МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ   
ІМЕНІ ВАСИЛЯ СТЕФАНИКА

Кафедра комп’ютерних наук та інформаційних систем

**Лабораторна робота №5**

з курсу «Платформи корпоративних інформаційних систем»

на тему:

«Завантаження файлів на сервер.»

**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 завантаження файлів на сервер з використання фреймворку Spring Boot. Набути практичних навичок завантаження файлів на сервер.

Завдання: розробити проект із підтримкою JPA для опрацювання Kml- (або GPX)-файлів. Для цього скористайтеся результатами лабораторної роботи No 3. Реалізуйте завантаження файлів на сервер. Реалізуйте у контролері обробку запиту, у кому вказують ім’я файла для порівняння із наявними у базі даних.

Хід роботи

1. Додайте GPX (KML) файл до ресурсів на сервері (реалізуйте аплоад).

2. Валідуйте даний XML за відповідною йому схемою.

3. Пропарсіть файл, отримайте дані, що описують трек (пари координат

“довгота”/“широта”.

4. Перетворіть форму представлення даних з числової у символьну послідовність згідно із доданим до роботи матеріалом. Скористайтеся методом поділу координат навпіл.

5. Розробіть клас для збереження даних з використанням JPA. Клас повинен містити загальний порядковий номер запису, номер треку, котрому належить запис, та символьну послідовність, що описує локацію точки треку. Уникайте дублювання записів у межах одного і того ж треку.

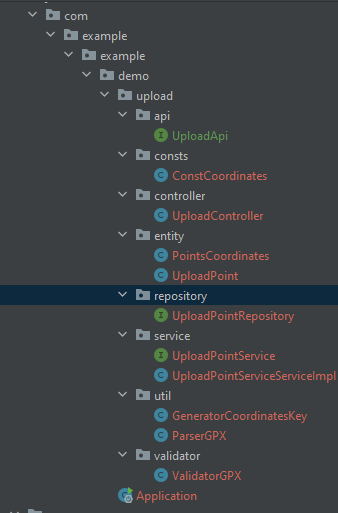
6. Занесіть до таблиці бази даних отримані результати перетворення.

7. Скористайтеся HTTP-запитом, у якому передають ім’я файла. Результатом виконання запиту повинен бути перелік подібних треків, інформація про які наявна у базі даних.

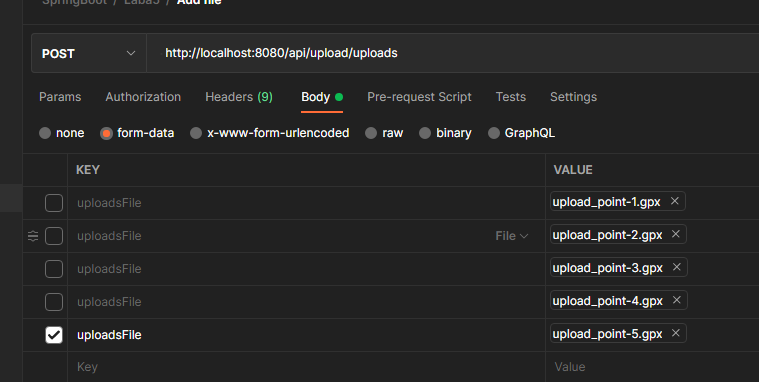
8. Продемонструйте розроблений проект у дії. Скористайтеся для надсилання запитів до сервера API-клієнтом (Postman, Insomnia тощо). Формат тіла HTTP- запитів та відповідей сервера — JSON. Формат демонстрації: 1) завантаження файла на сервер; 2) HTTP-запит, у якому серверові передають ім’я-треку і отримують відповідь у вигляді переліку подібних до нього трекі.

7. Оформте звіт та зробіть висновки.

**Результати виконання роботи**



**Рисунок 1. Структура проекту.**



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

**Файл UploadApi.java**

package com.example.example.demo.upload.api;  
  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
import org.springframework.web.multipart.MultipartFile;  
  
import java.util.List;  
  
@RequestMapping("/api/upload")  
public interface UploadApi {  
 @GetMapping(value = "/every")  
 @ResponseStatus(HttpStatus.*OK*)  
 ResponseEntity<List<String>> getAllTracks();  
  
 @GetMapping(value = "/onePoints={onePoints}&twoPoints={twoPoints}")  
 @ResponseStatus(HttpStatus.*OK*)  
 ResponseEntity<List<String>> getMatchingPoints(@PathVariable String onePoints, @PathVariable String twoPoints);  
  
 @PostMapping("/uploads")  
 @ResponseStatus(HttpStatus.*CREATED*)  
 ResponseEntity<String> uploadNewTrack(@RequestParam("uploadsFile") MultipartFile file);  
}

**Файл UploadController.java**

package com.example.example.demo.upload.controller;  
  
import com.example.example.demo.upload.api.UploadApi;  
import com.example.example.demo.upload.service.UploadPointService;  
import com.example.example.demo.upload.validator.ValidatorGPX;  
import org.apache.commons.io.FilenameUtils;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.RestController;  
import org.springframework.web.multipart.MultipartFile;  
  
import java.io.File;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.util.List;  
  
@RestController  
public class UploadController implements UploadApi {  
 private static String *uploadDirectory* = System.*getenv*("file\_upd\_dir") + "/uploads";  
 @Autowired  
 private UploadPointService uploadPointService;  
  
 @Override  
 public ResponseEntity<List<String>> getAllTracks() {  
 return ResponseEntity.*ok*(uploadPointService.getAllTracks());  
 }  
  
 @Override  
 public ResponseEntity<List<String>> getMatchingPoints(String onePoints, String twoPoints) {  
 return ResponseEntity.*ok*(uploadPointService.compareTracks(onePoints, twoPoints));  
 }  
  
 @Override  
 public ResponseEntity<String> uploadNewTrack(MultipartFile file) {  
 String fileExt = FilenameUtils.*getExtension*(file.getOriginalFilename());  
 String fineN = FilenameUtils.*removeExtension*(FilenameUtils.*getName*(file.getOriginalFilename()));  
 System.*out*.println("File name: " + fineN);  
  
 try {  
 File f = new File(*uploadDirectory*);  
 if (f.exists()) {  
 f.delete();  
 }  
 f.mkdir();  
  
  
 Path fileNamePath = Paths.*get*(*uploadDirectory*, file.getOriginalFilename());  
 Files.*write*(fileNamePath, file.getBytes());  
  
 if (ValidatorGPX.*isValidExtension*(fileExt) && ValidatorGPX.*isValidSchema*(fileNamePath)) {  
 uploadPointService.createTrackPoints(fileNamePath.toString(), FilenameUtils.*removeExtension*(FilenameUtils.*getName*(file.getOriginalFilename())));  
 return ResponseEntity.*ok*("File add to db !");  
 }  
  
 return ResponseEntity.*ok*("File uploads");  
 } catch (Exception e) {  
 e.printStackTrace();  
 return ResponseEntity.*badRequest*().body("Error url: " + e.getMessage());  
 }  
 }  
}

**Файл UploadPoint.java**

package com.example.example.demo.upload.entity;  
  
import javax.persistence.\*;  
  
@Entity  
@Table(name = "uploadPoint")  
public class UploadPoint {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 private Long id;  
 private String pointsName;  
 private String pointsString;  
  
 public UploadPoint() {  
 }  
  
 public UploadPoint(String pointsName, String pointsString) {  
 this.pointsName = pointsName;  
 this.pointsString = pointsString;  
 }  
}

**Файл PointsCoordinates.java**

package com.example.example.demo.upload.entity;  
  
public class PointsCoordinates {  
 private double lat;  
 private double lon;  
  
 public PointsCoordinates(double lat, double lon) {  
 this.lat = lat;  
 this.lon = lon;  
 }  
  
 public double getLat() {  
 return lat;  
 }  
  
 public double getLon() {  
 return lon;  
 }  
  
 @Override  
 public String toString() {  
 return "PointsCoordinates{" +  
 "lat=" + lat +  
 ", lon=" + lon +  
 '}';  
 }  
}

**Висновок:** на лабораторній роботі було розроблено проект із підтримкою JPA для опрацювання GPX-файлів. Реалізував завантаження файлів на сервер. Додав до таблиці бази даних отримані результати перетворення. Виконав запити на витягнення однакових символьних рядків із файлів.