

**SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY**

Evidenčné číslo: FEI-16607-104434

**VIZUALIZÁCIA POHYBU POUŽÍVATEĽOV
DIPLOMOVÁ PRÁCA**

2024

Bc. Richard Búri

**SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY**

Evidenčné číslo: FEI-16607-104434

**VIZUALIZÁCIA POHYBU POUŽÍVATEĽOV
DIPLOMOVÁ PRÁCA**

Študijný program: Aplikovaná informatika
Názov študijného odboru: Informatika
Školiace pracovisko: Ústav informatiky a matematiky
Vedúci záverečnej práce: Ing. Maroš Čavojský, PhD.

Bratislava 2024

Bc. Richard Búri



ZADANIE DIPLOMOVEJ PRÁCE

Študent: **Bc. Richard Búri**

ID študenta: 104434

Študijný program: aplikovaná informatika

Študijný odbor: informatika

Vedúci práce: Ing. Maroš Čavojský, PhD.

Vedúci pracoviska: doc. Ing. Milan Vojvoda, PhD.

Miesto vypracovania: Ústav informatiky a matematiky

Názov práce: **Vizualizácia pohybu používateľov**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

Cieľom práce je navrhnuť a implementovať algoritmus, ktorý bude schopný vizualizovať väčšie datasety GPS trajektórií prehľadne (používateľ bude schopný zrakovo zistíť základné pohybové vzory z trajektórií). Výstup algoritmu by mal byť zobrazený interaktívny spôsobom (priblížovanie, oddaľovanie) v internetovom prehliadači.

Úlohy:

1. Naštudujte si aktuálne spôsoby pri zobrazovaní väčšieho množstva GPS trajektórií v spoločnej oblasti.
2. Navrhnite algoritmus pre analýzu vstupných dát, pre ich následnú vhodnú vizualizáciu v internetovom prehliadači.
3. Implementujte navrhnutý algoritmus ako aj zobrazovanie výstupu algoritmu interaktívny spôsobom v internetovom prehliadači
4. Otestujte a vyhodnoťte implementovaný algoritmus a zobrazovanie v internetovom prehliadači pre rôzne GPS datasety.

Zoznam odbornej literatúry:

1. ČAVOJSKÝ, Maroš; DROZDA, Martin. Search by pattern in GPS trajectories. In: TAHERI, Javid; VILLARI, Massimo; GALLETTA, Antonino. *Mobile Computing, Applications, and Services*. Cham: Springer, 2023, s. 117–132. ISBN 978-3-031-31890-0.
2. ČAVOJSKÝ, Maroš; DROZDA, Martin; BALOGH, Zoltán. Analysis and experimental evaluation of the Needleman-Wunsch algorithm for trajectory comparison. *Expert Systems with Applications*, 165. s. 2021.
3. Dokumentácia GraphHopper – Map matching pre GPS trajektórie, URL: <https://docs.graphhopper.com/>

SÚHRN

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Študijný program: Aplikovaná informatika

Autor: Bc. Richard Bíri

Diplomová práca: Vizualizácia pohybu používateľov

Vedúci záverečnej práce: Ing. Maroš Čavojský, PhD.

Miesto a rok predloženia práce: Bratislava 2024

Cieľom tejto práce je navrhnúť a implementovať algoritmus, ktorý bude vizualizovať väčšie datasety GPS trajektórií prehľadne. To znamená, že používateľ bude schopný rozlíšiť základné pohybové vzory zrakom z trajektórii. Výstup práce bude zobrazený interaktívne vo webovom prehliadači. Vytvoríme webovú aplikáciu, v ktorej sa používateľ autentifikuje a nahrá svoje GPS trajektórie. Po nahratí sa tieto trajektórie spracujú algoritmom a uložia sa na serverové úložisko. Používateľ bude môcť tieto upravené trajektórie zobraziť a porovnať s neupravenými trajektóriami. Trajektórie budú zobrazené v interaktívnej mape, ktorá sa bude dať priblížiť a oddialiť. Práca sa skladá zo siedmych kapitol. Prvá kapitola sa venuje problematike zobrazovania GPS trajektórií. Druhá kapitola sa venuje problematike vývoja webových aplikácií. V tretej kapitole si predstavíme dostupné riešenia, ich výhody a nevýhody. V štvrtej kapitole predstavíme technológie, s ktoré sme použili pri implementovaní algoritmu a implementovaní webovej aplikácie. V piatej kapitole definujeme funkcionálne a nefunkcionálne požiadavky aplikácie a predstavíme diagram prípadov použitia aplikácie, spolu so sekvenčnymi diagramami niektorých procesov. Taktiež navrhнемe spôsob akým budeme ukladať údaje na server a ako budeme upravovať trajektórie. V šiestej kapitole sa venujeme implementácií webovej aplikácie a úpravy trajektórií. V poslednej, siedmej kapitole hodnotíme vytvorenú aplikáciu a kvalitu úpravy trás na základe odpovedí na dotazník od testérov aplikácie.

Kľúčové slová: webová aplikácia, javascript, map-match, trajektórie

ABSTRACT

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA
FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

Study Programme:	Applied Informatics
Author:	Bc. Richard Búri
Master's thesis:	Visualization of user's movement
Supervisor:	Ing. Maroš Čavojský, PhD.
Place and year of submission:	Bratislava 2024

The goal of this work is to design and implement an algorithm that will visualize larger datasets of GPS trajectories clearly. This means that the user will be able to recognize basic movement patterns by sight from the trajectory. The work output will be displayed interactively in a web browser. We will create a web application in which the user authenticates and uploads his GPS trajectories. Once uploaded, these trajectories are processed by the algorithm and saved to server storage. The user will be able to view and compare these adjusted trajectories with the unmodified trajectories. Trajectories will be displayed in an interactive map that can be zoomed in or out. The work consists of seven chapters. The first chapter deals with issues of displaying GPS trajectories. The second chapter deals with the issue of web application development. In the third chapter, we will present the available solutions, their advantages and disadvantages. In the fourth chapter, we will present the technologies we used to implement the algorithm and implement the web application. In the fifth chapter, we define the functional and non-functional requirements of the application and present a diagram of the use cases of the application, along with sequence diagrams of some processes. We will also propose a way to store data on the server and how to edit trajectories. In the sixth chapter, we deal with the implementation of the web application and editing trajectories. In the last, seventh chapter, we evaluate the created application and the quality of route modification based on the answers to the questionnaire from the application testers.

Keywords: web application, javascript, map-match, routes

Vyhľásenie autora

Richard Búri čestne vyhlasujem, že som diplomovú prácu Vizualizácia pohybu používateľov vypracoval na základe poznatkov získaných počas štúdia a informácií z dostupnej literatúry uvedenej v práci. Vedúcim mojej diplomovej práce bol Ing. Maroš Čavojský, PhD.

Bratislava, dňa 10.5.2024

Podakovanie

Chcem sa podakovať vedúcemu záverečnej práce, ktorým bol Ing. Maroš Čavojský, PhD., za odborné vedenie, rady a pripomienky, ktoré mi pomohli pri vypracovaní tejto diplomovej práce.

Obsah

Úvod	1
1 Problematika zobrazovania GPS trás	2
2 Problematika vývoja webových aplikácií	4
2.1 Webové aplikácie a ich vývoj	4
2.2 Vývojový proces v 8 krokoch[3]:	4
2.3 Výber technológií	5
3 Podobné aplikácie	7
3.1 Mapbox	7
3.2 GPS Visualizer	8
4 Použité technológie	10
4.1 Docker	10
4.2 Valhalla api	10
4.3 Geojson.io	12
4.4 Kepler.gl	13
4.5 Express (Node.js)	14
4.6 Visual Studio Code	15
4.7 MySQL	15
4.8 Leaflet	17
5 Návrh riešenia	18
5.1 Špecifikácia požiadaviek	18
5.1.1 Funkcionálne požiadavky	18
5.1.2 Nefunkcionálne požiadavky	19
5.1.3 Diagramy	20
5.2 Ukladanie údajov	22
5.3 Pripínanie trás k cestnej sieti	24
6 Implementácia	28
6.1 Pripínanie trasy k cestnej sieti	28
6.2 Webová aplikácia	31
6.2.1 Autentifikácia	31
6.2.2 Stránka s mapou	34

7	Zhodnotenie	39
7.1	Výsledok práce	39
7.2	Hodnotenie práce reálnymi používateľmi	41
	Záver	42
	Zoznam použitej literatúry	43

Zoznam obrázkov a tabuliek

Obrázok 1	Neprehľadne zobrazené trasy.	2
Obrázok 2	Upravené trasy.	3
Obrázok 3	Porovnanie Mapbox a Valhalla map-matching	8
Obrázok 4	Prostredie GPS Visualizer	9
Obrázok 5	Prostredie Geojson.io	13
Obrázok 6	Prostredie Kepler.gl	14
Obrázok 7	Prostredie Visual Studio Code	16
Obrázok 8	Diagram prípadov použitia	20
Obrázok 9	Sekvenčný diagram procesu nahrania súborov na server.	21
Obrázok 10	Sekvenčný diagram procesu zobrazenia trás na mape.	21
Obrázok 11	Sekvenčný diagram procesu zobrazenia chybových súborov . . .	22
Obrázok 12	Štruktúry priečinkov <i>uploads</i> a <i>routes</i>	23
Obrázok 13	Trasa pripnutá pomocou <i>routing</i> funkcie	25
Obrázok 14	Trasa s nedostatočne hustým počtom bodov	26
Obrázok 15	Znázornenie interpolácie bodov	27
Obrázok 16	Optimalizácia pripnutia trasy k cestnej sieti pomocou interpolácie bodov	27
Obrázok 17	Možné chybové hlášky pri registrácii pri chybe používateľa. . . .	32
Obrázok 18	Chybová hláška pri nedostupnej databáze.	33
Obrázok 19	Stránka s mapou	34
Obrázok 20	Dialógové okno zobrazené po kliknutí <i>Profile</i> tlačidla.	36
Obrázok 21	Dialógové okno zobrazené po kliknutí <i>Show Files</i> tlačidla.	36
Obrázok 22	Dialógové okno zobrazené po kliknutí na názov nahraného ZIP súboru.	36
Obrázok 23	Dialógové okno zobrazené po kliknutí <i>Upload File</i> tlačidla. . . .	37
Obrázok 24	Dialógové okno zobrazené po kliknutí <i>Error Files</i> tlačidla. . . .	38
Obrázok 25	Zobrazenie väčšieho množstva trás na rovnakom mieste	40
Obrázok 26	Zobrazenie väčšieho množstva trás upravených map-matchom na rovnakom mieste	40

Zoznam skratiek

AI	Artificial Intelligence
API	Application programming interface
CSS	Cascading Style Sheets
GPS	Global Positioning System
HTML	HyperText Markup Language
IDE	Integrated Development Environment
KB	Kilobyte
MVP	Minimum viable product
SQL	Structured Query Language

Zoznam výpisov

1	Dáta požiadavky na <i>Valhalla</i> kontajner	29
2	Odpoveď z <i>Valhalla</i> kontajnera	29
3	<i>retDict</i> premenná	31

Úvod

V dnešnej dobe je veľmi dôležité sledovanie polohy používateľov pomocou GPS. Či už sa jedná o GPS údaje zozbierané inteligentným mobilným zariadením, inteligentnými hodinkami, automobilom alebo iným zariadením. Tieto zozbierané údaje sú užitočné na niekoľko rôznych súvislostí, napríklad navigácia a mapovanie, aktivita a trasy alebo analytika a dátový výskum.

Dáta získané GPS senzormi však nie sú vždy dostatočne presné. Väčšina zariadení, ktoré snímajú GPS údaje sú nepresné, pretože nie sú určené primárne na toto využitie [1]. Trasy zobrazené takýmito nepresnými senzormi vyzerajú neprehľadne a kvôli svojej neprehľadnosti neodovzdajú takmer žiadnu informáciu.

V našej práci sa budeme zaoberať tým, ako tieto nepresné trasy upraviť a zobraziť tak, aby používateľ dokázal zistiť pohybové vzory pohľadom. Trasy bude možné zobraziť na interaktívnej mape, ktorá sa bude dať priblížiť a oddialiť. Trasy bude možné zobraziť po nahratí do aplikácie. Trasy bude možné zobraziť všetky naraz alebo jednotlivo.

Práca bude obsahovať oboznámenie sa s problematikou vývoja webových aplikácií, výber vhodného vyvojového prostredie a vhodného programovacieho jazyka, v ktorom budeme webovú aplikáciu programovať. Ďalej obsiahne nejaké už existujúce aplikácie, v ktorých analyzujeme chyby a nedostatky, ktoré sa budeme snažiť v našom riešení odstrániť. Urobíme návrh riešenia a implementujeme aplikáciu. Vo finálnej forme aplikáciu otestujeme, či funguje správne a podľa očakávaní.

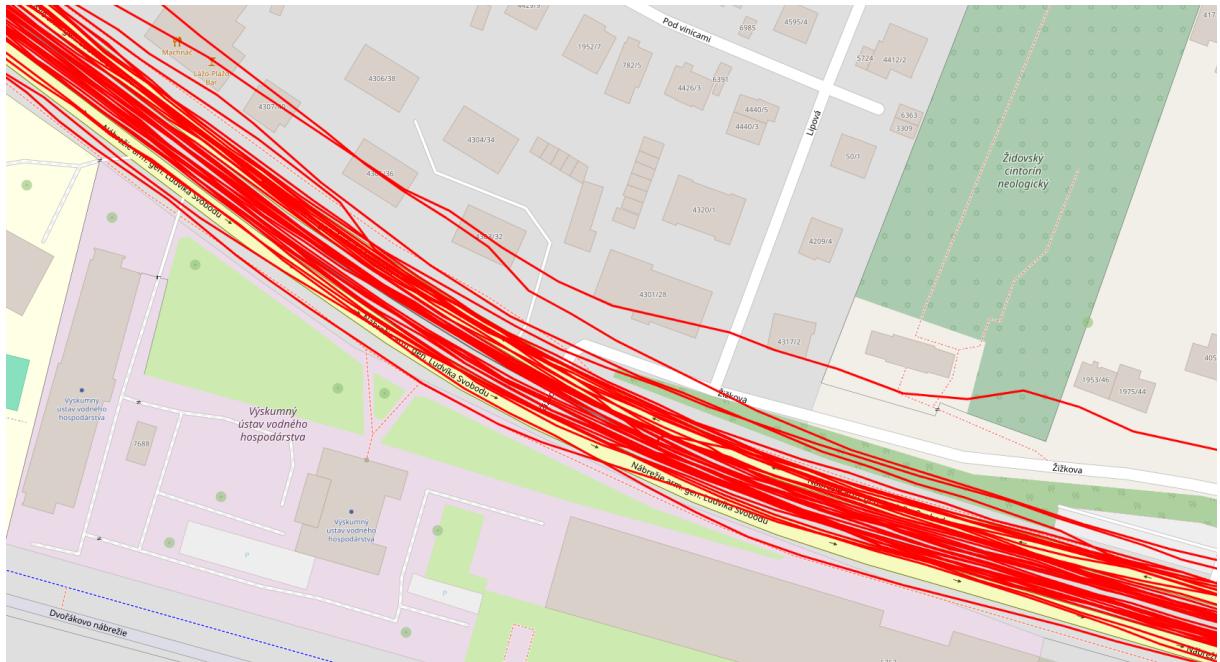
Cieľom práce je vytvoriť webovú aplikáciu, do ktorej používateľ nahrá GPS trajektóriu, ktoré budú spracované našim algoritmom a trasy pripnuté k cestej sieti. Trasy budú ležať len na cestnej sieti a tým sa zvýší celková prehľadnosť. Webová aplikácia následne zobrazí mapu s jednotlivými trasami. Zobrazené trasy budú informovať o názve nahraného súboru, názvu trasy a času nahrania.

1 Problematika zobrazovania GPS trás

Zobrazovanie GPS trás je zásadné pre množstvo aplikácií, od navigácie a mapovania až po analýzu údajov o pohybe. GPS zariadenia získavajú informácie z drúžic a používajú trianguláciu na určenie polohy. Tieto informácie sa zhromažďujú vo forme súradníc spolu s časovými údajmi[2].

Trasu získame spojením súradníc získaných z GPS zariadenia. Problém nastáva, keď sú GPS signály ovplyvnené faktormi ako sú terén, počasie alebo rôznymi prekážkami. Ovplyvnené signály môžu viest k nepresným alebo chybným údajom[2]. Spojením chybných údajov získame nepresnú trasu, ktorá leží mimo skutočnej, po ktorej sme sa reálne pohybovali.

Ked sa rovnakou trasou prechádza opakovane, údaje z GPS zariadenia môžu vykazovať variabilitu. Pri vizualizácii trás, ktoré sú zaznamenané prejdením rovnakej trasy, môže dôjsť k prekrývaniu a neprehľadnosti v dôsledku odchýlok a nepresnosti. Prekryvajúce sa trasy je možné vidieť na obrázku 1. Z dôvodu nepresnosti je zložité rozpoznať detaile mapy, ako aj cestnú sieť, čo stáhuje identifikáciu jednotlivých trás.

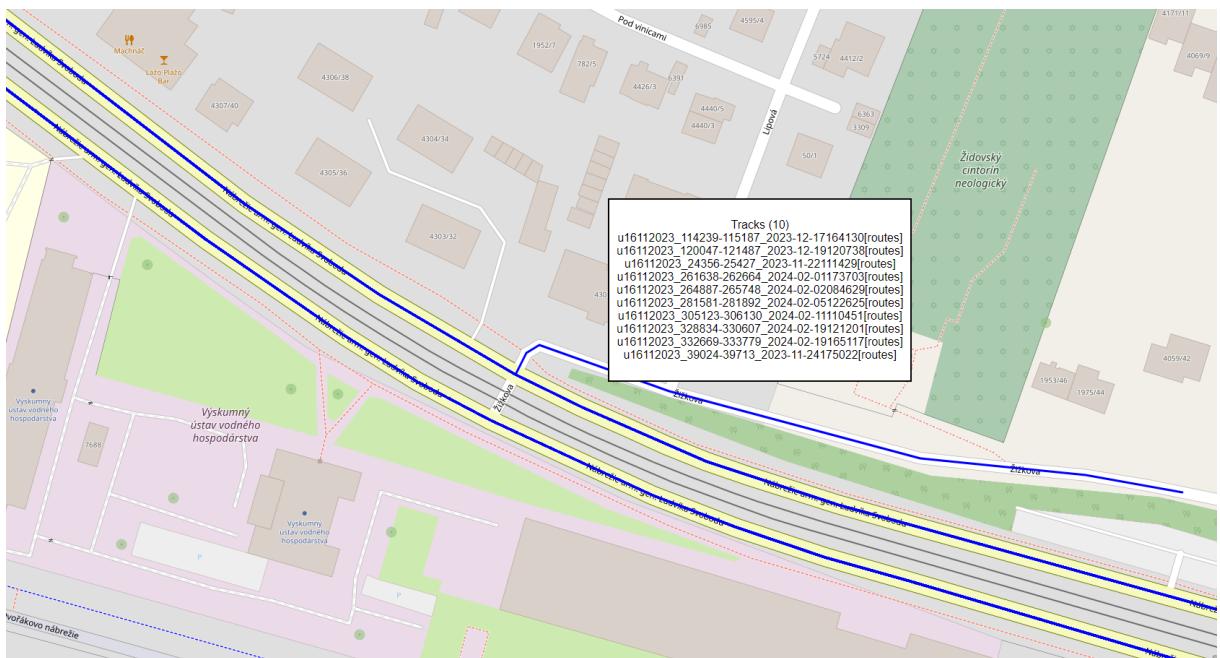


Obr. 1: Neprehľadne zobrazené trasy.

Pre zlepšenie viditeľnosti a sprehľadnenie mapy všetky trasy upravíme tak, aby sa nachádzali na cestnej sieti, po ktorej sa používateľ reálne pohyboval. Pozrieme sa na rôzne metódy, ktorými túto úpravu dosiahneme. Jednou z nich je *routing*, ktorá nájde trasu

medzi každým bodom v pôvodnej trase. Druhou metódou je *map-match*, ktorá vstupné GPS súradnice posunie k cestnej sieti. Tieto dve metódy si bližšie popíšeme v sekcií 5.3.

Vďaka týmto úpravám budú všetky trasy ležať na cestnej sieti. Všetky trasy, ktoré sme získali pri pohybe po rovnakej trase budú prekryté a cestná sieť bude viditeľná a mapa prehľadná. Po prejdení myšou po trase sa zobrazí okno, ktoré bude obsahovať názvy prekrytých trás. Taktiež v aplikácii umožníme zobraziť originálne trasy a upravené trasy, pre porovnanie rozdielu. Mapa s upravenými trasami bude vyzeráť prehľadne ako na obrázku 2. Upravené trasy a pôvodné budú farebne rozlíšené.



Obr. 2: Upravené trasy.

Trasy budú upravené a uložené, aby sa mohli neskôr rýchlejšie vykresliť na mapu. Nebudú sa upravovať pri zobrazovaní, keďže úprava dlhej trasy môže trvať až 200 ms. Zobrazenie celého nahraného ZIP súboru na mape, ktorý môže obsahovať ľubovoľný počet trás spolu s úpravou všetkých trás môže trvať minúty, čo by znepríjemnilo používateľskú skúsenosť.

2 Problematika vývoja webových aplikácií

V sekcií si predstavíme webové aplikácie a ich výhody. Vymenujeme technológie, s ktorými sa môžeme stretnúť pri vývoji a predstavíme technológie, ktoré sme zvolili na vývoj našej webovej aplikácie. Taktiež si predstavíme vývojový proces webovej aplikácie.

2.1 Webové aplikácie a ich vývoj

Webové aplikácie a ich vývoj je veľmi dôležitý v oblasti informačných technológií. Umožňuje vývojárom tvoriť a distribuovať softvér cez internet. Výhodou webových aplikácií je, že poskytujú interaktívne a dynamické prostredie pre používateľov prostredníctvom internetových prehliadačov. Pri vývoji webových aplikácií je možné sa stretnúť s mnoho technológiami a programovacími jazykmi[3]:

- **Frontend** Predstavuje prezentačnú vrstvu, ktorú vidí používateľ. Je tvorená pomocou *HTML*, ktorý tvorí obsah webovej stránky a *CSS*, ktoré upravujú vzhľad obsahu na stránke. Pre zjednodušenie vývoja a zlepšenie efektivity je možné použiť rôzne knižnice a frameworky, napríklad *React*[4] alebo *Tailwind*[5]. Pre vytvorenie dynamickej a interaktívnej webovej stránky je možné použiť *Javascript*[6].
- **Backend** Jazyk, ktorý je použitý na serverovú časť webovej aplikácie, ako napríklad prihlásenie, prácu s databázou a inej logiky na pozadí, ktorú používateľ nevidí. Patria sem napríklad: *Javascript*[6], *Python*[7], *PHP*[8], *Java*[9] a ďalšie.
- **Databázy** V mnoho aplikáciach treba dátá ukladať a tieto dátá sa ukladajú v databázach. Uložené dátá je možné prehľadávať alebo inak s nimi manipulovať. Medzi rôzne databázy patria napríklad *MySQL*[10], *PostgreSQL*[11], *MongoDB*[12] a ďalšie.
- **Knižnice a frameworky** Používajú sa pre zjednodušenie vývoja a zlepšenie efektivity. Patria sem napríklad *Angular*[13], *React*[4], *Vue.js*[14] pre frontend alebo *Django*[15], *Laravel*[16], *Express*[17], *Flask*[18] pre backend.

2.2 Vývojový proces v 8 krokoch[3]:

Nižšie sa nachádza osem krokov. Splnením týchto ôsmich krokov získame webovú aplikáciu.

1. **Prieskum trhu:** V prvom rade treba zistiť, aký problém používateľ má. Treba naštudovať existujúce riešenia problému. Pri existujúcich riešeniach treba zvážiť

výhody a nevýhody riešení. Vo vlastnej webovej aplikácii sa budeme snažiť nevýhody odstrániť.

2. **Definovanie funkcionality:** Aplikácia musí mať určené, aké funkcionality bude ponúkať. Tieto funkcionality majú riešiť problém používateľa. Chceme dosiahnuť, aby naša aplikácia riešila všetky problémy používateľa a aby nemala zbytočnú funkciu, ktorú používateľ nebude používať a bude nevyužitá. Touto zbytočnou funkciu sa proces vývoja zbytočne predĺži.
3. **MVP:** Vytvorenie základnej verzie aplikácie, kde sa nedáva špeciálny dôraz na dizajn. Je to návrh ako by mala aplikácia fungovať s implementáciou hlavných funkcií.
4. **Databáza:** Vytvorenie a pripojenie aplikácie na databázu, kde sa budú dátá bezpečne ukladať pre neskôršie načítanie a používanie v aplikácii.
5. **Frontend:** V tomto kroku vytvoríme dizajn aplikácie. S týmto dizajnom budú používatelia pracovať, preto by mal byť prehľadný, pekný a zaujímavý.
6. **Backend:** Vytvorenie serverovej časti. Patrí sem komunikácia s databázou, výpočty na pozadí, ktoré používateľ nevidí.
7. **Testovanie:** Testovanie aplikácie, či funguje správne. Odskúšame funkciu aplikácie a odhalíme prípadné chyby a zistíme, či sa aplikácia správa podľa očakávaní.
8. **Nasadenie:** Nahranie aplikácie na server, aby ju mohol používať používateľ odkialkoľvek.

2.3 Výber technológií

V sekúri si predstavíme technológie, s ktorými budeme pracovať a v krátkosti ich opíšeme.

Javascript[6]

Javascript je jedným z najpopulárnejších a najuniverzálnejších programovacích jazykov pre vytváranie webových aplikácií. Umožňuje dynamické a interaktívne funkcie, ktoré zlepšujú používateľskú skúsenosť s aplikáciou a funkčnosť webových stránok. Môže bežať na strane klienta aj na strane servera. Je to interpretovaný jazyk. Má bohatý ekosystém knižníc a frameworkov, ktoré poskytujú rôzne nástroje a abstrakcie pre vývoj webových aplikácií. Jedným z týchto frameworkov je aj Express, ktorý sme použili na vytvorenie serverovej časti. *Express* bližšie opíšeme v sekcií 4.5.

Visual Studio Code

Spomedzi viacerých vývojových prostredí a textových editorov, v ktorých je možné vytvárať webové aplikácie sme si vybrali *Visual Studio Code*. *Visual Studio Code* je textový editor, ktorý podporuje množstvo rozšírení a programovacích jazykov. Prináša mnoho výhod, ktoré sú opísané nižšie v sekciu 4.6.

MySQL

Aplikačné údaje, napríklad údaje o registrovaných používateľoch, informáciu o prihlásení a podobne ukladáme do aplikačných databáz. Tieto databázy umožňujú štrukturalizáciu dát do tabuľiek, čo pomáha efektívnej správe informácií. *MySQL* je najpopulárnejšia open-source databáza[10]. Použili sme ju na uloženie informácií o používateľoch a na uloženie informácie pre správcu aplikácie v prípade zlého pripnutiu trasy k cestnej sieti. Viac o *MySQL* je popísané v sekciu 4.7.

Valhalla

Existuje viacero služieb, ktoré pripínajú trasy k cestnej sieti, napríklad MapBox, Google Maps alebo Valhalla. Valhalla sme si vybrali, pretože je open-source, dokáže pracovať s veľkým množstvom dát a nastavenia pripínania trasy k cestnej siete sú dostatočne nastaviteľné. To znamená, že vieme meniť parametre ako napríklad presnosť GPS zariadenia, dĺžka polomera okruhu, v ktorom sa od bodu hľadajú kandidáti na pripnutie alebo penalizácia odbočovania na cestnej sieti. Všetky tieto parametre pomáhajú k lepšiemu pripnutiu trasy k cestnej sieti. Viac o *Valhalla* si v sekciu 4.2.

Leaflet

Aby sme mohli na mape zobraziť trasy, potrebujeme najprv v našej aplikácii zobraziť mapu. Na toto sme zvolili Leaflet. Leaflet je open-source knižnica napísaná v *Javascripte*. Ponúka vývojárom silnú a flexibilnú platformu na vytváranie interaktívnych máp pre webové aplikácie. Je jednoduchá na použitie a pre našu webovú aplikáciu ideálna. Viac o *Leaflet* v sekciu 4.8.

3 Podobné aplikácie

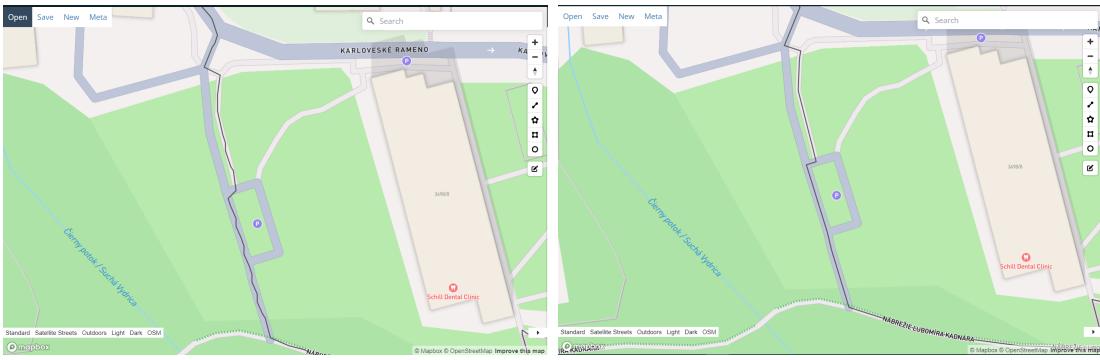
3.1 Mapbox

Mapbox je platforma údajov o polohe pre mobilné a webové aplikácie[19]. Poskytuje mnoho produktov: *Mapbox GL JS* knižnicu, pomocou ktorej je možné vytvoriť interaktívnu, prispôsobenú a efektívnu mapu vo webovej aplikácii[20], *Navigation SDK for mobiles*, riešenie pre vytváranie navigácie v mobilných telefónoch dostupné pre *Android* aj *iOS*[21], *MapGPT*, prvého AI asistenta, s ktorým je možné mať konverzácie ohľadom ciest, navigačných inštrukcií alebo atrakcií[22]. Výhody Mapboxu:

- Ponúka 50 tisíc načítaní mapy vo webovej aplikácii.
- Poskytuje map-matching, ktorý pripne vstupné GPS údaje trasy k cestnej sieti pre zaručenie prehľadného zobrazenie trás, to znamená, že trasy nebudú zobrazené mimo cesty ale budú ležať na cestnej sieti.
- Prispôsobenie štýlov mapy, markerov alebo vyskakovacích okien.

Nevýhody

- Neposkytuje vlastnú webovú aplikáciu, na ktorú je možné nahrať trasu, ktorú chceme zobraziť. Trasa sa dá zobraziť až po použití Mapbox GL JS a implementovaní vlastnej webovej aplikácie.
- Vstupné dátá treba upraviť na požadovaný formát pred odoslaním do Mapbox API.
- Do jednej požiadavky na map-match je možné vložiť len 100 bodov, preto je potrebné pre dlhšie trasy (tisíc bodov a viac) urobiť niekoľko požiadaviek. Pri aplikácií, ktorú používa veľa používateľov (rádovo tisíc) môže rýchlo dôjsť k vyčerpaniu kvóty pre map-match (100 tisíc mesačne) zadarmo. Taktiež rozdelenie trás do viacerých požiadaviek spomaluje rýchlosť map-matchingu.
- Trasa je ku cestnej sieti pripnutá nepresne, viď obrázok 3.
- Je potrebné použiť online API, čo môže spomalovať map-matching.



(a) Map-match od Mapbox

(b) Map-match od Valhally

Obr. 3: Porovnanie Mapbox a Valhalla map-matching

3.2 GPS Visualizer

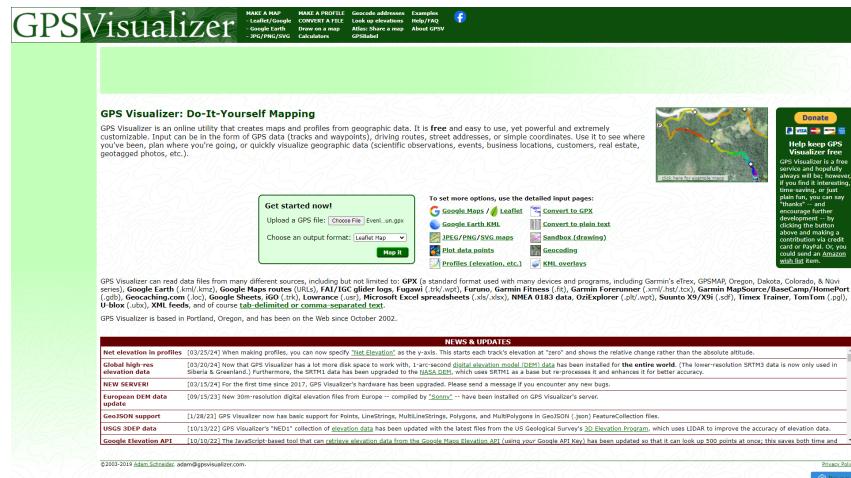
GPS Visualizer je online nástroj, ktorý vytvára mapy a profily z geografických údajov. Ľahko sa používa, je prehľadný a rýchly, to znamená, že pri prvom pohľade na nástroj používateľ vie, ako nástroj používať. Vstup môže byť vo forme údajov GPS (trasy a body na ceste), jazdných trás, ulíc alebo jednoduchých súradníc. Používa sa na zobrazenie trás, plánovanie trasy alebo na rýchlu vizualizáciu geografických údajov (na vedecké pozorovania, zobrazenie nehnuteľností, geograficky označených fotografií a podobne). Na webe je od októbra 2002[23]. Výhody:

- Rýchly, prehľadný a ľahký na použitie.
- Dokáže načítať dátá z rôznych zdrojov (GPX, URL, FIT, CSV, TRK a ďalšie).
- Mapu so zobrazenou trasou je možné stiahnuť vo formáte HTML stránky na neskôršie zobrazenie.

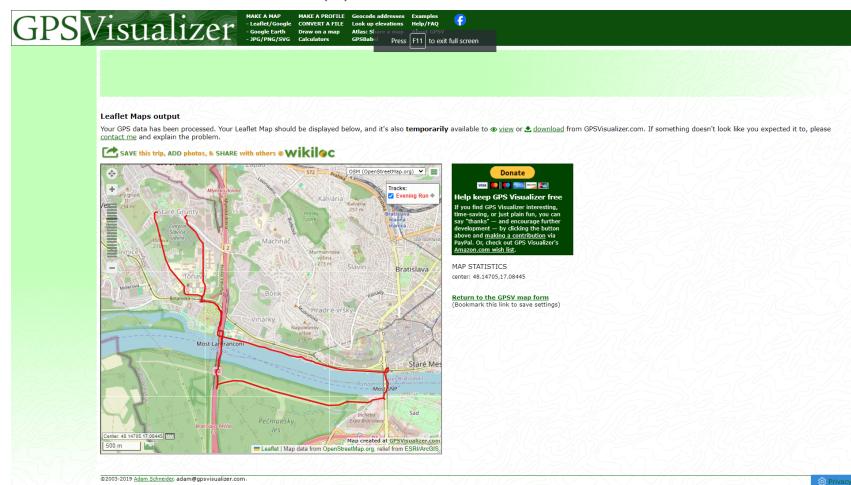
Nevýhody:

- Neponúka map-matching, iba zobrazuje trasy.
- Pre zobrazenie trasy je nutné zakaždým nahrať súbor s GPS údajmi.
- Nutnosť pripojenia na internet kvôli načítaniu knižníc.

Prostredie GPS visualizer je zobrazené na obrázkoch nižšie 4.



(a) Nahratie trasy



(b) Zobrazenie trasy

Obr. 4: Prostredie GPS Visualizer

4 Použité technológie

4.1 Docker

Docker je softvérová platforma, ktorá umožňuje rýchlo vytvárať, testovať a nasadzovať aplikácie. Docker balí softvér do štandardizovaných jednotiek nazývaných kontajnery, ktoré majú všetko, čo softvér potrebuje na spustenie vrátane knižníc, systémových nástrojov, kódu a runtime. Pomocou Dockeru môžete rýchlo nasadiť a škálovať aplikácie do akéhokoľvek prostredia a vediet, že váš kód sa spustí.[24]. Docker sa využíva z rôznych dôvodov[25]:

- **Spravovanie závislostí:** Docker umožňuje špecifikovať a spravovať všetky závislosti aplikácie v súvisiacom kontajneri, čo uľahčuje riadenie verzií a aktualizácií.
- **Rýchlosť nasadzovania:** Kontajnery sa rýchlo spúšťajú a zastavujú, čo zjednodušuje vývoj a nasadzovanie aplikácií.
- **Izolácia:** Kontajnery izolujú aplikáciu a jej závislosti od zvyšku systému, čo zabezpečuje konzistentné a predvídateľné spustenie aplikácie bez ohľadu na prostredie.
- **Portabilita:** Kontajnery sú prenosné medzi rôznymi prostrediami, čo znamená, že sú spustiteľné na rôznych serveroch, vývojových počítačoch a v cloude bez problémov.
- **Škálovateľnosť:** Docker kontajnery môžu byť ľahko škálovateľné pridávaním viacerých inštancií alebo zvyšovaním výkonu existujúcich inštancií.

4.2 Valhalla api

Valhalla je open source trasovací engine so sprievodnými knižnicami na použitie s dátami OpenStreetMap. Valhalla tiež obsahuje nástroje ako výpočet matice času a vzdialenosť, izochróny, vzorkovanie nadmorskej výšky, porovnávanie máp a optimalizácia trás (Travelling Salesman)[26].

Odin (generovanie trasy, rozpoznávanie manévrov)

Odin slúži ako nástroj inštrukcii na anotáciu cesty ako vstup zo smerovacieho enginu na použitie v navigácii. V súlade so severskou mytologickou témou bolo zvolené meno Odin, pretože sa o ňom často hovorilo, že je velmi múdry. Keďže knižnica sa zaoberá väčšinou poskytovaním rozumného navádzania po ceste, ktoré sa má použiť na navigáciu, zdalo sa to ako vhodné meno. Odin obsahuje súbor rôznych dátových štruktúr a algoritmov, ktoré sa zaoberajú vecami ako: generovanie manévrov, porovnávanie názvov ulíc a generovanie

rozprávania. [27]

Thor (trasovacie algoritmy)

Thor slúži ako trasovací mechanizmus podporovaný trasovacími dátami s otvoreným zdrojom. Thor je spoločníkom Sif, na ktorý sa veľmi spolieha pri určovaní vhodného prechodu grafom. Výsledná cesta môže byť použitá ako vstup pre vytvorenie manévr. meno Thor bolo zvolené ako skratka pre: *Tiled Hierarchical Open Routing* a bola základnou myšlienkou, okolo ktorej sa sformovala organizácia Valhalla a jej téma severskej mytológie.

Knižnica Thor je v podstate súbor rôznych dátových štruktúr a algoritmov, ktoré sa zaoberajú vecami ako: prechod grafom A*, kalkulácia hrán, kalkulácia vrcholov a konštrukcia cesty. Zahŕňa tiež metódy na výpočet matíc časových vzdialenosí, optimalizované trasovanie a izochróny.

Sif (cena pre hrany a prechody)

Sif poskytuje dynamickú, rozšíriteľnú kalkuláciu pre hrany a prechody medzi hranami (náklady na odbočenie). Jeho primárne využitie je v trasovacom engine pri vytváraní najlepšej cesty medzi miestami. V súlade so severskou mytologickou térou bolo zvolené meno Sif, pretože Sif je spoločníkom Thora.

Sif je v podstate súbor rôznych dátových štruktúr a algoritmov, ktoré sa zaoberajú vecami ako: korelácia vstupného miesta so základným grafom, čiastočná vzdialenosť pozdĺž hrany a filtrovanie hrán, ktoré by sa nemali brať do úvahy pri korelácii.

Loki (priradenie lokácie grafovým vrcholom)

Loki sa používa na priradenie informácií o polohe k základnému objektu grafu na použitie pri vytváraní vstupu do trasovacieho enginu. V súlade so severskou mytologickou térou bolo meno Loki vybrané ako slovná hračka nájsť. Kedže Loki sa zaoberá väčšinou koreláciou nejakého vstupu (minimálne zemepisnej šírky, dĺžky) k objektu v rámci grafu, zdalo sa to ako vhodný názov!

Loki je v podstate súbor rôznych dátových štruktúr a algoritmov, ktoré sa zaoberajú vecami ako: korelácia vstupného miesta so základným grafom, čiastočná vzdialenosť pozdĺž okraja a filtrovanie hrán, ktoré by sa nemali brati do úvahy pri korelácii.

Mjolnir (trasovací graf a generovanie dlaždíc)

Mjolnir je v súbor aplikácií, dátových štruktúr a algoritmov, ktoré sa zaoberajú vecami ako: spracovanie dátových extraktov OSM, rezanie smerovateľných „grafových“

dlaždíc, generovanie hierarchií dlaždíc a testovanie na nedostatky údajov. Tieto dlaždicové smerovacie údaje sa používajú pri smerovaní a vyhľadávaní v rámci organizácie Valhalla. V súlade so severskou mytologickou tému bol zvolený názov Mjölnir, pretože predstavuje zbraň hromadného ničenia. Zdalo sa to vhodné, pretože hlavná aplikácia sa zaobrá väčšinou rozbíjaním údajov OSM veľkosti planéty do malých fragmentov smerovateľných dlaždíc.

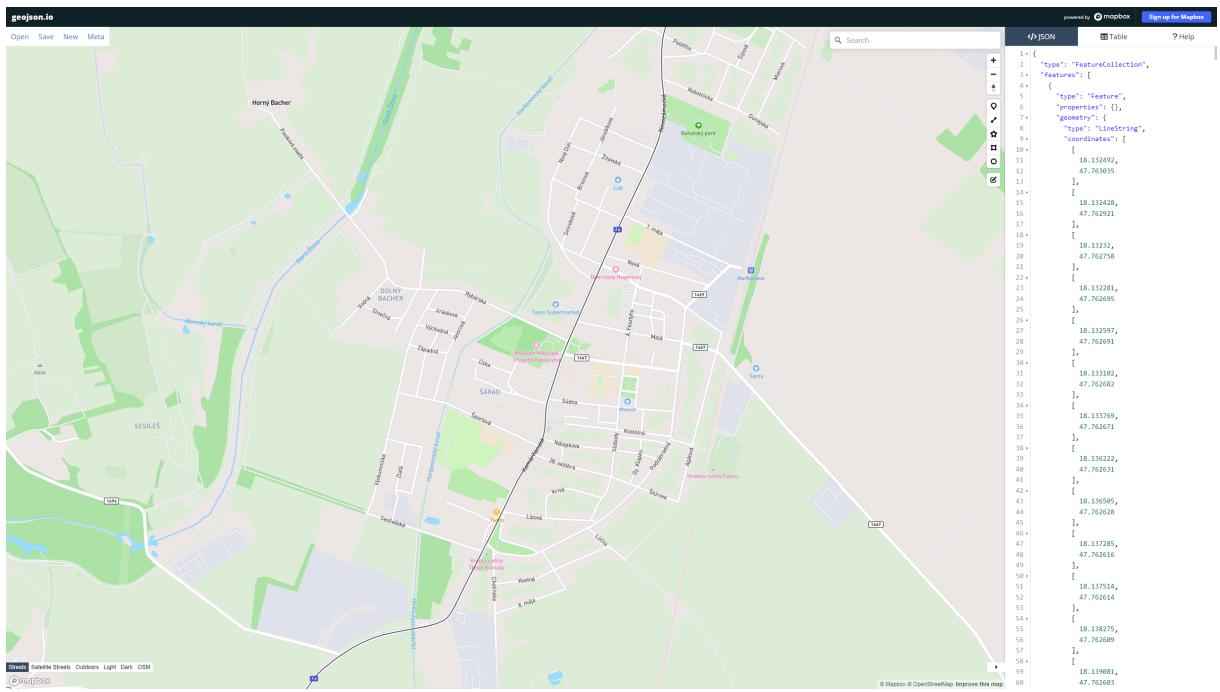
Meili (map matching)

Meili, poskytuje sadu algoritmov a dátových štruktúr na porovnávanie máp. Priraduje sekvenčiu miest (zvyčajne nepresných, napr. trajektória GPS) k základnej cestej sieti. V súlade so severskou mytologickou tematikou bolo zvolené meno Meili, Thorov brat. Kedže porovnávanie máp úzko súvisí so smerovaním a kedže Thor je trasovacia knižnica Valhally, Meili sa zdalo najvhodnejšie.

4.3 Geojson.io

GeoJSON.io je webový nástroj, ktorý umožňuje používateľom vytvárať, upravovať a vizualizovať údaje GeoJSON. GeoJSON je formát na kódovanie geografických dátových štruktúr, ako sú body, čiary a polygóny, pomocou JavaScript Object Notation (JSON). Bežne sa používa na reprezentáciu priestorových informácií a je podporovaný mnohými nástrojmi mapovania a GIS (Geografický informačný systém). Hlavnými výhodami sú:

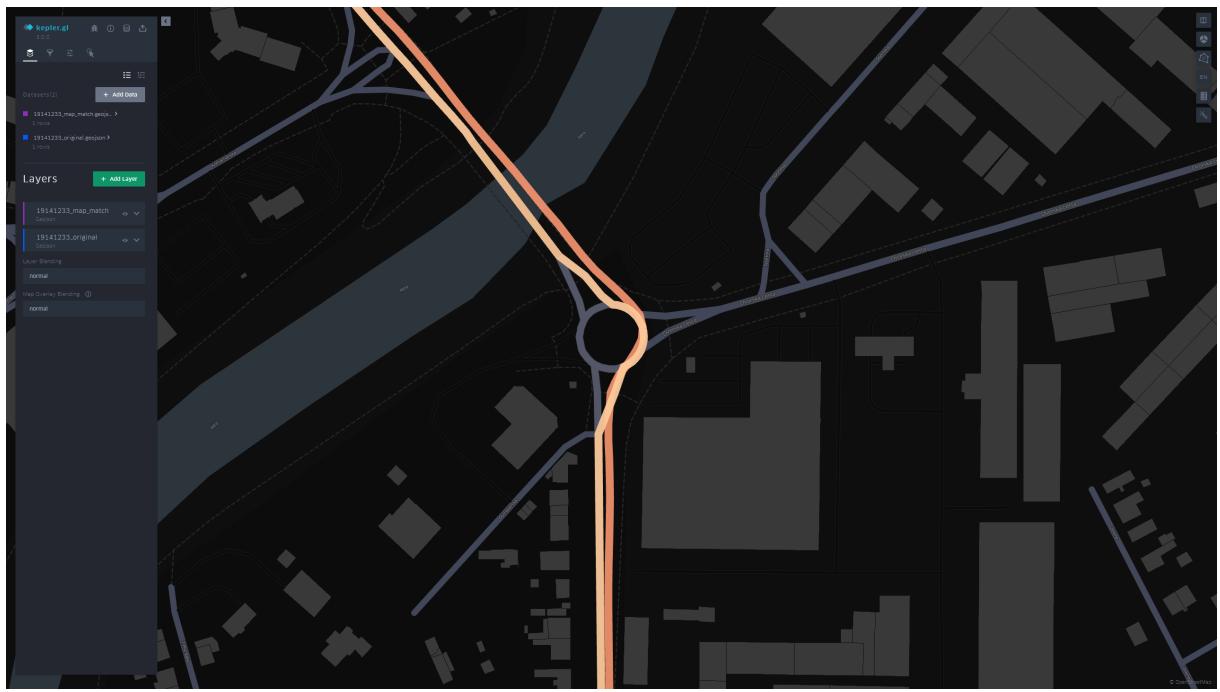
- **Interaktívne rozhranie:** Nástroj poskytuje interaktívnu mapu, na ktorej si používateľ dokáže nakresliť a upraviť GeoJSON.
- **Import/Export dát:** Nástroj umožňuje jednoducho nahrať a zobrazit GeoJSON, alebo uložiť GeoJSON vytvorený v nástroji.
- **Dizajn:** Jednoduchý a intuitívny dizajn pre rýchle použitie bez inštalácie.



Obr. 5: Prostredie Geojson.io

4.4 Kepler.gl

Kepler.gl je open source nástroj na geopriestorovú analýzu vyvinutý spoločnosťou Uber. Ide o webovú aplikáciu, ktorá používateľom umožňuje vizualizovať a skúmať rozsiahle súbory geolokačných údajov. Kepler.gl je údajovo agnostický a dokáže vykresliť milióny bodov reprezentujúcich tisíce ciest, čo z neho robí výkonný nástroj na priestorovú analýzu a vizualizáciu. Je postavený na platforme deck.gl WebGL na vizualizáciu údajov a možno ho okamžite použiť z webového prehliadača bez toho, aby ste museli čokoľvek inštalovať. Kepler.gl je v Uberi široko používaný na podporu pokročilých geopriestorových analýz inžiniermi, analytikmi a dátovými vedcami[28][29][30].



Obr. 6: Prostredie Kepler.gl

4.5 Express (Node.js)

Express je backend aplikačný framework na vytváranie serverovej časti webových aplikácií. Je vydaný ako bezplatný softvér s licenciou MIT. Dá sa nazvať štandardným serverovým frameworkom pre Node.js[17]. Jeho prvá verzia, aspoň podľa histórie Githubu, bola dňa 22. Mája 2010. Prvá vydaná verzia bola 16. Novembra 2010[31]. Express je napísaný v Javascripte a beží na Node.js platforme. Výhody Express[32]:

- **Jednoduchý na použitie:** Express je účinný a užívateľsky prívetivá platforma. Má intuitívny API a jednoduchú štruktúru. Umožňuje zostaviť aplikáciu len pomocou párov riadkov kódu, ktoré majú priamu a minimalistickú syntax.
- **Veľká komunita:** Mnoho vývojárov používa tento framework, takže je veľmi populárny a existuje množstvo návodov, pomôcok a fór, kde sa dá obrátiť pri probléme pri vývoji. Taktiež má množstvo rozšírení, middleware a ďalších nástrojov, ktoré môžu byť integrované do aplikácie.
- **Správanie HTTP požiadaviek:** Vďaka jednoduchému a flexibilnému spôsobu definovanie ciest pre rôzne HTTP požiadavky môžeme ľahko určiť ako bude naša aplikácia reagovať a pracovať s dátami podľa našich potrieb.
- **Škálovateľnosť:** Express je ľahko škálovateľný, takže pri vhodnom návrhu aplikácie

môžeme efektívne spracovať aj veľké množstvo požiadaviek.

Nevýhody Express[32]:

- **Bezpečnosť:** Bolo ohlásených mnoho bezpečnostných problémov, ako napríklad vzdialené spustenie kódu pomocou *ejs* modulu alebo častný *denial of service* pri čerpstvom module.
- **Aktualizácie:** Kedže sa tento framework stále vyvíja a zlepšuje, môže nastať, že niektoré moduly už nebudú podporované alebo sa budú používať inak ako sa očakáva. Preto je potrebné kontrolovať aktualizácie a zmeny a uistíť sa, že tieto aktualizácie neprinesú do našej aplikácie nové chyby.

Použitie Express v populárnych aplikáciách: Uber, Netflix, PayPal a LinkedIn [32].

4.6 Visual Studio Code

Visual Studio Code je jednoduchý, ale výkonný editor zdrojového kódu, ktorý je stiahnutelný pre Windows, MacOS aj Linux. Dodáva sa so vstavanou podporou pre JavaScript, TypeScript a Node.js [33]. Má bohatý ekosystém rozšírení pre ďalšie jazyky. Taktiež obsahuje množstvo pluginov a rozšírení. Ponúka množstvo vývojárskych nástrojov ako integrácia gitu, debugger alebo vstavaný príkazový terminál. Bol vydaný 29. Apríla 2015 [34]. Jeho hlavnými výhodami sú [35]:

- Je zadarmo.
- Dostupný pre Windows, MacOS aj Linux.
- Je rozšíriteľný a je možné nainštalovať rozšírenia pre takmer všetky populárne programovacie jazyky. Nie je potrebné stahovať zvlášť *IDE* pre každý programovací jazyk, s ktorým pracujete.
- Pracuje rýchlejšie aj veľkým množstvom pluginov ako ostatné *IDE*.
- Je bohatý na funkcie a s vhodnými nainštalovanými rozšíreniami je porovnatelný s plnohodnotnými IDE aj napriek tomu, že ide stále len o textový editor.

4.7 MySQL

Všetky softvérové aplikácie vyžadujú ako primárny zdroj údajov databázy. Databáza ukladá údaje, keď niekto vykoná online vyhľadávanie, prihlási sa do účtu alebo dokončí

```

File Edit Selection View Go Run Terminal Help
DPI
EXPLORER
  -> DFI
    -> webapp
      -> public
        -> db.sql
      -> index.js
        -> map_match.py
        -> map.js
        -> package.json
        -> upload.js
        -> .gitignore
        -> Dockerfile
        -> README.md
      -> OUTLINE
      -> TIMELINE
index.js
1 webapp > index.js > (app.post('/register') callback
2   connectToDatabase();
3
4   // Route to handle user registration
5   app.post('/register', async (req, res) => {
6     const { username, password } = req.body;
7     try {
8       const hashedPassword = bcrypt.hashSync(password, 10);
9       const sql = 'INSERT INTO users (username, password) VALUES (?, ?)';
10      connection.query(sql, [username, hashedPassword], (err, result) => {
11        if (err)
12          console.error(`Error registering user: ${err.message}`);
13        res.status(200).send(err.code);
14        return;
15      });
16    } catch (err) {
17      console.error(`Error registering user: ${err.message}`);
18      res.status(400).send(err.code);
19    }
20  });
21
22  app.post('/logout', (req, res) => {
23    req.session.username = null;
24    res.redirect('/login');
25  });
26
27  // Route to handle login
28  app.post('/login', async (req, res) => {
29    const { username, password } = req.body;
30    const sql = 'SELECT * FROM users WHERE username = ?';
31    connection.query(sql, [username], async (err, results) => {
32      if (err)
33        console.error(`Error fetching user: ${err.message}`);
34      res.status(500).send(err.code);
35      return;
36    });
37    if (results.length === 0) {
38      res.status(401).send('Invalid username or password');
39      return;
40    }
41    const user = results[0];
42    if (await bcrypt.compareSync(password, user.password)) {
43      req.session.username = user.username;
44      res.status(200).send('SUCCESS');
45    } else {
46      res.status(401).send('Invalid username or password');
47    }
48  });

```

Obr. 7: Prostredie Visual Studio Code

transakciu, aby bolo neskôr možné tieto údaje získať. Namiesto ukladania všetkých údajov do jedného veľkého súboru, uchováva relačná databáza údaje usporiadane do tabuľiek. Skutočné súbory, ktoré tvoria štruktúru databázy, sú usporiadane tak, aby maximalizovali rýchlosť. Je možné nastaviť vzťahy medzi jednotlivými údajmi 1:1, 1:N. Dátové polia môžu byť jedinečné, povinné alebo voliteľné. Databáza presadzuje tieto pravidlá, čo zaistuje, že v prípade dobre navrhnutej databázy vaša aplikácia nikdy nenarazí na nekonzistentné, duplicitné, ojedinelé, zastarané alebo chýbajúce údaje. Časť SQL v *MySQL* známená Structured Query Language. SQL je najbežnejší štandardizovaný jazyk používaný na prístup k databázam. Výhody *MySQL*[10]:

- Databáza je jednoduchá na inštaláciu, to znamená, že je vývojár nainštaluje do pár minút.
- Je ľahko škálovateľná. Zvládne operovať nad veľkým množstvom dát, rovnako ako obslužiť veľké množstvo pripojení. Je preto vhodná aj pre veľké aplikácie ako *Facebook*.
- Poskytuje mnoho bezpečnostných prkov, napríklad autentifikáciu alebo nastavenie práv pre rôznych používateľov. Vďaka tomuto sú dáta v bezpečí pred neautorizovanými používateľmi.

4.8 Leaflet

Leaflet je poprená open-source knižnica napísaná v *Javascripte* pre interaktívne mapy či už pre mobilné telefóny alebo počítače. Spolu má menej ako 50KB a poskytuje všetky funkcie mapy, ktoré väčšina vývojávor potrebuje[36]. Je jednoduchá, prehľadná a dajú sa pomocou nej jednoducho vykreslovať trasy na mapu.

Výhody[37]:

- Jednoduchosť - *Leaflet* je dizajnovaný pre jednoduché použitie. Je možné vytvoriť mapu jednoduchým skopírovaním kódu z *QuickStart* tutoriálu [38].
- Dokumentácia - Dokumentácia pre *Leaflet* je dobre štrukturovaná s mnohými príkladmi a návodmi.
- Komunita - Kedže *Leaflet* je najpopulárnejšou *Javascriptovou* knižnicou pre prácu s mapami, má obrovskú komunitu používateľov. Chýbajúce príklady zložitejšieho využitia sú doplnené mnohými príkladmi používateľov z komunity. "leaflet stackoverflow" na *Google* vyhľadávaní vráti takmer 400 tisíc výsledkov.
- Flexibilita - V ponuke sú všetky funkcie, ktoré väčšina vývojárov pri práci s mapami potrebuje. V prípade potreby je možné funkcionality doplniť pluginmi.

5 Návrh riešenia

5.1 Špecifikácia požiadaviek

V tejto sekcii sa oboznámime s požiadavkami, ktoré má naša aplikácia spĺňať. Rozdelíme ich na funkcionálne a nefunkcionálne požiadavky. Taktiež si predstavíme diagram prípadov použitia a sekvenčné diagramy niektorých procesov.

5.1.1 Funkcionálne požiadavky

Funkcionálne požiadavky určujú, čo presne je možné v aplikácii robiť, alebo akú má funkciunalitu.

Zoznam funkcionálnych požiadaviek

- **registrácia** - používateľ sa môže zaregistrovať pomocou používateľského mena a hesla
- **prihlásenie** - používateľ sa môže prihlásiť pomocou používateľského mena a hesla
- **odhlásenie** - používateľ sa môže z aplikácie odhlásiť
- **nahranie trás** - používateľ môže do aplikácie nahrať ZIP súbor obsahujúci trasy
- **zobrazenie nahraných trás** - používateľ môže zobraziť zoznam všetkých súborov, ktoré nahral a boli úspešne spracované aplikáciou
- **zobrazenie všetkých trás v nahranom súbore** - používateľ môže na mape zobraziť trasy, ktoré nahral v ZIP súbore
- **zobrazenie jednotlivých trás v nahranom súbore** - používateľ môže na mape zobraziť jednodlivo trasy, ktoré obsahuje ZIP súbor, ktorý nahral
- **pripnutie trás k cestnej sieti** - používateľ môže zobraziť trasy spracované aplikáciou, ktoré sú pripnuté k cestnej sieti. Zobraziť môže všetky trasy, ktoré obsahuje ZIP súbor, ktorý nahral, alebo jednotlivou.
- **zobrazenie chybových nahraných súborov** - v prípade, že nahraný súbor nemal správny formát alebo bola jeho štruktúra nevhodná, sa tento súbor uloží na neskôršie stiahnutie a používateľ je informovaný o chybe, ktorá nastala
- **znovu-spustenie chybných súborov** - v prípade, že sa nejaké trasy nachádzajú v priečinku chybových súborov, je možné na týchto trasách znova spustiť algoritmus pripnutia trás k cestnej sieti. Ide o špeciálny prípad použitia, keď si používateľ nahrá

na server súbory bez toho, aby ich dával do ZIP súboru. Týmto sa ušetrí čas pri komprimovaní trás do ZIP súboru a pri rozbalovaní ZIP súboru.

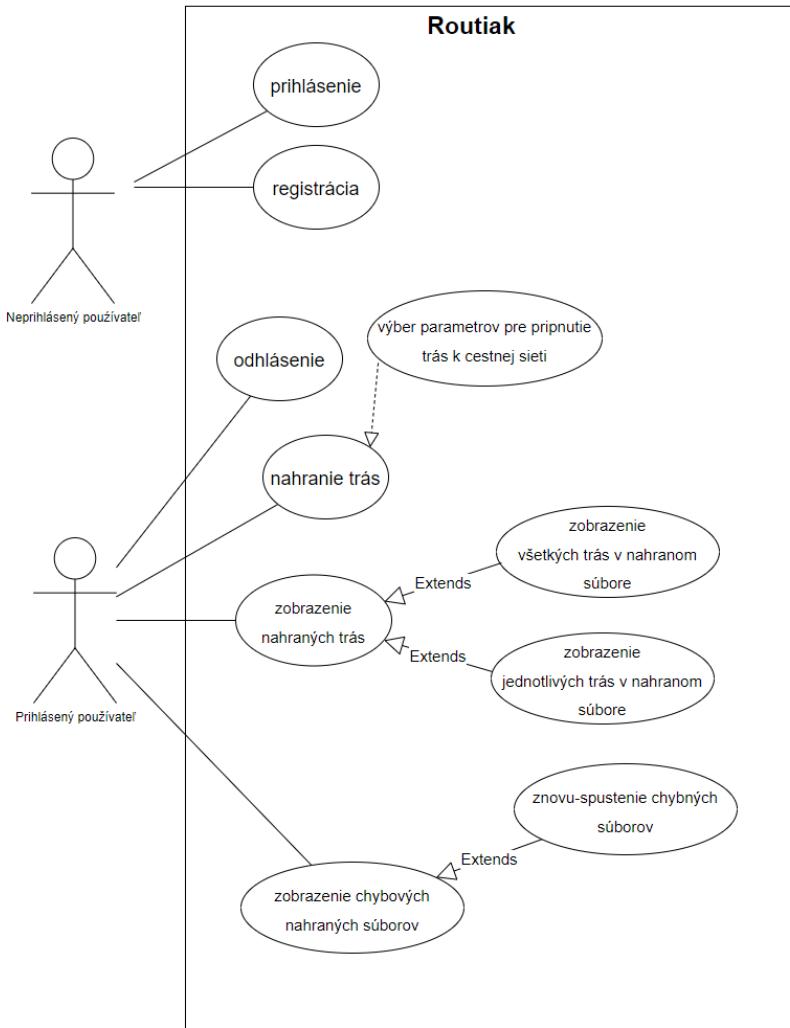
5.1.2 Nefunkcionálne požiadavky

Nefunkcionálne požiadavky sa týkajú toho, ako má systém pracovať. Popisujú parametre systému a nie to, čo má systém robiť.

Zoznam nefunkcionálnych požiadaviek

- **jednoduchosť a prehľadnosť** - aplikácia má byť jednoduchá na použitie a po prečítaní návodu na použitie má byť jasné ako má byť použitá
- **prehľadný kód** - kód má byť písaný prehľadne, aby sa v budúcnosti dal jednoducho rozšíriť a aby bol udržateľný
- **jasná informácia o chybnej trase** - ak nastane chyba pri nahraní alebo spracovaní trás, používateľ má byť jasne informovaný, kde nastala chyba

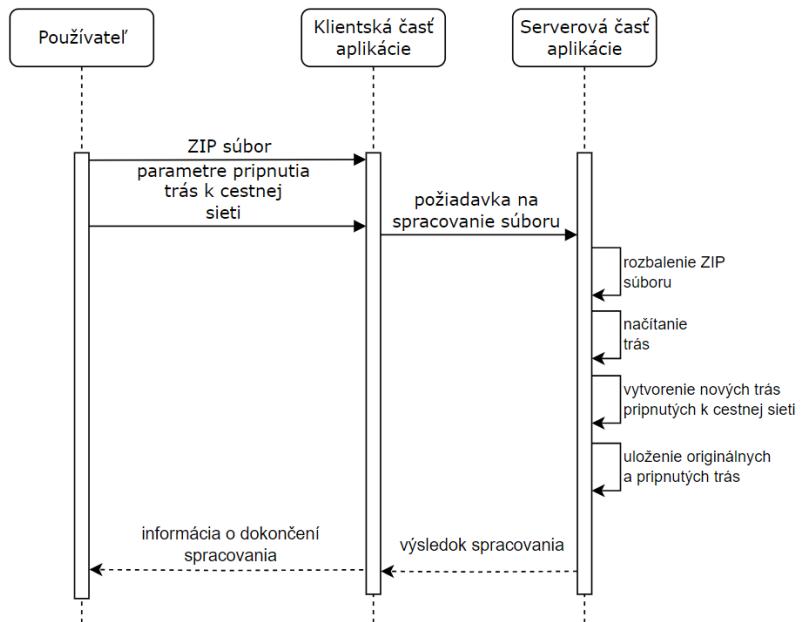
5.1.3 Diagramy



Obr. 8: Diagram prípadov použitia

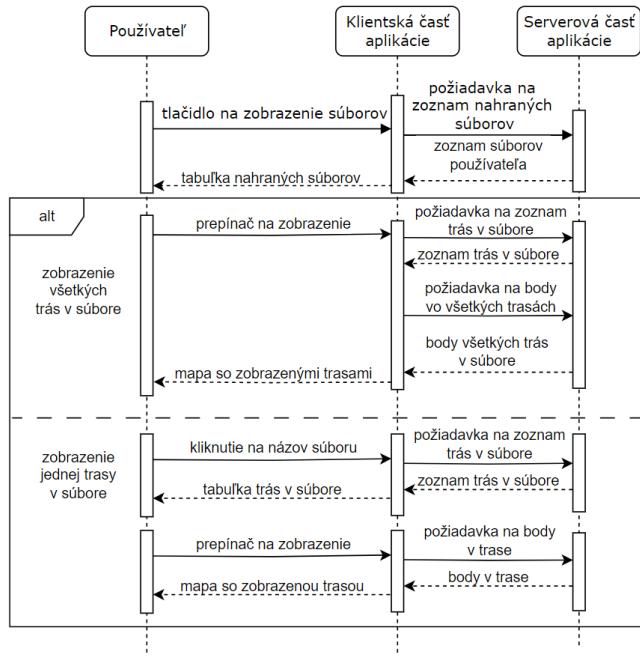
Používateľ sa bude musieť po otvorení webovej aplikácie prihlásiť alebo registrovať. Po prihlásení sa rola používateľa zmení na prihláseného. Ako prihlásený bude používateľ môcť nahrať trasy vo formáte ZIP. Pri nahrávaní súboru musí taktiež zvoliť parametre, ktoré budú použité systémom pri pripínaní trás k cestnej sieti. Prihlásený používateľ si bude môcť zobraziť trasy rozdelené podľa súborov, ktoré nahral. Trasy v týchto súboroch bude potom môcť zobraziť všetky naraz, alebo jednotlivo. Taktiež si bude môcť zobraziť chybové nahrané súbory v prípade, že nejaké chybové súbory nahral. Na týchto chybových súboroch bude potom môcť spustiť algoritmus pripínania trás k cestnej sieti znova. Nižšie je možné vidieť priebeh procesov nahrania súborov (obr.9), zobrazenia trás (obr.10) alebo zobrazenia chybových súborov (obr. 11) pomocou sekvenčných diagramov.

Nahranie súborov

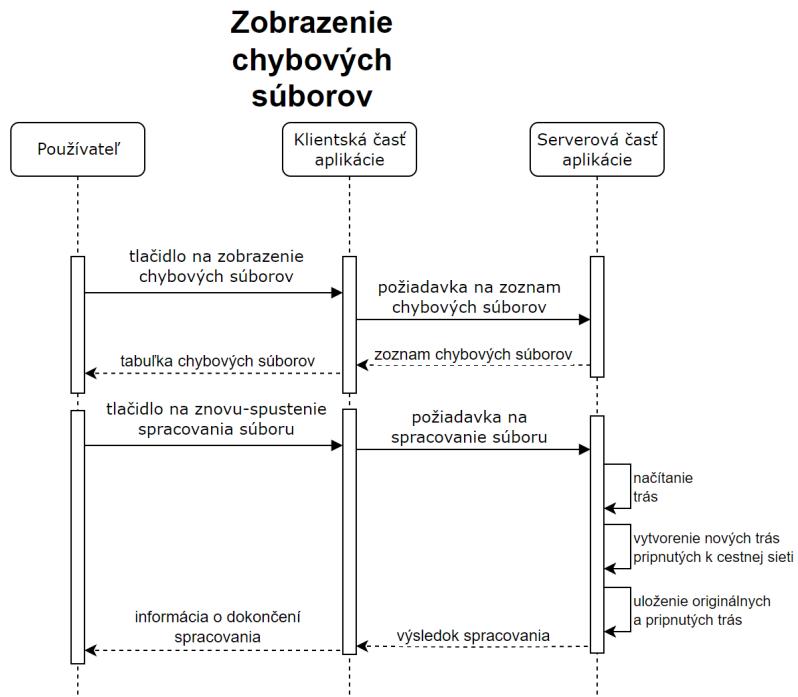


Obr. 9: Sekvenčný diagram procesu nahrania súborov na server.

Zobrazenie trás



Obr. 10: Sekvenčný diagram procesu zobrazenia trás na mape.



Obr. 11: Sekvenčný diagram procesu zobrazenia chybových súborov

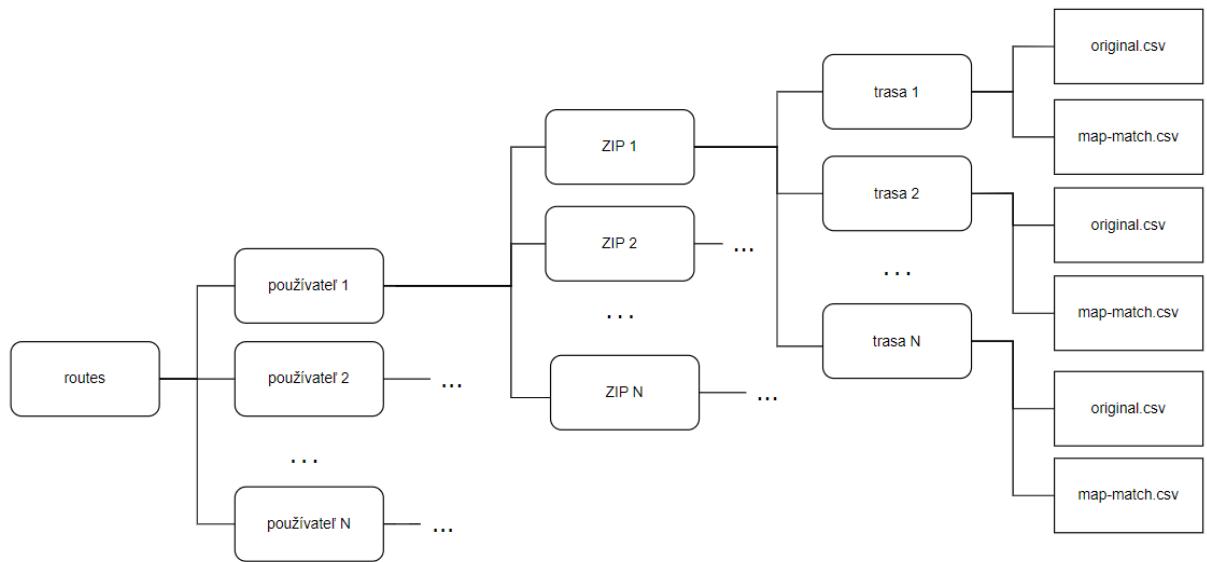
5.2 Ukladanie údajov

V aplikácii sú vytvorené priečinky *uploads* a *routes*. Do priečinka *uploads* sa budú ukladať všetky nahrané súbory. Po nahratí ZIP súboru na server bude pre používateľa vytvorený priečinok, ktorého názov bude zodpovedať prihlásovaciemu menu používateľa. V priečinku používateľa sa vytvorí nový priečinok *unzipped*, v ktorom sa vytvorí priečinok s názvom nahraného ZIP súboru. Do tohto priečinka sa rozbalí nahraný ZIP súbor. Cesta k tomuto priečinku sa uloží do premennej, s ktorou algoritmus neskôr bude pracovať. Po skončení algoritmu prebehne kontrola, ktorá určí či boli všetky trasy v ZIP súbore spracované správne. V prípade, že v ZIP súbore bude nejaká chybová trasa, ZIP súbor spolu s rozbalenými súbormi ostane na serveri pre neskôršie stiahnutie používateľom alebo prípadné znova-spustenie algoritmu. V opačnom prípade sa priečinok aj s nahraným ZIP súborom vymaže, aby sa ušetrilo miesto na serveri.

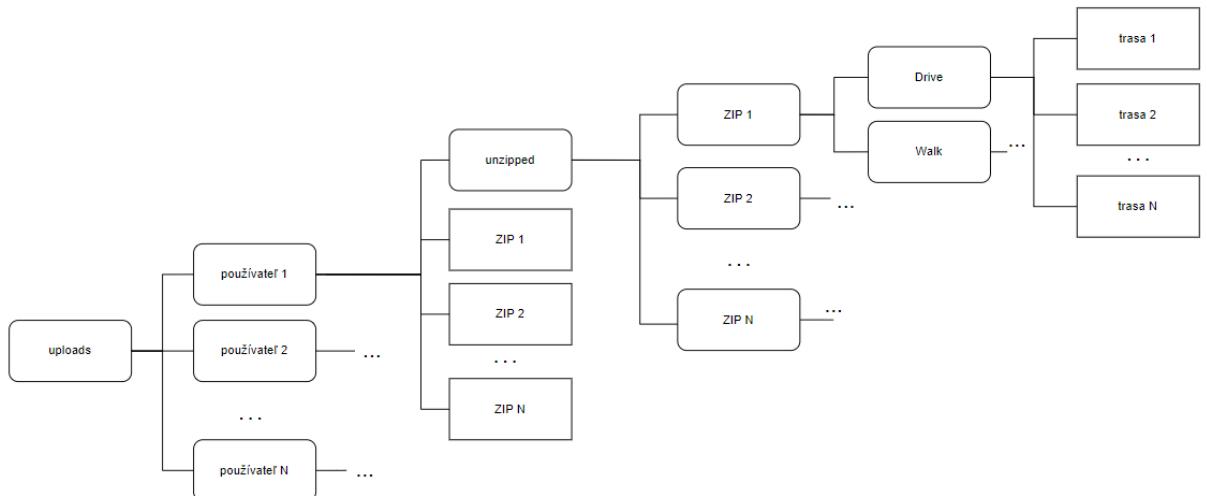
Štruktúra priečinka *routes* bude podobná priečinku *uploads*. Do priečinka *routes* sa po dokončení algoritmu vytvorí priečinok (ak ešte neexistuje), ktorého názov bude zodpovedať prihlásovaciemu menu používateľa. V priečinku používateľa bude vytvorený priečinok s názvom zodpovedajúcim názvu nahraného ZIP súboru. Do priečinku s názvom súboru budú vytvorené priečinky, ktorých názov bude zodpovedať názvom trás, ktoré boli nahrané v ZIP súbore a ktoré boli úspešne spracované algoritmom pripínania trás k cestnej sieti. V

každom takomto priečinku budú dva súbory s názvom *original.csv* a *map-match.csv*, ktoré budú obsahovať konkrétnie body trasy.

Štruktúry priečinkov sú vizualizované na obrázkoch . Priečinky sú zobrazené zaobleným obdĺžnikom a súbory sú zobrazené obdĺžnikom. Tri bodky "... " vertikálne medzi priečinkami a súbormi symbolizujú, že v priečinku môže byť 0 až N súborov. To znamená, že môže byť priečinok prázdny, alebo v ňom môže byť ľubovoľný počet súborov/priečinkov. Tri bodky "... " horizontálne za priečinkom symbolizujú, že graf pokračuje ale pre zjednodušenie obrázka bol graf skrátený.



(a) Štruktúra priečinku *routes*



(b) Štruktúra priečinku *uploads*

Obr. 12: Štruktúry priečinkov *uploads* a *routes*.

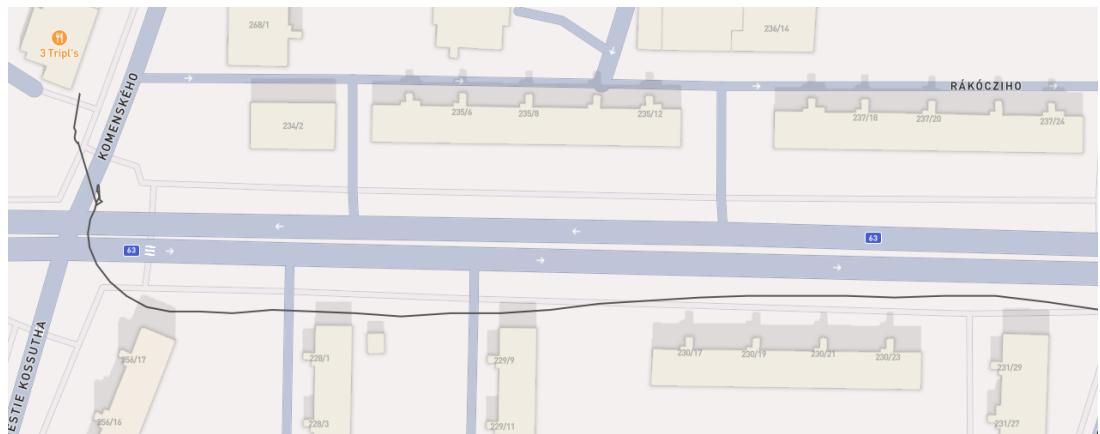
5.3 Pripínanie trás k cestnej sieti

Základná myšlienka

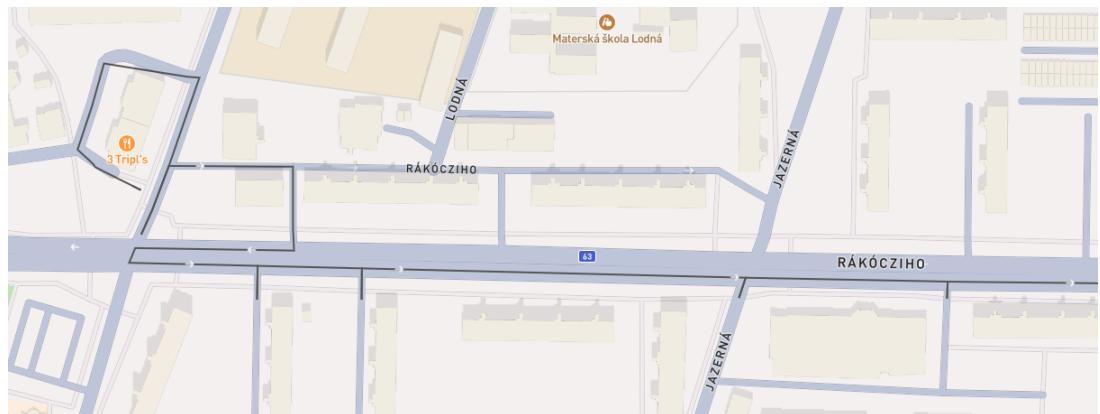
Po oboznámení sa s knižnicami, ktoré poskytujú funkciu pripínania trás k cestnej sieti sme dospeli k záveru, že trasy môžeme pripnúť dvoma spôsobmi[19][26][39]. Pomocou funkcie *routing* alebo pomocou funkcie *map-match*.

Prostredníctvom funkcie *routing* by sme medzi každým bodom v trase našli optimálnu trasu, podobne ako v navigácii. Spojením týchto trás dostaneme výslednú trasu pripnutú k cestnej sieti. Môže však nastať situácia, kedy *routing* funkcia kvôli nepresnosti GPS pripne bod k druhej ceste, viditeľné na obrázku 13b. Pre odstránenie takýchto situácií vytvoríme skript, ktorý odstráni body, ktoré sa v trase opakujú dva alebo viackrát. Takisto odstráni aj body medzi týmito dvoma opakujúcimi sa bodmi. Výslednok je možné vidieť na obrázku 13c. Po otestovaní zistili, že algoritmus odstraňujúci duplicitné body môže odstrániť aj dôležité body, ktoré by v trase mali zostať. Rozhodli sme sa v tomto riešení nepokračovať.

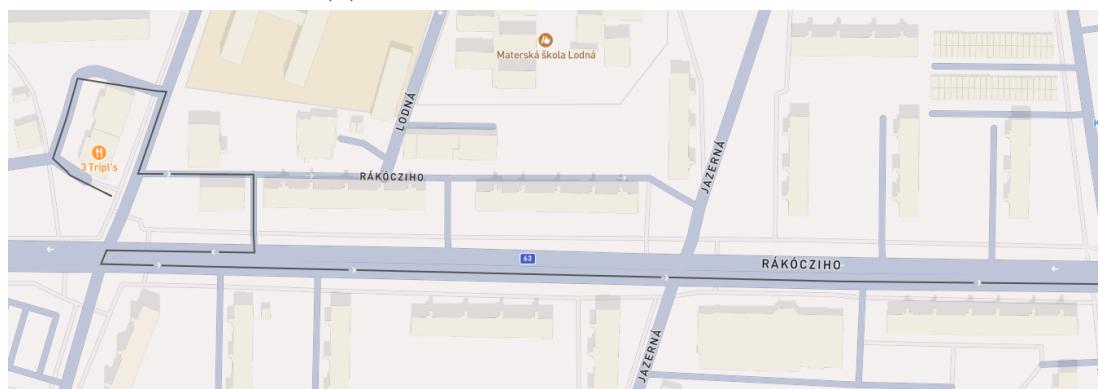
Druhou možnosťou je použiť funkciu *map-match*, ktorá body v trase posunie k cestnej sieti. V prípade, že body v trase nie sú dostatočne husté, môže byť výsledná trasa hranatá a jej tvar nemusí zodpovedať cestnej sieti. Toto je možné vidieť na obrázku 14b. Obrázok 14a znázorňuje trasu pred použitím *map-match* funkcie. Na obrázku 14b vidieť, že body trasy boli pripnúté k ceste, avšak nebolo ich dosť a výsledná trasa nezodpovedá cestnej sieti.



(a) Pôvodná trasa



(b) Trasa po použití *routing* funkcie



(c) Trasa po použití *routing* funkcie s odstránením duplicitných bodov

Obr. 13: Trasa priprnutá pomocou *routing* funkcie



(a) Trasa pred použitím *map-match* funkcie (b) Trasa po použití *map-match* funkcie

Obr. 14: Trasa s nedostatočne hustým počtom bodov

Optimalizácia výsledkov

Pre dosiahnutie lepšieho pripnutia trasy k cestnej sieti body interpolujeme. To znamená, že medzi každú dvojicu po sebe idúcich bodov v trase vložíme ďalší bod do stredu medzi dva vybrané. Na obrázku 15 možno vidieť pôvodné body v modrom kruhu a pridané body označené červeným znakom. Týmto spôsobom dostaneme trasu obsahujúcu viac bodov, ktorá môže byť lepšie pripnutá k cestnej sieti.

Vyskúšali sme rôzne nastavenia interpolácie, napríklad opakovanie iterovanie pridávania bodov medzi pôvodné body, až pokial vzdialenosť medzi bodmi nebola menšia ako x metrov, pričom x bola premenná, ktorú sme menili. Po vyskúšaní rôznych nastavení interpolácie je viditeľné, že tvar trasy viac zodpovedá cestnej sieti. Táto optimalizácia však nie je dostatočná, preto vyskúšame inú. Rozdiel je pozorovateľný na obrázkoch 16a a 16b.



Obr. 15: Znázormenie interpolácie bodov



(a) Pred použitím interpolácie

(b) Po použití interpolácie

Obr. 16: Optimalizácia pripnutia trasy k cestnej sieti pomocou interpolácie bodov

6 Implementácia

6.1 Pripínanie trasy k cestnej sieti

Ukladanie trás spracovaných funkciou *map-match*, rovnako ako ukladanie pôvodných bodov trás prebieha pomocou skriptu, ktorý je implementovaný v *Python*. Tento skript dostane na vstup slovník, ktorý obsahuje konfiguračné nastavenia skriptu, ako aj parametre pripínania trás k cestnej sieti. Medzi nastavenia skriptu patrí:

- unzipdir - cesta k priečinku, ktorý má byť spracovaný *map-match* algoritmom
- container - názov *Valhalla* kontajnera, ktorý poskytuje *map-match* funkcionalitu.
- user - meno používateľa, ktorý nahral súbor alebo požiadal o znova-spustenie algoritmu
- zipname - názov nahraného ZIP súboru
- parameters - vnorený slovník, ktorý obsahuje parametre pripínania trás k cestnej sieti.

Po načítaní nastavení skriptu prebehne inicializácia pomocných premenných, do ktorých sa bude ukladať priebeh pripnutia trás k cestnej sieti. Inicializuje sa *successful* premenná, ktorá reprezentuje pole, do ktorého sa budú ukladať názvy úspešne spracovaných trás. Taktiež sa inicializuje *failed* premenná, ktorá reprezentuje slovník, do ktorého sa budú ukladať názvy neúspešne spracovaných trás ako klúče a textový retazec informujúci chybu ktorá nastala ako hodnotu ku klúču.

Po inicializácii prebehne kontrola *unzipdir* nastavenia. Kontroluje sa, či zadaná cesta odkazuje na priečinok, ktorý obsahuje len priečinky. V prípade, že cesta odkazuje na priečinok, ktorý obsahuje súbor, algoritmus končí a výstupom algoritmu je slovník, ktorý obsahuje chybovú hlášku "**názov_súboru** is not a directory, check the zip structure.", ktorá je neskôr zobrazená používateľovi. Ďalej sa kontrolujú názovy podpriečinkov, ktoré priečinok s cestou *unzipdir* obsahuje. Ak sa v priečinku nachádza priečinok s názvom iným ako *Walk* alebo *Drive*, algoritmus končí a výstupom je slovník, ktorý obsahuje chybovú hlášku "**názov_podpriečinku** does not match the specified directory name 'Walk' or 'Drive', check the zip structure." Následne sa prechádzajú jednotlivé súbory (trasy) najprv v *Drive* priečinku, neskôr v *Walk* priečinku. Skript na základe mena súboru identifikuje, o aký typ súboru ide a podľa toho načíta body trasy do premennej. Skript dokáže načítať body z *csv*, *geojson* a *gpx* súboru. V prípade, že je súbor iného typu, do premennej *failed* sa uloží meno trasy ako klúč a chybová hláška "*Points couldn't be extracted.*" ako hodnota.

Po načítaní bodov do premennej vstupuje táto premenná do funkcie *map-match*. Do tejto funkcie vstupujú aj ďalšie argumenty, *container*, *parameters* a *costing*. Premenná *costing* je určuje typ dopravy a je nastavená na základe priečinka, z ktorého bola trasa načítaná. Ak bola trasa načítaná z priečinka *Walk*, ide o pešiu chôdzu a do premennej sa uloží hodnota *pedestrian*. V opačnom prípade sa do nej uloží hodnota *auto*, reprezentujúca jazdu autom. Premenná *parameters* reprezentuje slovník parametrov pripínania trás k cestnej sieti. Patria sem parametre, ktoré určujú presnosť GPS zariadenia v metroch, ktorým bola trasa meraná, dĺžku polomeru kruhu, v ktorom sa majú hľadať kandidáti cestnej siete, ku ktorým sa má bod v trase pripnúť, taktiež v metroch a penalizácia odbáčania na druhú cestu určená celočíselnou hodnotou. Vo funkcií sa naformátujú body spolu s parametrami a druhom dopravy do textového refazca, ktorý je vložený do požiadavky na *Valhalla* kontajner. Ukážku textového refazca je možné vidieť vo výpise 1.

```
{
  "shape": [{"lat": 47.993351,"lon": 18.174553}, {"lat": 47.993351,"lon": 18.174553}],
  "shape_match": "map_snap",
  "costing": "auto",
  "costing_options": {"pedestrian": {"ignore_access": true}},
  "format": "osrm",
  "trace_options": {
    "search_radius": 50,
    "turn_penalty_factor": 200,
    "gps_accuracy": 5
  }
}
```

Listing 1: Dáta požiadavky na *Valhalla* kontajner

Je možné vidieť jednotlivé parametre pripínania a ostatné nastavenia. Nastavenie *ignore_access* pre pešiu chôdzu nastavujeme, aby sme algoritmu oznámili, že má ignorovať typy ciest pri hľadaní kandidátov cestnej siete, ku ktorým má trasu pripnúť. Vďaka tomuto nastaveniu môže algoritmus pripnúť pešiu chôdzu aj na cyklotrasu.

Po odoslaní požiadavky na *Valhalla* kontajner a prijatí odpovede funkcia vráti odpoveď, ktorá je neskôr spracovaná. Z ukážky 2 možno vidieť, že trasa je zložená z rôznych segmentov, označenými atribútom *legs*. V ukážke 2 je zobrazený len jeden segment, aby výpis nebol zbytočne veľký. Je možné vidieť, že geometria nájdenej trasy obsahuje geometriu z prvého segmentu v *legs* atribúte.

```
{
  {
    "matchings": [
      {
        "weight_name": "pedestrian", "weight": 10027.532, "duration": 7257.535, "distance": 10239.001,
        "legs": [
          {
            "via_waypoints": [], "admins": [{}], "weight": 10027.532, "duration": 7257.535,
            "steps": [
              {
                "order": 1, "x": 18.174553, "y": 47.993351, "type": "walk"
              },
              {
                "order": 2, "x": 18.174553, "y": 47.993351, "type": "walk"
              }
            ]
          }
        ]
      }
    ]
  }
}
```

```

        "intersections": [
            { "bearings": [ 253 ], "entry": [ true ], "admin_index": 0, "out": 0,
              "geometry_index": 0, "location": [ 17.063915, 48.158126 ] }
        ],
        "maneuver": {
            "instruction": "Walk west.",
            "type": "depart",
            "bearing_after": 253,
            "bearing_before": 0,
            "location": [ 17.063915, 48.158126 ]
        },
        "name": "",
        "duration": 4.941,
        "distance": 7,
        "driving_side": "left",
        "weight": 4.941,
        "mode": "walking",
        "geometry": "{yizzAu}np_@b@rD"
    }],
    "distance": 10239.001,
    "summary": "Stare Grunty, Dvorakovo nabrezie"
}
],
"geometry":
    "{yizzAu}np_@b@rDxHaD~GsGjHiH|BsBpAdAgFnUs@~CzIoClp@mSni@}0j\\kJt@UltBun@jLgD|\\gKnKaG~QyKrBaA~EoCjK{GfNgKzLcK~AcC~CqFxc@{j@jIoKrB}CjGiJxt@q'AllKjLbCtSlCbEeCtDiGr@}J~@_b@NcHRyIhByz@|@e`@f@)T VcJhL^hJl@tETLlh@d@mSt@DcA|c@GdA_AYq@GhAmd@f@qJh@ag@z@g`pAiQjCaQbAaEhAeDbEkKdBgEhCkEhByBdDoC'Ri MnAg@bEk@`@UTq@jBmFvBiGXYjAgAfPuArBn@hSvFdYbIjFtAjh@vN|InCdGh@NiEd@gNeAGD{A@]Bu@HuBhAJGvBEpAw@I DqADwAr@)Wf~Hny@jw@fHla@tBBg@]1L1@F_BkYmBiEu}sHio@{CcWcCcWkAkQgA{WYyNO{N@yMhsM\`s0t@qQr@yL`A}L dCgV|C_VzFgc@bs@qsF|m@omGxs@ynHbC_WjAoZVqSEkS_AeZkBkZeD{W_Gq]}BuN_BoNaAeRUG0~AexAJuKnA{jAm@wSgB qNiCyNgGiSmAkHQoCYeEqIsmBMkCo@wMh@A@CnEK?}@egEpHs^gAq@CAeLj@?x@ACuKAgCbJYnC?bA?r@?v@?p@?hA?rC? fAhJDzTFzPBrNgC|zA}AjrA_BxrA_|@}Iht@kIrw@", "confidence": 1
}
],
"tracepoints": [
{
    "matchings_index": 0, "waypoint_index": 0, "alternatives_count": 0,
    "distance": 43.15, "name": "", "location": [ 17.063915, 48.158126 ]
}, { ... }
],
"code": "Ok"
}
}

```

Listing 2: Odpoved z *Valhalla* kontajnera

Z odpovede získame atribút *geometry*, ktorý reprezentuje zakódovanú geometriu cestnej siete, po ktorej sa predpokladá, že trasa prechádzala. Táto geometria sa dekóduje na body, ktoré sú uložené do súboru s názvom *map-match.csv*. Pôvodné body sú uložené do súboru s názvom *original.csv*. Tieto dva súbory sú potom uložené do priečinka *routes* pre príslušného používateľa, príslušný nahraný súbor a príslušnú trasu. Viac o štruktúre

routes priečinka je popísané v kapitole 5.2.

Nakoniec sa inicializuje premenná *retDict*, ktorá bude reprezentovať slovník. Príklad *retDict* premennej vidno na výpise 3, obsahuje nasledujúce kľúč-hodnota páry:

- failed - počet neúspešne spracovaných trás
- successful - počet úspešne spracovaných trás
- failed-info - premenná *failed*, ktorá obsahuje názov trasy ako kľúč a textový reťazec s informáciou o chybe, ktorá nastala ako hodnotu

Výstupom skriptu je premenná *retDict* konvertovaná na textový reťazec. Informácie z tohto slovníka sú neskôr zobrazené používateľovi, aby v prípade neúspešného pripnutia trás dostať informáciu o chybe a chybnej trase.

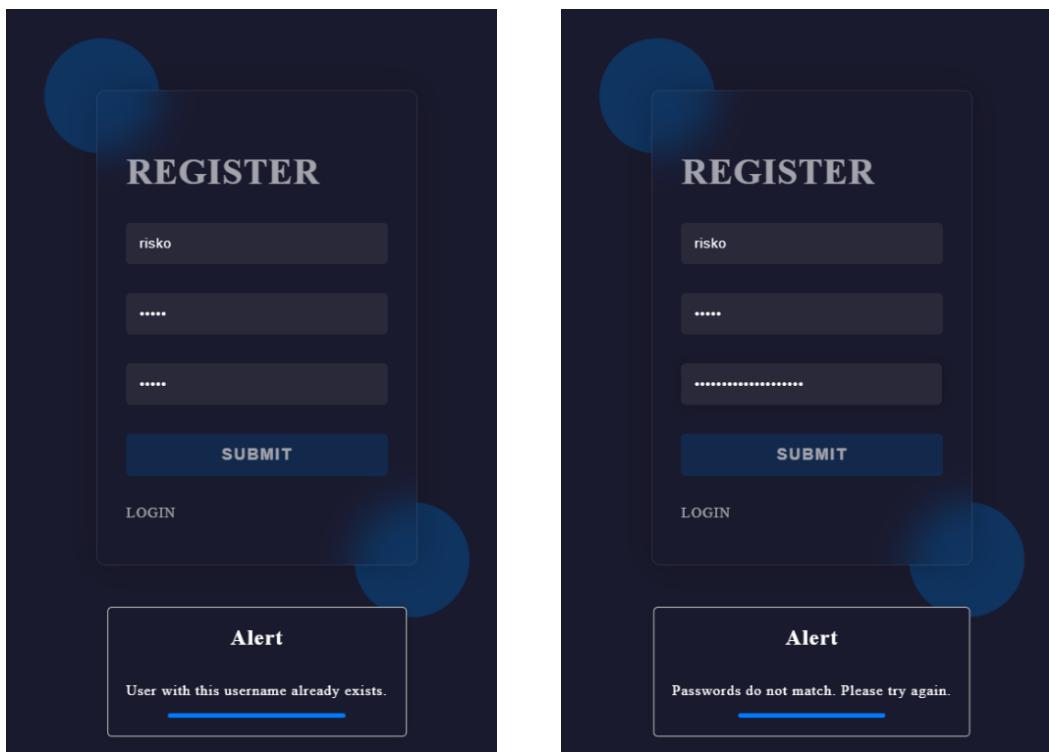
```
{  
  {  
    "failed": 3,  
    "successful": 519,  
    "failed_info": {  
      "u16112023_187923-189171_2024-01-13200431": "b'{\"code\": \"NoRoute\", \"message\": \"Impossible  
route between points\"}",  
      "u16112023_74346-75016_2023-12-09121131": "b'{\"code\": \"NoSegment\", \"message\": \"One of the  
supplied input coordinates could not snap to street segment.\"}",  
      "u16112023_85191-85958_2023-12-09151521": "b'{\"code\": \"NoSegment\", \"message\": \"One of the  
supplied input coordinates could not snap to street segment.\"}"  
    }  
  }  
}
```

Listing 3: *retDict* premenná

6.2 Webová aplikácia

6.2.1 Autentifikácia

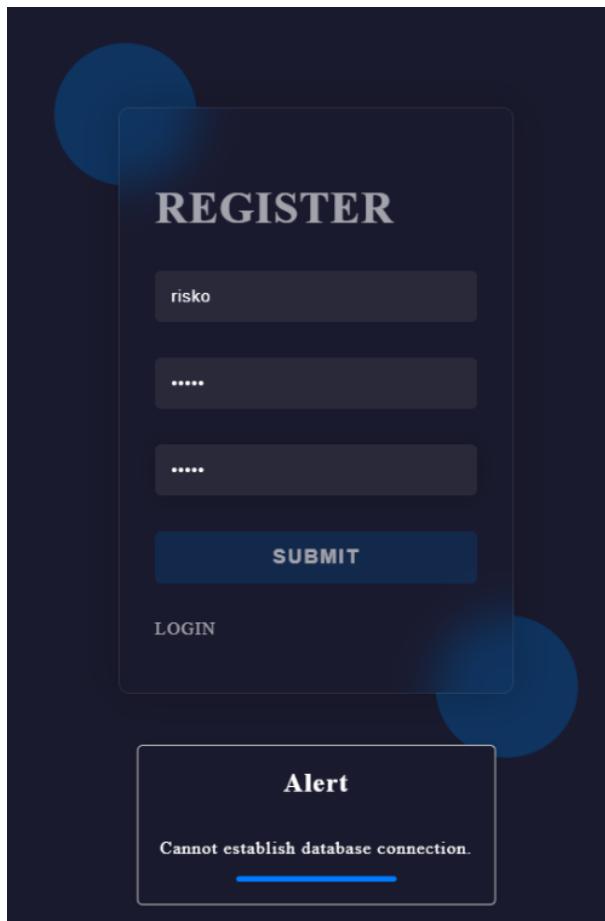
Po otvorení webovej aplikácie je používateľovi zobrazená stránka, na ktorej sa musí autentifikovať. Pre použitie aplikácie je potrebné, aby sa používateľ zaregistroval a neskôr prihlásil. Pri registrácii je nutné zadať používateľské meno, ktoré musí byť unikátné. To znamená, že nemôžu existovať dvaja používatelia s tým istým používateľským menom. Pri pokuse o registráciu používateľského mena, ktoré je už registrované, je používateľovi zobrazená chybová hláška viditeľná na obrázku 17a. Pri registrácii je taktiež nutné zadat heslo. Toto heslo je nutné zopakovať znova, aby sa predišlo prípadným chybám alebo preklepom v hesle. Pri pokuse o registráciu s nezhodnými heslami je používateľovi zobrazená chybová hláška viditeľná na obrázku 17b.



(a) Chybová hláška pri pokuse o registráciu existujúceho používateľského mena. (b) Chybová hláška pri pokuse o registráciu s nezhodnými heslami.

Obr. 17: Možné chybové hlášky pri registrácii pri chybe používateľa.

Chyba môže nastať aj na strane aplikácie, napríklad ak je databáza nedostupná alebo nie je zapnutá. Pri vypnutej alebo nedostupnej databáze sa zobrazí chybová hláška zobrazená na obrázku 18. Pri inej možnej chybe sa zobrazí chyba podobne s chybovým kódom.



Obr. 18: Chybová hláška pri nedostupnej databáze.

Po vyplnení údajov a kliknutí na tlačidlo *Submit* sú údaje z polí vložené do požiadavky, ktorá je odoslaná na server. Na serveri sa z požiadavky extrahuje meno a heslo. Heslo je *zahashované* pomocou knižnice *Bcrypt*[40]. *Zahashované* heslo spolu s používateľským menom je vložené do databázy. Pri chybe, napríklad duplicitnom zázname používateľského mena, nastane konflikt a databáza vygeneruje chybovú hlášku, ktorá je spracovaná a zobrazená používateľovi. Pri správnom vložení údajov do databázy je používateľ presmerovaný na stránku prihlásenia.

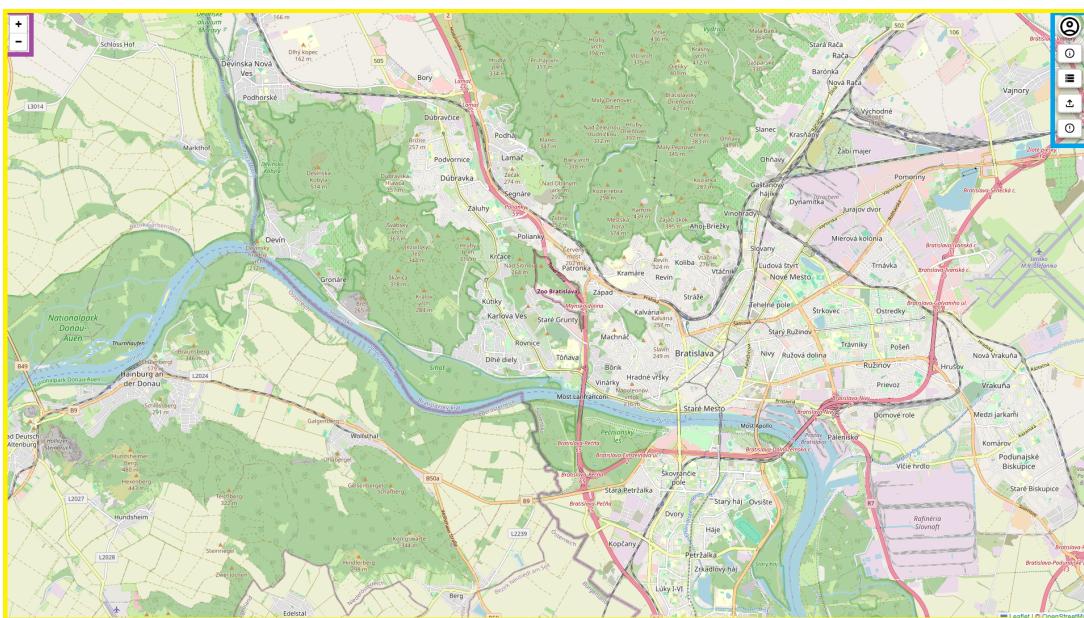
Na stránke prihlásenia používateľ zadá používateľské meno a heslo. Po kliknutí na tlačidlo *Submit* sú údaje vložené do požiadavky, ktorá je odoslaná na server. Tieto údaje sú na serveri extrahované z požiadavky. Na databázu je vytvorený dotaz s používateľským menom. V prípade, že sa používateľské meno nenachádza v databáze, používateľovi je zobrazená chybová hláška *"Invalid username or password"*. V prípade, že sa používateľské meno nachádza v databáze, heslo z požiadavky je *zahashované* pomocou *Bcrypt* funkcie a porovnané so *zahashovaným* heslom z databázy. V prípade, že sa heslá rovnajú, používateľ

je presmerovaný na stránku s mapou. V opačnom prípade je používateľovi zobrazená hláška *Invalid username or password*".

6.2.2 Stránka s mapou

Po autentifikácii je používateľ presmerovaný na stránku s mapou. Stránka je zložená z troch vždy viditeľných častí a šiestich častí viditeľných na základe akcií vykonaných používateľom. Tieto časti stránky sú vytvorené pomocou *div* HTML tagov. *Div* tagy slúžia na definovanie rozdelenia stránky alebo sekcií na stránke. Vždy viditeľné časti stránky sú zobrazené na obrázku 19 a sú vyznačené farebnými obdlžníkmi:

- mapa - žltý obdlžník
- tlačidlá na priblíženie mapy - fialový obdlžník
- panel akcií - modrý obdlžník



Obr. 19: Stránka s mapou

Mapa

Pomocou HTML je definovaný *div* tag, ktorý je prázdny. Má nastavený identifikátor s hodnotou "*map*". Mapa je inicializovaná pomocou *Leaflet* knižnice[36]. Pri inicializácii konštruktora mapy sa poskytuje identifikátor HTML elementu *div* a nastavujú sa súradnice pre zobrazenie mapy. V súčasnosti je predvolená konfigurácia mapy nastavená tak, aby zobrazovala Bratislavu, pričom stred mapy sa nachádza v Mlynskej Doline v Bratislave. Po inicializácii bude mapa zobrazená na stránke. Je responzívna, takže pri rôznych veľkostiah

displejov alebo pri rôznych veľkostiah okna je mapa vždy zobrazená na celej stránke. Trasy na mape sú zobrazené modrou a červenou čiarou. Červenou čiarou sú zobrazené pôvodné trasy a modrou farbou trasy pripnuté k cestnej sieti. Pri prejdení myšou ponad zobrazené trasy sa zobrazí okno, ktoré obsahuje trasy ležiace pod ukazovateľom myši. Táto funkciionalita je implementovaná z dôvodu, že môže na jednej ceste ležať niekoľko trás po pripnutí.

Tlačidlá na priblíženie mapy

Tlačidlá na priblíženie sú súčasťou mapy a poloha tlačidiel je nastavená pomocou *Leaflet* knižnice[36]. Tlačidlo s označením '+' slúži na priblíženie mapy a '-' slúži na oddialenie. Na počítačoch je možné mapu priblížiť alebo oddiaľiť pomocou rolovacieho kolieska na myši. Na mobilných zariadeniach sa tento efekt dosiahne gestom priblíženia alebo oddialenia, potiahnutím prstami po obrazovke. Mapu je možné priblížiť aj dvojklikom, či už na mobilnom zariadení alebo počítači.

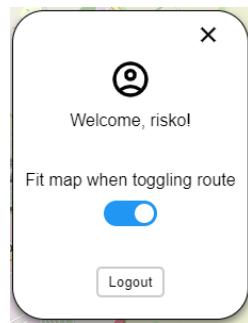
Panel akcii

Panel sa skladá z niekoľkých tlačidiel, ktoré po kliknutí vykonávajú rôzne akcie:

- Profile  *
- About 
- Show Files 
- Upload File 
- Error Files 

Po kliknutí na tlačidlo *Profile* sa otvorí okno, ktoré zobrazuje uvítaciu hlášku s používateľským menom. Taktiež obsahuje prepínač, ktorý nastavuje, či sa má mapa prispôsobiť zobrazeným trasám pri zobrazení alebo skrytí trasy a tlačidlo, ktoré po kliknutí odhlási používateľa. Okno je zobrazené na obrázku 20.

*Obrázok prevzatý z <https://fonts.google.com/icons>.



Obr. 20: Dialógové okno zobrazené po kliknutí *Profile* tlačidla.

Tlačidlo *About* používateľa presmeruje na druhú stránku, kde je možné vidieť informácie o webovej aplikácii spolu s návodom na použitie aplikácie.

Pre zobrazenie nahraných súborov používateľ klikne na tlačidlo *Show Files*. Otvorí sa dialógové okno viditeľné na obrázku 21, ktoré zobrazí zoznam nahraných súborov v tabuľke. Riadok v tabuľke reprezentuje nahraný súbor.

List of files				
File Name	Creation Date	Original route	Map-matched route	Bad map-match
strava_run	2024-04-25T06:26:53Z	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	!
zipecko	2024-04-29T10:20:14Z	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	!
zkpo	2024-04-29T10:20:26Z	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	!

Obr. 21: Dialógové okno zobrazené po kliknutí *Show Files* tlačidla.

Tabuľka obsahuje stĺpce:

- File Name - názov nahraného ZIP súboru, po kliknutí na názov súboru sa otvorí nové dialógové menu, ktoré zobrazuje trasy, ktoré obsahujе nahraný ZIP súbor

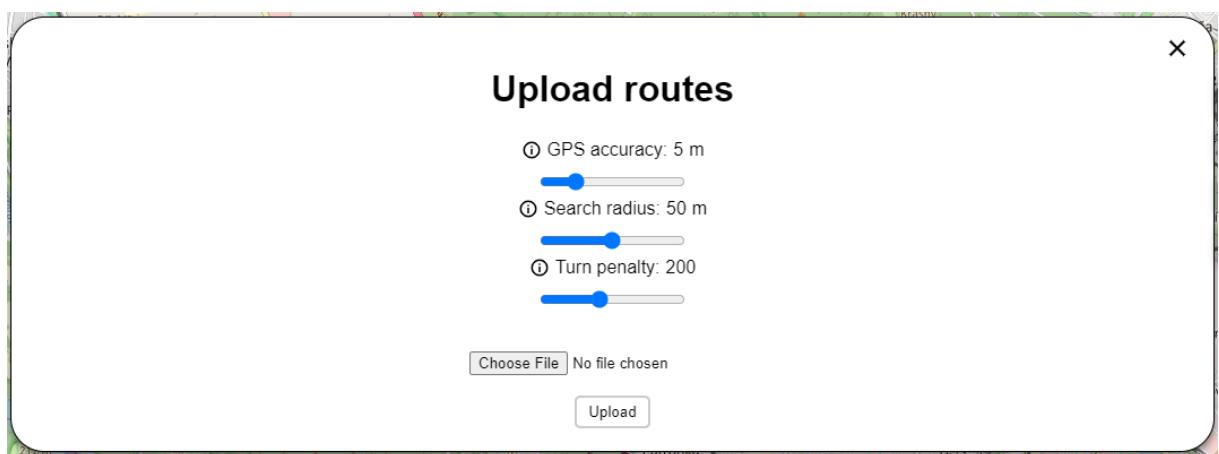
Routes of strava_run			
File Name	Original route	Map-matched route	Bad map-match
Evening_Run	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	!
strava	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	!

Obr. 22: Dialógové okno zobrazené po kliknutí na názov nahraného ZIP súboru.

- Creation Date - dátum a čas, kedy bol ZIP súbor spracovaný

- Original route - prepínač, ktorý zobrazí na mape všetky pôvodné trasy, ktoré ZIP súbor obsahuje
- Map-matched route - prepínač, ktorý zobrazí na mape všetky trasy priprnuté k cestnej sieti, ktoré ZIP súbor obsahuje
- Bad map-match - ak trasa nie je správne priprnutá k cestnej sieti, stlačením tohto tlačidla používateľ odošle informácie správcovi systému. Tieto informácie zahŕňajú názov trasy, názov nahraného súboru a používateľské meno

Po kliknutí na tlačidlo *Upload file* sa otvorí dialógové okno, ktoré obsahuje vstupné polia a tlačidlo. V okne sú tri vstupné polia typu *slider*, ktoré nastavujú parametre priprnutia trasy k cestnej sieti. Každý parameter je nastavený na predvolenú hodnotu, no používateľ môže hodnotu meniť. Ku jednotlivým parametrom sú vysvetlivky, ktoré sa zobrazia po prejdení myšou po ikone ⓘ. Posledné vstupné pole slúži na výber ZIP súboru. Nahranie súboru je nutné potvrdiť stlačením tlačidla *Upload*.



Obr. 23: Dialógové okno zobrazené po kliknutí *Upload File* tlačidla.

Pre zobrazenie chybových nahraných súborov používateľ klikne na tlačidlo *Error Files*. Kliknutím otvorí dialógové okno, ktoré pomocou tabuľky zobrazí nahrané ZIP súbory, v ktorých sa pri pripínaní trás k cestnej sieti vyskytla chyba. Každý riadok v tabuľke reprezentuje nahraný súbor. Tabuľka obsahuje stĺpce:

- File Name - názov nahraného ZIP súboru
- Download zip - tlačidlo, ktorým je možné stiahnuť nahraný ZIP súbor. Je zobrazené vždy, keď je na serveri nahraný ZIP súbor. Nie je zobrazené iba v špeciálnom prípade,

kedy sa používateľ pripojí na server pomocou *FTP* a nahrá súbory na server bez komprimovania alebo rozbalovania trás do ZIP súboru.

- Rerun map-matching - tlačidlo, ktorým sa znova-spustí algoritmus pripínania trás k cestnej sieti. Je viditeľné vždy, keď je v priečinku používateľa rozbalený nahraný ZIP súbor. Nie je zobrazené len v prípade, kedy sa nepodarilo rozbalíť nahraný súbor.
- Delete zip and unzipped files - tlačidlá, ktoré vymažú zo servera nahraný ZIP súbor, alebo rozbalený ZIP súbor. Sú zobrazené pod podmienkou, že sa na serveri nachádza v priečinku používateľa rozbalený ZIP súbor, alebo/aj samotný ZIP súbor.

Ukážku okna je možné vidieť na obrázku nižšie.

File Name	Download zip	Rerun map-matching	Delete zip and unzipped files
test	⬇️	⟳	🗂️
zip_with_pic	⬇️	⟳	🗂️

Obr. 24: Dialógové okno zobrazené po kliknutí *Error Files* tlačidla.

Po znova-spustení algoritmu pripínania trás k cestnej sieti je po úspešnom pripnutí nutné manuálne vymazať nahraný ZIP súbor a rozbalené súbory ZIP súboru tlačidlami zobrazenými v *Delete zip and unzipped files*.

7 Zhodnotenie

V tejto sekcií predstavíme výsledok našej práce. Webovú aplikáciu sme nasadili na verejný server, aby sme ju mohli odovzdať na testovanie reálnym používateľom. Týmto spôsobom získame objektívnu odozvu a hodnotenie našej aplikácie.

7.1 Výsledok práce

Vytvorili sme webovú aplikáciu pomocou knižnice *Node.js*. Naštudovali sme trasovací engine *Valhalla* a spôsoby jeho použitia. Funkcie, ktoré *Valhalla* poskytuje sa nám podarilo integrovať do našej webovej aplikácie *Routiak*.

Routiak je webová aplikácia, pomocou ktorej si môže používateľ vizualizovať vlastné trasy na mape prehľadne. To znamená, že aj pri veľkom počte trás (10 a viac) prechádzajúcich tým istým miestom, môže používateľ nastaviť zobrazenie tak, aby bolo prehľadné a na mape nevznikali tzv. "špagety" (obr.25). Vďaka správnemu nastaveniu zobrazenia je možné vidieť trasy a cestnú sieť prehľadne. Používateľ sa môže v aplikácii registrovať a prihlásiť a teda je možné nahrané trasy zobraziť neskôr bez nutnosti znova trasy do aplikácie nahrať. Trasy sa do aplikácie nahrávajú v ZIP súbore, čo umožňuje nahrať viac trás súčasne. Tieto trasy je možné zobraziť na mape jednotlivo, alebo všetky naraz obsiahnuté v ZIP súbore. Aplikácia ponúka map-match funkcionality, čo znamená, že trasy budú pripnuté na cestnú sieť a tým sa zlepší viditeľnosť mapy a trás na mape, obrázok 26.

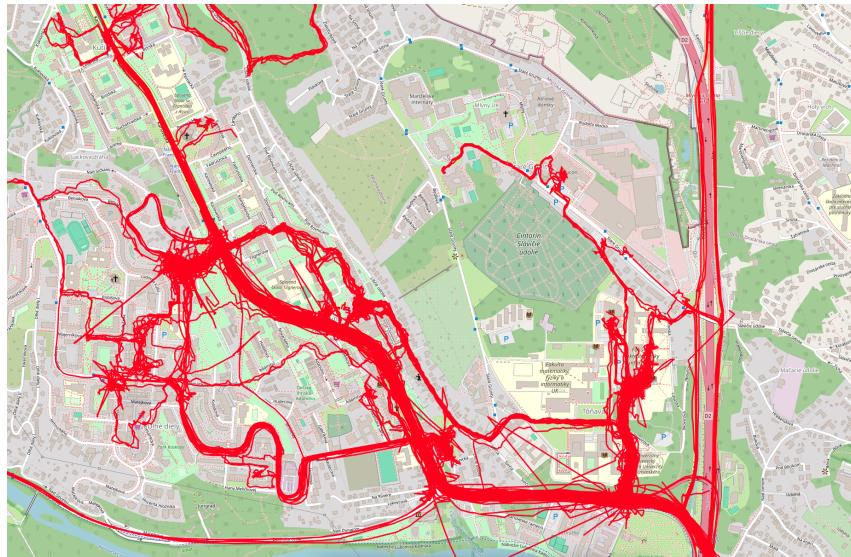
Výhody:

- map-match, pripnutie trás na cestnú sieť pre zlepšenie viditeľnosti
- možnosť nahrať viac trás naraz
- zobrazenie vlastných trás po nahratí bez nutnosti nahrávať trasy pri každom zobrazení
- možnosť nahrať trasy vo viacerých formátoch (CSV,GEOJSON,GPX)
- neobmedzený počet zobrazení mapy s neobmedzeným počtom nahraných súborov (počet nahraných súborov je obmedzený velkostou vyhradeného miesta na serveri)
- do jednej požiadavky na map-match môže ísť až 20 tisíc súradníc
- informácia pre používateľa v prípade, že niektorá z trás nemala požadovaný formát

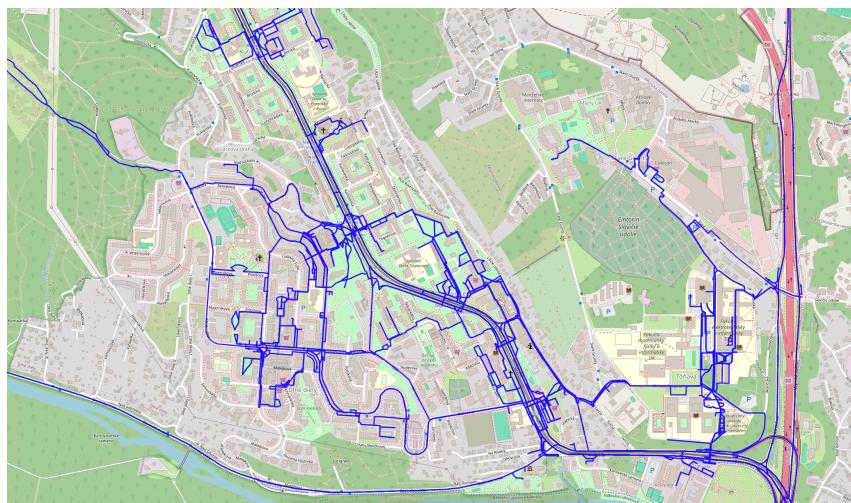
Nevýhody:

- trasa môže obsahovať maximálne 20 tisíc bodov

- nie je možné pripnúť trasu zaznamenanú rôznymi typmi dopravy (auto + pešia chôdza)
- priebeh *map-match* ZIP súboru nie je zobrazený, iba informácia po dokončení
- pri nahratí jednej trasy je nutné vložiť trasu do ZIP súboru so špeciálne určenou štruktúrou



Obr. 25: Zobrazenie väčšieho množstva trás na rovnakom mieste



Obr. 26: Zobrazenie väčšieho množstva trás upravených map-matchom na rovnakom mieste

7.2 Hodnotenie práce reálnymi používateľmi

Sem pojde dotaznicek TODO

Záver

Cieľom práce bolo navrhnúť a implementovať algoritmus, ktorý bude zobrazovať väčšie datasety GPS trajektórií prehľadne. To znamená, že používateľ bude vedieť zrakovo rozlísiť pohybové vzory. Výstup práce má byť zobrazený interaktívne vo webovom prehliadači, preto sme vytvorili webovú aplikáciu, v ktorej budeme trajektórie zobrazovať.

Najprv sme analyzovali problematiku zobrazovania GPS trajektórií na mape. Zistili sme, že pri veľkom množstve trás zobrazených na mape sa trasy prekrižujú a mapa sa stáva neprehľadnou. Tieto prekriženia vznikajú kvôli nepresnosti GPS zariadení, ktoré snímajú polohu používateľa.

Urobili sme prieskum trhu a zistili sme dostupné riešenia zobrazovania GPS trás. Tieto riešenia sme vyskúšali a analyzovali sme ich nedostatky, ktoré sme sa v našej aplikácii snažili odstrániť.

Rozhodli sme sa, že pre sprehľadnenie mapy budeme GPS trajektórie pripínať k cestnej sieti. To znamená, že každú trasu upravíme tak, aby ležala na ceste, po ktorej sa používateľ reálne pohyboval pri snímaní polohy pomocou GPS zariadenia. Vyskúšali sme dve metódy, ktorými môžeme trasy upravovať a rozhodli sme sa pre *map-match* metodiku, ktorá mala lepšie výsledky. Implementovali sme algoritmus, ktorý využíva funkciu *map-match Valhalla* enginu.

Po implementácii algoritmu sme naštudovali problematiku vývoja webových aplikácií a vytvorili sme webovú aplikáciu. V tejto webovej aplikácii sme integrovali náš algoritmus. Používateľ sa v aplikácii autentifikuje a nahrá svoje trasy, ktoré môže neskôr zobraziť pred a po úprave našim algoritmom. Používateľ môže meniť parametre pripínania trás k cestnej sieti a v prípade zlých výsledkov upozorniť správcu aplikácie.

Nakoniec sme vytvorili používateľský manuál, ktorý sme spolu s aplikáciou odovzdali testerom, ktorí aplikáciu testovali. Pre testerov sme vytvorili krátke dotazník, ktorý po otestovaní aplikácie vyplnili. Odpovede z dotazníka sme vyhodnotili a na základe odpovedí vieme povedať, že naša aplikácia zobrazuje dokáže zobraziť väčší počet GPS trajektórií tak, aby používateľ vedel zrakovo určiť pohybové vzory.

Zoznam použitej literatúry

1. BAJAJ, R., RANAWERA, S.L. a AGRAWAL, D.P. GPS: location-tracking technology. *Computer*. 2002, roč. 35, č. 4, s. 92–94. Dostupné z DOI: 10.1109/MC.2002.993780.
2. HEGARTY, Christopher J. The Global Positioning System (GPS). In: *Springer Handbook of Global Navigation Satellite Systems*. Ed. TEUNISSEN, Peter J.G. a MONTENBRUCK, Oliver. Cham: Springer International Publishing, 2017, s. 197–218. ISBN 978-3-319-42928-1. Dostupné z DOI: 10.1007/978-3-319-42928-1_7.
3. *How to Build a Web App in 12 Simple Steps* [online]. [cit. 2024-03-30]. Dostupné z : <https://kissflow.com/application-development/how-to-create-a-web-application/>.
4. *React* [online]. [cit. 2024-05-01]. Dostupné z : <https://react.dev/>.
5. *Rapidly build modern websites without ever leaving your HTML*. [online]. [cit. 2024-05-01]. Dostupné z : <https://tailwindcss.com/>.
6. *What is the role of JavaScript in web application programming?* [online]. [cit. 2024-04-06]. Dostupné z : <https://www.linkedin.com/advice/0/what-role-javascript-web-application-programming-rb4kc>.
7. [online]. [cit. 2024-05-01]. Dostupné z : <https://www.python.org/>.
8. [online]. [cit. 2024-05-01]. Dostupné z : <https://www.php.net/>.
9. *What is Java?* [online]. [cit. 2024-05-01]. Dostupné z : <https://aws.amazon.com/what-is/java/>.
10. *What is MySQL?* [online]. [cit. 2024-04-06]. Dostupné z : <https://www.oracle.com/mysql/what-is-mysql/>.
11. *PostgreSQL: The World's Most Advanced Open Source Relational Database* [online]. [cit. 2024-05-01]. Dostupné z : <https://www.postgresql.org/>.
12. [online]. [cit. 2024-05-01]. Dostupné z : <https://www.mongodb.com/>.
13. *Deliver web apps with confidence* [online]. [cit. 2024-05-01]. Dostupné z : <https://angular.io/>.
14. *The Progressive JavaScript Framework* [online]. [cit. 2024-05-01]. Dostupné z : <https://vuejs.org/>.

15. *Meet Django* [online]. [cit. 2024-05-01]. Dostupné z : <https://www.djangoproject.com/>.
16. *The PHP Framework for Web Artisans* [online]. [cit. 2024-05-01]. Dostupné z : <https://laravel.com/>.
17. *Express: Fast, unopinionated, minimalist web framework for Node.js* [online]. [cit. 2024-04-01]. Dostupné z : <https://expressjs.com/>.
18. [online]. [cit. 2024-05-01]. Dostupné z : <https://flask.palletsprojects.com/en/3.0.x/>.
19. *Mapbox* [online]. [cit. 2024-04-01]. Dostupné z : <https://github.com/mapbox>.
20. *Mapbox GL JS* [online]. [cit. 2024-04-01]. Dostupné z : <https://www.mapbox.com/mapbox-gl-js>.
21. *Navigation SDK for mobile* [online]. [cit. 2024-04-01]. Dostupné z : <https://www.mapbox.com/navigation-mobile>.
22. *MapGPT* [online]. [cit. 2024-04-01]. Dostupné z : <https://www.mapbox.com/mapgpt>.
23. *GPS Visualizer: Do-It-Yourself Mapping* [online]. [cit. 2024-04-05]. Dostupné z : <https://www.gpsvisualizer.com/>.
24. *What is Docker?* [online]. [cit. 2024-01-04]. Dostupné z : <https://aws.amazon.com/docker/>.
25. *What is Docker and what are its advantages?* [online]. [cit. 2024-01-04]. Dostupné z : <https://www.dimensiona.com/en/what-is-docker-and-what-are-its-advantages/>.
26. [online]. [cit. 2024-01-04]. Dostupné z : <https://valhalla.github.io/valhalla/>.
27. [online]. [cit. 2024-01-04]. Dostupné z : <https://valhalla.github.io/valhalla/odin/>.
28. *Make an impact with your location data* [online]. [cit. 2024-01-04]. Dostupné z : <https://kepler.gl/>.
29. *Visualizing Data on a map has never been easier Kepler.GL* [online]. [cit. 2024-01-04]. Dostupné z : <https://www.linkedin.com/pulse/visualizing-data-map-has-never-been-easier-keplergl-ali>.
30. *From Beautiful Maps to Actionable Insights: Introducing kepler.gl, Uber's Open Source Geospatial Toolbox* [online]. [cit. 2024-01-04]. Dostupné z : <https://www.uber.com/en-SK/blog/keplergl/>.

31. [online]. [cit. 2024-04-01]. Dostupné z : <https://github.com/expressjs/express/releases>.
32. *Express.js: The Good, the Bad, and the Ugly* [online]. [cit. 2024-04-01]. Dostupné z : <https://www.linkedin.com/pulse/expressjs-good-bad-ugly-aziz-taha/>.
33. [online]. [cit. 2024-04-01]. Dostupné z : <https://code.visualstudio.com/docs>.
34. *Announcing Visual Studio Code - Preview* [online]. [cit. 2024-04-01]. Dostupné z : <https://web.archive.org/web/20151009211114/http://blogs.msdn.com/b/vscode/archive/2015/04/29/announcing-visual-studio-code-preview.aspx>.
35. *Why did we build Visual Studio Code?* [online]. [cit. 2024-04-01]. Dostupné z : <https://code.visualstudio.com/docs/editor/whyvscode>.
36. *Leaflet: an open-source JavaScript library for mobile-friendly interactive maps* [online]. [cit. 2024-04-15]. Dostupné z : <https://leafletjs.com/>.
37. *Leaflet vs OpenLayers. What to choose?* [online]. [cit. 2024-04-15]. Dostupné z : <https://www.geoapify.com/leaflet-vs-openlayers>.
38. *Leaflet Quick Start Guide* [online]. [cit. 2024-04-15]. Dostupné z : <https://leafletjs.com/examples/quick-start/>.
39. *The GraphHopper Directions API Route Planning For Your Application* [online]. [cit. 2024-05-02]. Dostupné z : <https://www.graphhopper.com/>.
40. *bcrypt* [online]. [cit. 2024-04-28]. Dostupné z : <https://www.npmjs.com/package/bcrypt>.