demo.book.entity;  
  
import javax.persistence.\*;  
import java.util.List;  
  
@Entity  
public class Koefs {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 private Long id;  
 @ElementCollection  
 private List<Integer> xKoefs;  
 @ElementCollection  
 private List<Integer> yKoefs;  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public List<Integer> getxKoefs() {  
 return xKoefs;  
 }  
  
 public void setxKoefs(List<Integer> xKoefs) {  
 this.xKoefs = xKoefs;  
 }  
  
 public List<Integer> getyKoefs() {  
 return yKoefs;  
 }  
  
 public void setyKoefs(List<Integer> yKoefs) {  
 this.yKoefs = yKoefs;  
 }  
}

**Висновок:** на лабораторній роботі було створено клас моделі даних. Реалізував інтерфейс, в якому розробив операції додавання послідовностей y(x). 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.