Univerzita Palackého v Olomouci
Přírodovědecká fakulta
Katedra geoinformatiky

TVORBA KLIENTA PRO VYHLEDÁVAČ SPOJENÍ VEŘEJNOU DOPRAVOU OPENTRIPPLANNER 2

Bakalářská práce

Michal POTOČIAR

Vedoucí práce: Ing. Jan MASOPUST

Olomouc 2022
Geoinformatika a geografie

ANOTACE

Bakalářská práce se zabývá vytvořením webového plánovače spojení veřejnou dopravou pro vybraný region ČR. Serverová část používá vhodně nakonfigurovaný program OpenTripPlanner 2, data jízdních řádů Jihomoravského kraje ve formátu GTFS od společnosti KORDIS JMK a prostorová data regionu z OpenStreetMap. Webový klient je napsán v programovacím jazyce Python v kombinaci s jazyky pro psaní webu. Uživatelské rozhraní plánovače obsahuje formulářové okno, ve kterém si uživatel vybírá různé parametry cesty (druh přepravy, datum a čas, příjezd/odjezd) a interaktivní mapové pole umožňující zobrazení výsledných cest ve formě linie. Uživatel zadává počáteční místo a konečnou destinaci cesty výběrem v mapě nebo zapsáním adresy do formuláře. Plánovač také obsahuje informační panel, ve kterém jsou statistické údaje o výsledné cestě a detailní informace o jednotlivých krocích cesty.

KLÍČOVÁ SLOVA

OpenTripPlanner 2; Python; plánovač; programování

Počet stran práce: 42
Počet příloh: 2 (z toho 2 volné)

ANOTATION

The bachelor thesis deals with the creation of a web-based public transport connection planner for a selected region of the Czech Republic. The server part uses a suitably configured OpenTripPlanner 2 program, timetable data of the South Moravian Region in GTFS format from KORDIS JMK and spatial data of the region from OpenStreetMap. The web client is written in Python programming language in combination with web writing languages. The user interface of the planner includes a form window in which the user selects various trip parameters (type of transport, date and time, arrival/departure) and an interactive map field allowing the display of the resulting trips in the form of a line. The user enters the starting point and destination of the journey by selecting it on the map or by entering the address in the form. The planner also includes an information panel that provides statistics about the resulting trip and detailed information about each step of the trip.

KEYWORDS

OpenTripPlanner 2; Python; planner; programming

Number of pages: 42
Number of appendixes: 2

Prohlašuji, že

- bakalářskou práci včetně příloh jsem vypracoval samostatně a uvedl jsem všechny použité podklady a literaturu.

- jsem si vědom, že na moji bakalářskou práci se plně vztahuje zákon č.121/2000 Sb. - autorský zákon, zejména § 35 – využití díla v rámci občanských a náboženských obřadů, v rámci školních představení a využití díla školního a § 60 školní dílo,

- beru na vědomí, že Univerzita Palackého v Olomouci (dále UP Olomouc) má právo nevýdělečně, ke své vnitřní potřebě, bakalářskou práci užívat (§ 35 odst. 3),

- souhlasím, aby jeden výtisk bakalářské práce byl uložen v Knihovně UP k prezenčnímu nahlédnutí,

- souhlasím, že údaje o mé bakalářské práci budou zveřejněny ve Studijním informačním systému UP,

- v případě zájmu UP Olomouc uzavřu licenční smlouvu s oprávněním užít výsledky a výstupy mé bakalářské práce v rozsahu § 12 odst. 4 autorského zákona,

- použít výsledky a výstupy mé bakalářské práce nebo poskytnout licenci k jejímu využití mohu jen se souhlasem UP Olomouc, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly UP Olomouc na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Olomouci dne Michal Potočiar

Děkuji vedoucímu práce Ing. Janovi Masopustovi za podněty a připomínky při vypracování práce. Za poskytnuté rady a nápady ohledně knihovny jQuery děkuji i Tomášovi Potočiarovi. Také děkuji všem respondentům, kteří se podíleli na testování plánovače tvořeného v rámci této práce.

UNIVERZITA PALACKÉHO V OLOMOUCI

Přírodovědecká fakulta

Akademický rok: 2020/2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Michal POTOČIAR**
Osobní číslo:	**R190350**
Studijní program:	**B1301 Geografie**
Studijní obor:	**Geoinformatika a geografie**
Téma práce:	**Tvorba klienta pro vyhledávač spojení veřejnou dopravou OpenTripPlanner 2**
Zadávající katedra:	**Katedra geoinformatiky**

Zásady pro vypracování

Cílem bakalářské práce je vytvořit webový plánovač spojení veřejnou dopravou pro vybraný region ČR. Serverová část bude používat vhodně nakonfigurovaný program OpenTripPlanner 2 a data jízdních řádů ve formátu GTFS a prostorová data z OpenStreetMap. Student provede rešerši existujících vyhledávačů, programů pro vyhledávání v ČR a API i ve světě. Dále vybere na základě rešerše vhodnou technologii a programovací jazyk pro realizaci webového klienta.

Praktické části práce bude obsahovat instalaci a konfiguraci OpenTripPlanner a dat na serveru, tvorbu části klientské aplikace pro zadávání dotazů, tvorbu části klientské aplikace pro zobrazování výsledků ve formě textu a mapy, testování výsledné aplikace a její srovnání s existujícími vyhledávači.

Výstupem práce bude kód a postup tvorby webového vyhledávače spojení veřejnou dopravou připravený pro nasazení na datech jízdních řádů z ČR.

Celá práce (text, přílohy, výstupy, zdrojová a vytvořená data) se odevzdá v digitální podobě na paměťovém nosiči (CD, DVD, SD karta, flash disk). Text práce s vybranými přílohami bude odevzdán ve dvou svázaných výtiscích na sekretariát katedry. O diplomové práci student vytvoří webovou stránku v souladu s pravidly dostupnými na stránkách katedry. Práce bude zpracována podle zásad dle Voženílek (2002) a závazné šablony pro diplomové práce na KGI. Povinnou přílohou práce bude poster formátu A2.

Rozsah pracovní zprávy:	**max. 50 stran**
Rozsah grafických prací:	**dle potřeby**
Forma zpracování bakalářské práce:	**tištěná**

Seznam doporučené literatury:

1. GITHUB. **opentripplanner / OpenTripPlanner**. GitHub [online]. [San Francisco]: GitHub, 2021 [vid. 13. května 2021]. Dostupné z: https://github.com/opentripplanner/OpenTripPlanner
2. ESZTERGÁR-KISS, Domokos a Csaba CSISZÁR. Evaluation of Multimodal Journey Planners and Definition of Service Levels. *International Journal of Intelligent Transportation Systems Research* [online]. 2015, 13(3), 154-165 [cit. 2021-5-13]. ISSN 1348-8503. Dostupné z: doi:10.1007/s13177-014-0093-0
3. NĚTEK, Rostislav. *Webová kartografie – specifika tvorby interaktioních map na webu*. Olomouc: Univerzita Palackého v Olomouci, 2020. ISBN 978-80-244-5827-4.
4. HARTSON, H. Rex a Pardha S. PYLA. *The UX Book: process and guidelines for ensuring a quality user experience*. Boston: Elsevier, c2012. ISBN 978-0123852410.
5. GARRETT, Jesse James. *The elements of user experience: user-centered design for the Web and beyond*. 2nd ed. Berkeley, CA: New Riders, c2011. Voices that matter. ISBN 978-0321683687.
6. SHEKHAR S., XIONG H., ZHOU X. (eds.) Encyclopedia of GIS. Springer, Cham, 2017

7. VOŽENÍLEK, V. Diplomové práce z geoinformatiky. Olomouc, Univerzita Palackého v Olomouci, 2002

Vedoucí bakalářské práce: **Ing. Jan Masopust**
Katedra geoinformatiky

Datum zadání bakalářské práce: **4. května 2021**
Termín odevzdání bakalářské práce: **4. května 2022**

UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA
KATEDRA GEOINFORMATIKY
17. listopadu 50, 771 46 Olomouc
-1-

L.S.

doc. RNDr. Martin Kubala, Ph.D.
děkan

prof. RNDr. Vít Voženílek, CSc.
vedoucí katedry

V Olomouci dne 13. září 2021

OBSAH

SEZNAM POUŽITÝCH ZKRATEK

Zkratka	Význam
API	Application Programming Interface
CGI	Common Gateway Interface
CSS	Cascading Style Sheets
GIS	geografický informační systém
GPS	Global Positioning System
GTFS	General Transit Feed Specification
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JAR	Java ARchive
JDK	Java Development Kit
JS	JavaScript
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
MaaS	Mobility as a Service
MHD	městská hromadná doprava
MS4W	MapServer for Windows
OSM	OpenStreetMap
OTP 2	OpenTripPlanner 2
PBF	Protocolbuffer Binary
PHP	PHP: Hypertext Preprocessor
REST	Representational State Transfer
XLT	Legacy Excel templates

ÚVOD

Lidé v moderní době mají stále větší potřebu cestovat. Proto je nutné vytvářet různé systémy a aplikace na plánování cest. Díky těmto plánovačům dokážou lidé při přesouvání z místa na místo ušetřit mnoho času a budou se cítit bezpečněji v místech, kde ještě nikdy nebyli.

V dnešní době je velké množství plánovačů v různých podobách, proto je při jejich tvorbě nutné brát velký důraz na jejich kvalitu a zároveň jednoduchost. Nejpoužívanější inovací současnosti je možnost využívat data o dopravě, která jsou vytvářena v reálném čase přímo ze silnic. Tyto údaje pak mohou vstupovat do plánovačů a nežádoucí cesty, například dopravní zácpy, mohou být jednoduše eliminovány. Lidé tak ušetří velké množství času a starostí.

1 CÍLE PRÁCE

Cílem bakalářské práce je vytvořit webový plánovač spojení veřejnou dopravou pro vybraný region ČR. Serverová část bude používat vhodně nakonfigurovaný program OpenTripPlanner 2 a data jízdních řádů ve formátu GTFS a prostorová data z OpenStreetMap. Student provede rešerši existujících vyhledávačů, programů pro vyhledávání v ČR a API i ve světě. Dále vybere na základě rešerše vhodnou technologii a programovací jazyk pro realizaci webového klienta.

Praktické části práce budou obsahovat instalaci a konfiguraci OpenTripPlanner 2 a dat na serveru, tvorbu části klientské aplikace pro zadávání dotazů, tvorbu části klientské aplikace pro zobrazování výsledků ve formě textu a mapy, testování výsledné aplikace a její srovnání s existujícími vyhledávači.

Vytvořený plánovač umožní vyhledávat cesty různými dopravními prostředky po celém Jihomoravském kraji. V budoucnu může být použit na veškeré města, kraje i státy, které budou disponovat GTFS daty.

2 METODY A POSTUPY ZPRACOVÁNÍ

Na začátku této bakalářské práce bylo nutné pořádně nastudovat a seznámit se s danou problematikou, konkrétně s funkčností programu OpenTripPlanner 2. Na základě toho vybrat vhodnou technologii pro vytvoření webového klienta. Pro správné fungování klienta bylo zapotřebí vhodně nakonfigurovat a zprovoznit daný software a současně mít spuštěný webový server na lokálním zařízení. Bylo nutné seznámit se s CGI (Common Gatewy Interface) způsobem programování pomocí jazyku Python a se strukturou JSON (JavaScript Object Notation) formátu obsahujícího informace o vygenerovaných cestách. Na závěr byl klient vhodně otestován a nalezené chyby a nedostatky byly eliminovány.

Použité metody

Tvorba klienta pro plánování cest v této bakalářské práci byla rozdělena na dvě části. Nejprve proběhla konfigurace a zprovoznění serverů. Na začátku byla důkladně nastudována dokumentace i API (Application Programming Interface) softwaru a byl zprovozněn jeho testovací klient. Poté proběhla analýza, jakým způsobem získávat vygenerovaná data z programu OTP 2 (OpenTripPlanner 2) a vhodně je zobrazovat v novém klientovi. Bylo rozhodováno mezi dvěma možnostmi přístupu: použít programovací jazyk PHP (PHP: Hypertext Preprocessor) nebo CGI programování v Pythonu. Na základě jednoduché operability Pythonu s JSON souborem, který je generován programem OTP 2, byl vybrán způsob CGI.

V důležitější druhé části proběhla tvorba nového klienta pro plánování a zobrazování cest. Metoda programování CGI umožňovala naprogramovat klient pomocí jazyku Python v kombinaci s jazyky pro psaní webu. Nejprve bylo navrženo uživatelské rozhraní, do kterého byly postupně přidávány funkce a možnosti pro lepší vykreslování cesty. Veškeré informace byly získávány ze současně generované JSON struktury a zobrazovány ve vhodné podobě v Leaflet mapě a v jednotlivých krocích cesty ve formě textu.

Výsledný klient byl porovnán s ostatními vyhledávači a otestován několika odbornými i neodbornými uživateli. Na základě jejich připomínek byl klient dodatečně upraven a vylepšen.

Použitá data

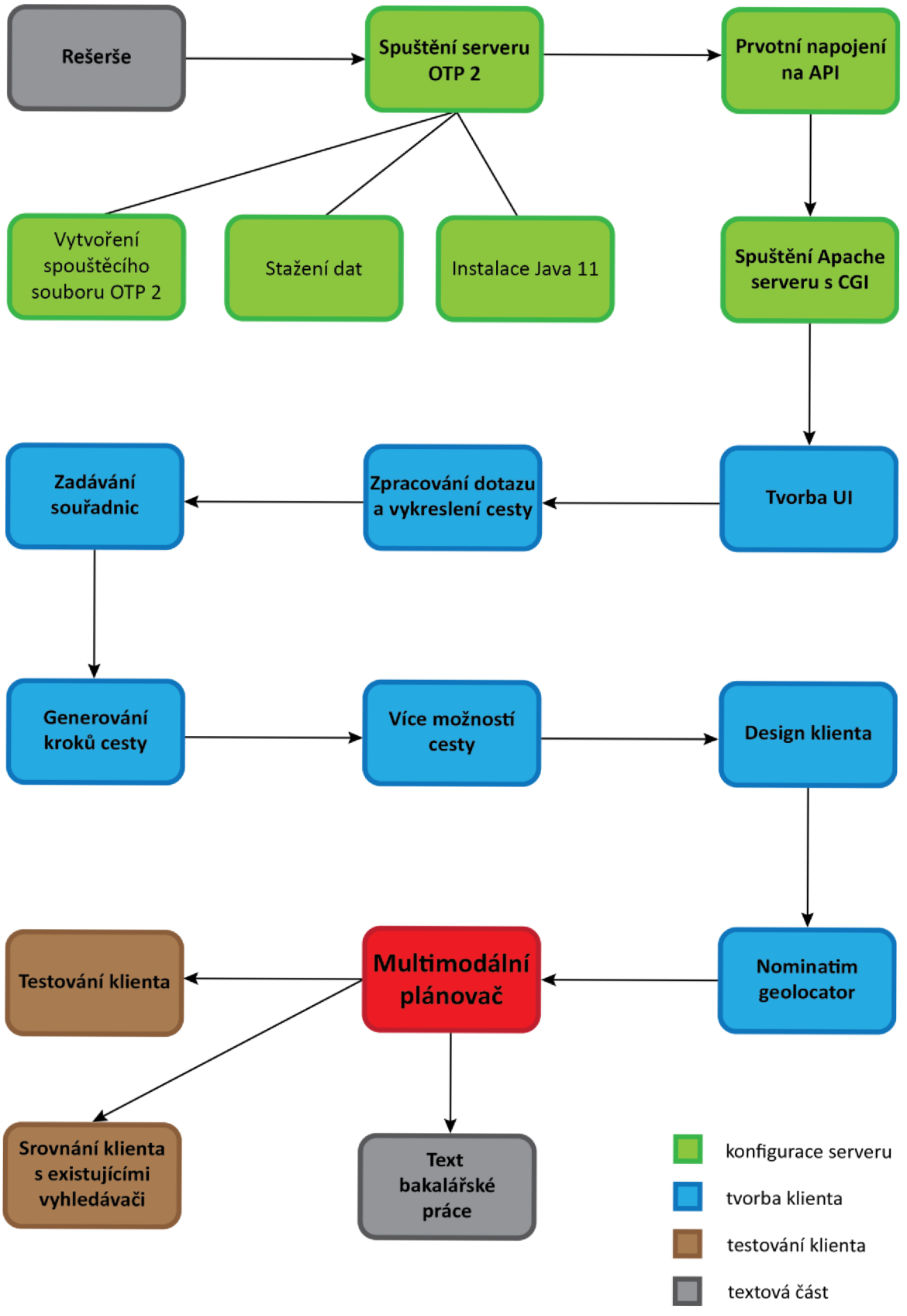
Pro správný chod programu OTP 2 byla stažena a použita data jízdních řádů ve formátu GTFS (General Transit Feed Specification) od společnosti KORDIS JMK pokrývající celý Jihomoravský kraj. Dalším nutným datovým formátem pro chod programu byl PBF (Protocolbuffer Binary) formát z OpenStreetMap oblasti Jihomoravského kraje získaný stažením ze stránky *geofabrik.de* a následnou úpravou pomocí softwaru OSMConvert. Detailnější popis stažení a úpravy dat je v podkapitole 4.1.1.

Použité programy

Většina funkcionality klienta v této bakalářské práci byla tvořena převážně programovacím jazykem Python a jeho některými nezbytnými knihovnami. Pro vizualizaci pak převážně sloužily jazyky pro psaní webu. Serverovou část obstarával hlavní program OTP 2 a webový server Apache.

- **OpenTripPlanner 2** – Nejdůležitějším programem v této práci byl OpenTripPlanner Snapshot 2.1.0 verze pro vývojáře sloužící pro generování cest na základě dvou datových formátů: GTFS formát jízdních řádů a prostorový formát PBF z OpenStreetMap. Software je napsán v jazyku Java a je funkční na každém zařízení obsahující Javu verze 11 a vyšší.
- **PyScripter a Notepad++** – Pro psaní Python a webových kódů sloužila vývojová prostředí PyScripter verze 4.1.0 a Notepad++ verze 8.2.1.
- **OSMConvert** – Pomocí nástroje OSMConvert verze 0.8.8p byl ořezán PBF formát celé České republiky na oblast Jihomoravského kraje.
- **Apache server** – Klient byl nejprve spouštěn na lokálním zařízení pomocí softwaru MapServer for Windows (MS4W) s Apache serverem verze 2.4.46 s podporou CGI programování. V Apache serveru byla využívána funkce CGI a programovací jazyk Python verze 3.9.1.
- **Java SDK 11** – Pro chod programu OTP 2 musela být na operační systém nainstalována Java SE Development Kit verze 11.0.11.
- **Git a Maven** – Pro získání vývojářské verze OTP 2 sloužil kontrolní systém verzí Git a nástroj pro správu, řízení a automatizaci buildů Maven.
- **CommandLine** – Server na lokálním zařízení byl spouštěn pomocí příkazového řádku operačního systému Windows 10.
- **Python knihovny** – Programový kód klienta využíval různé Python moduly (knihovny). Modul CGI sloužil pro zpracování dotazu od uživatele z formulářového pole klienta. Requests vhodně odesílal dotaz do programu OTP 2 s parametry zpracované předchozím modulem. Modul JSON umožnoval jednoduše načítat a zpracovávat vygenerovanou JSON strukturu cesty. Knihovna Polyline dekódovala zašifrované souřadnice cesty do vhodné podoby. Pro zpracování časových hodnot sloužily moduly Datetime a Time.

Postup zpracování



Obr. 2.1 Schéma postupu zpracování bakalářské práce.

3 SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY

V této kapitole je vysvětlena problematika plánovačů cest z technického hlediska (z pohledu algoritmu). Dále jsou zde zmíněny nejpoužívanější vyhledávače v ČR i ve světě. Poté jsou uvedeny dva nejpoužívanější softwary pro tvorbu plánovačů a popsány jejich potřebné datové formáty. Nakonec jsou vybrány možnosti, jak k těmto softwarům nejlépe přistupovat.

3.1 Plánovače cest

Systémy na plánování cest se stávají čím dál více populární. Hlavními faktory, které ovlivňují tuto rostoucí popularitu, je poptávka uživatelů po spolehlivějších plánovačích, všudypřítomnost mobilních zařízení připojených na internet a vysoká dostupnost relevantních údajů ze senzorů, jako jsou například záznamy GPS (Global Positioning System) z autobusů či MHD (městská hromadná doprava). Ve své nejjednodušší podobě je plánování cest problémem nejkratší cesty, kdy je dáno grafické znázornění dopravní sítě, ve které se hledá nejkratší možná vzdálenost mezi počátečním a cílovým místem. Ovšem realističtější modelování, jako je zohlednění nejistoty, například při zpoždění spojů, může vést k variantám problémů, které jsou výpočetně náročné (Botea et al. 2013).

3.2 Multimodální plánovače

Multimodální plánovače umožňují zkombinovat více druhů dopravy, jako je například jízda autobusem nebo jízda na kole, do jedné cesty. Většina stávajících přístupů k multimodálnímu plánování cest fungují na základě deterministických předpokladů (Zografos a Androutsopoulos 2008). V reálném životě však klíčové informace potřebné při plánování cesty, například časy příjezdů autobusů, se vyznačují nejistotou. I malé odchylky v časech příjezdu autobusu mohou vést ke zmeškání spoje, což má velký negativní dopad na skutečný čas příjezdu do cíle. Plánovače by proto měly být schopny zvládnout takové neočekávané situace bez jakéhokoli narušení plánu (Botea et al. 2013).

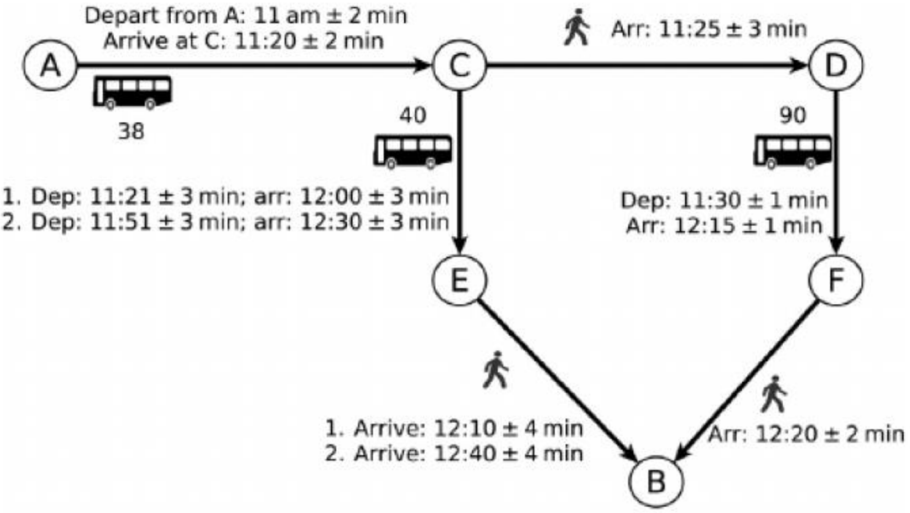
Multimodální plánovač je specializovaný vyhledávač, který slouží k nalezení optimální cesty mezi dvěma nebo více danými místy pomocí více než jednoho druhu dopravy. Ačkoli se první plánovače objevily již v 70. letech 20. století, teprve růst a běžná dostupnost internetu spolu s rozšířením geoprostorových dat a rozvojem informačních technologií vedl k rychlému rozvoji mnoha různých forem plánovačů cest, jako jsou například v Česku široce používané Google Maps, Mapy.cz nebo český vyhledávač jízdních řádů IDOS (Graczyk-Raczyńska et al. 2020).

3.3 Módy transportace

V dnešní době mohou lidé v rámci města cestovat mnoha různými způsoby. Například pěšky, sdílenými vozy, taxíky nebo různými druhy veřejné dopravy. Existuje dokonce ještě více možností dopravy při cestování na delší vzdálenosti nebo v neobvyklém prostředí. Může to být například cestování lodí, podzemní tunely, letadlem apod. V plánovačích představuje každý druh dopravy řadu různých faktorů a problémů, které je potřeba vzít v úvahu.

3.3.1 Veřejná doprava

Veřejná doprava zahrnuje různé dopravní prostředky, jako jsou autobusy, tramvaje, vlaky apod., avšak z technického hlediska (z pohledu algoritmu) je se všemi zacházeno stejně. V grafickém algoritmu lze na každý z nich pohlížet jako na uzly a hrany s přiřazenými váhami, které mohou popisovat vzdálenost nebo dobu trvání mezi nimi. Ovšem každý druh veřejné dopravy má svůj vlastní jízdní řád. Ačkoli se ve většině algoritmů považuje za pevnou, konstantní hodnotu, z mnoha různých důvodů to nemusí být vždy pravda. Některé systémy MaaS (Mobility as a Service) proto berou délku cesty jako stochastickou (náhodnou) proměnnou, která umožňuje analyzovat riziko zpoždění. Vyhledávání v multimodálních plánovačích, které tento problém neřeší, mohou vést k delším neplánovaným zpožděním. Když se jeden spoj zpozdí, tak další může být vynechán a teoreticky nejrychlejší cesta může být ve skutečnosti mnohem delší a problematická (Obr. 3.1). Nejlepší způsob, jak snížit, nebo dokonce eliminovat tuhle nejistotu trvání cesty u dopravních prostředků závislých na provozu, např. autobusů, je do plánovače zahrnout údaj o průměrné rychlosti konkrétního dopravního prostředku v daný den nebo hodinu. (Graczyk-Raczyńska et al. 2020).



Obr 3.1 Příklad, kdy plánovače zohledňující nejistotou mají výhodu oproti plánovačům s pevným časovým plánem. Nejrychlejší cesta z bodu A do bodu B vede přes bod E, ale vzhledem k nejistotě času příjezdu do bodu C, by mohlo být lepší jít přes body D a F (Botea et al. 2016).

3.3.2 Sdílení vozidel

S rostoucím počtem využívání sdílených aut, skútrů a kol, roste i počet potřeb začlenit je do multimodálních plánovačů. Využívají se především jako dopravní prostředek na konečných nebo počátečních úsecích cesty. U tohoto typu dopravy je potřeba vymodelovat dostupnost všech sdílených vozidel v oblasti. Za účelem zpřesnění dostupnosti je pro stanice s více vozidly, jako jsou například kola, nutné sestavit více modelů pro jednotlivé stanice (namísto vytvoření jednoho modelu pro každou stanici). Jedním z hlavních problémů systémů sdílení kol je nerovnoměrné rozdělení kol mezi

stanicemi v době dopravní špičky nebo v důsledku topografie oblasti. Tento problém lze řešit pomocí takzvané metody náhodné chůze. Je to metoda náhodného výběru, při níž se směr a vzdálenost mezi vybranými body určují pomocí náhodných čísel, které jsou nejčastěji vybírané z tabulek náhodných čísel (Tomaras et al. 2018). Až takto upravený model lze použít v multimodálním plánovači. Po vybrání nejkratší nebo nejoptimálnější trasy plánovač vyhledá blízké cyklistické stanice a navrhne uživateli ty stanice, které odpovídají požadavkům uživatele (Graczyk-Raczyńska et al. 2020).

3.3.3 Park-and-ride (Zaparkuj a jeď)

Park-and-ride je forma kombinované přepravy s návazností automobilové dopravy na veřejnou hromadnou dopravu. Pro osoby, které dojíždějí do zaměstnání a používají vlastní vozidlo, je zásadní, aby plánovač nenavrhoval pouze nejbližší cestu k destinaci, ale také destinaci s nejbližším parkovištěm. Výhodou je známá dostupnost parkovacích míst jednotlivých parkovišť (Graczyk-Raczyńska et al. 2020).

3.3.4 Chůze

Když lidé z nějakého důvodu nemohou cestovat veřejnou dopravou nebo se nemohou na konečné místo přepravit autem či jiným dopravním prostředkem, rozhodnou se cestovat pěšky. Ačkoli je to nejběžnější způsob cestování na počátečních a konečných úsecích cesty a zdá se to být jako jednoduchý výpočetní proces, některé aplikace pro plánování tras mají občas problémy s výběrem nejlepší pěší trasy. Například vyhledávač vypočítá cestu, která je kompletně nerealistická nebo z důsledku chybějících informacích o nově vybudovaném chodníku, navrhne cestu hromadnou dopravou, i když pěší cesta parkem je daleko pohodlnější a lepší (Graczyk-Raczyńska et al. 2020).

3.4 Plánovače v ČR a ve světě

V České republice i ve světě se nachází nespočet softwarů a aplikací, které umožňují naplánovat cestu. Tyto plánovače se na internetu vyskytují v nejrůznějších podobách a obsahují mnoho unikátních funkcí.

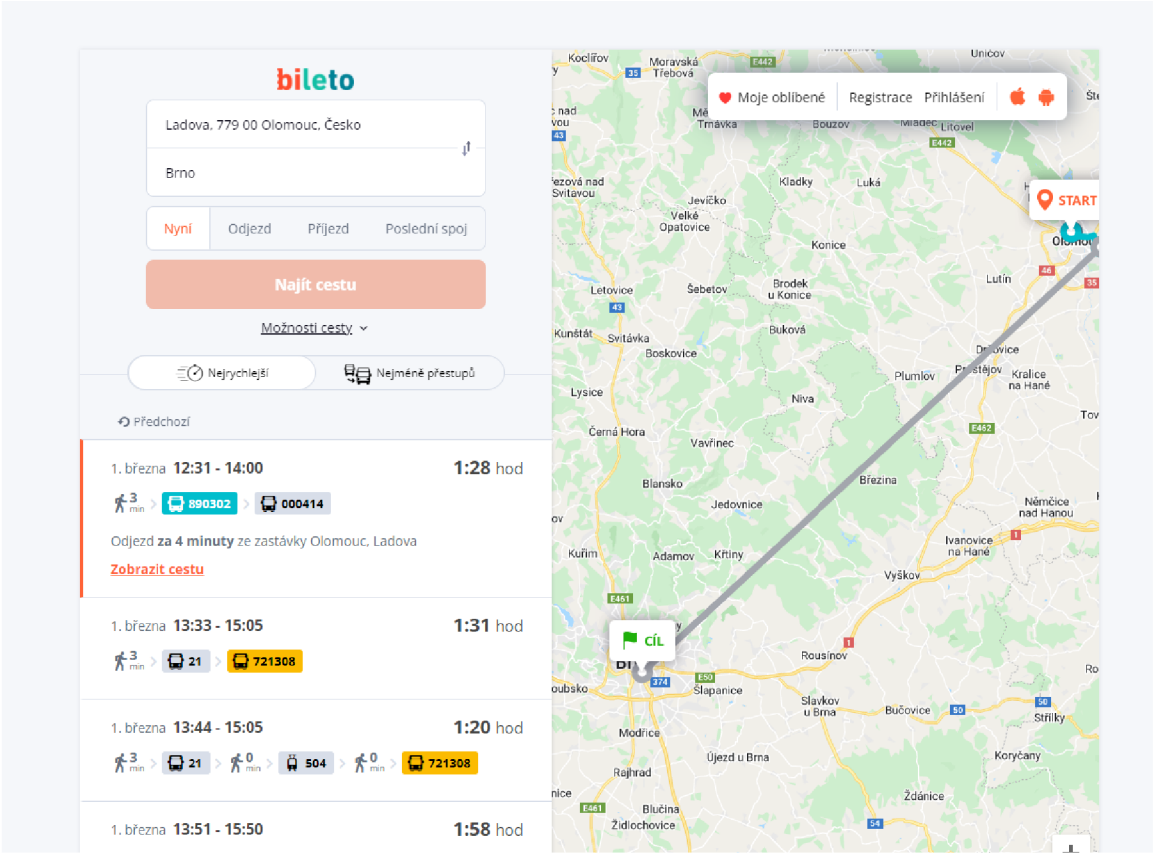
3.4.1 V ČR

V České republice je nejznámějším a nejvíce používaným plánovačem IDOS (CHAPS 2007). IDOS se zaměřuje na vyhledávání spojů autobusů, vlaků a integrovaných dopravních systémů ve větších městech. IDOS vyhledává optimální spoje po celém státu i na území Evropy. Proto je vhodný pro cestující, kteří si potřebují zjistit, kdy, odkud a kam jede konkrétní spoj.

Dalším plánovačem v ČR je webová aplikace Bileto (Bileto 2018). Na rozdíl od aplikace IDOS, tento plánovač nepotřebuje znát přesný název zastávky pro nalezení cesty, ale stačí, aby uživatel zadal jakoukoli adresu nebo místo v ČR a aplikace vyhledá nejoptimálnější cestu. Výsledná cesta je složená z jednotlivých kroků kombinující pěší přepravu s veřejnou dopravou. Uživatel má na výběr mezi nejrychlejší cestou a cestou s nejméně přestupy. Nechybí i výběr času příjezdu a odjezdu a také výběr jednotlivých druhů dopravy. Aplikace obsahuje přihlašovací okno, kdy po registraci a následném

přihlášení má uživatel možnost přidat si často hledané cesty do oblíbených a uložit si je pro budoucí opětovné použití (Obr. 3.2).

Dále je v ČR spojení veřejnou dopravou na mapovým portálu Mapy.cz. Aktuálně je pouze v testovací beta verzi. (Seznam.cz 1998). Aplikace má podobnou funkcionalitu jako Bileto, ale s mnohem větší informační hodnotu, jelikož uživatel může z výsledné cesty vyčíst více informací. Zobrazovaná cesta v mapě je více interaktivní a dynamická.



Obr. 3.2 Uživatelské rozhraní aplikace Bileto.

3.4.2 Ve světě

Ať už se jedná o multimodální plánovač nebo pouze o plánovač jízdních řádů (IDOS), téměř většina států světa má svůj vlastní se svým překladem a funkcemi. Na Slovensku je k dispozici hned několik plánovačů. Jedním z nich je mapový portál Mapa.sk od společnosti Zoznam (Zoznam nedatováno), který obsahuje funkci na plánování cest. Dalším slovenským webem na plánování cesty je webová aplikace Imhd.sk. V tomto případě se jedná o plánovač jízdních spojení na konkrétním území měst na Slovensku. Uživatel má na výběr z 20 slovenských měst (Mhd.sk 2000).

V anglickém Londýně existuje více veřejně dostupných multimodálních aplikací. Statutární orgán Transport for London zprostředkovává aplikaci, která dokáže naplánovat jakoukoli cestu na území celého Londýna (Greater London Authority 1999). Celonárodní britský plánovač poskytuje firma RAC (RAC 2019).

Jedním z nejznámějších amerických plánovačů je Inspirock od stejnojmenné společnosti. Aplikace nabízí mimo plánování cesty i tipy na zajímavé a populární místa,

které se nacházejí v okolí vygenerované cesty a také doporučuje hotely a penziony s možností přespání (Inspirock 2022).

3.5 OpenTripPlanner 2 vs Navitia

Pro tvorbu webových plánovačů se používají různé softwary a technologie. Nejčastěji používanými programy jsou typu open source, které jsou pro vývojáře nejvhodnější, protože si mohou volně zobrazit zdrojový kód, nebo ho dokonce upravit podle své potřeby, čímž docílí nejlepších výsledků pro výslednou aplikaci.

Nejrozšířenějšími softwary na plánování cesty jsou Navitia a OpenTripPlanner 2 (GITHUB 2020b). Zatímco Navitia se zaměřuje na to, aby byly jeho výsledné rámce (frameworky) co nejrozšířenější pro různé cestovatelské aplikace a webové stránky, OTP 2 se stále častěji používá při analýze sítí veřejné dopravy, její vizualizaci nebo mapování výsledků. Navitia se zaměřuje na individuální cesty cestujících a pomáhá jim vybírat z relevantních alternativ. Je speciálně navržen jako podklad, na kterém lze stavět různé informační aplikace. Navitia je složitý modulární systém, který lze použít v mnoha různých konfiguracích. Funguje tedy lépe, pokud je nainstalován a spouštěn na otevřené službě. Naopak OTP 2 je poměrně jednoduše spustitelný a lehce konfigurovatelný z osobního počítače na lokálním úložišti. Jeho architektura z jedné komponenty zjednodušuje instalaci a úpravu kódu.

Oba softwary využívají stejné rozhraní pro programování aplikací (API), REST API, které dokáže jednoduše vytvořit, číst, editovat nebo smazat informace ze serveru pomocí jednoduchých HTTP (Hypertext Transfer Protocol) dotazů. Programy také využívají podobné vstupní GTFS a OpenStreetMap datové formáty, které jsou pro správný chod nezbytné (Kisio Digital and Conveyal 2018).

GTFS

General Transit Feed Specification (GTFS) je datová specifikace, která umožňuje agenturám veřejné dopravy zveřejňovat svá tranzitní data ve formátu, který může být využíván širokou škálou softwarových aplikací. Datový formát GTFS dnes používají tisíce poskytovatelů veřejné dopravy. Formát GTFS je rozdělen na statickou složku a složku reálného času. Statická složka obsahuje jízdní řády, jízdné a geografické informace o dopravě. Reálná složka pak obsahuje předpovědi příjezdů, polohy vozidel a servisní zóny. Statická složka se skládá z několika textových souborů (.txt), které jsou společně zabaleny v jediném souboru ZIP. Každý soubor popisuje určitý aspekt informací o dopravě: zastávky, trasy, cesty, cena jízdného a jiné (MobilityData 2010).

Dvě společnosti v České republice volně poskytují data GTFS. Webový portál otevřených dat Opendata Praha (Opendata 2015) poskytuje tento formát, který obsahuje jízdní řády všech linek pražské integrované dopravy vždy na týden dopředu. Druhou společností, která GTFS data v České republice volně poskytuje je KORDIS JMK. Data jsou k dispozici od 16. srpna 2021 na portálu otevřených dat data.Brno, který spravuje magistrát Statutárního města Brno. Data jsou aktualizována vždy v neděli ve 12:00 hodin (Magistrát města Brna 2021).

OpenStreetMap

OpenStreetMap (OSM) je projekt tvořený komunitou uživatelů, kteří přidávají a udržují mapová data o silnicích, cestách, kavárnách, železničních stanicích a mnohém dalším po celém světě. OSM je volně editovatelná mapa celého světa, kterou vytvářejí dobrovolníci převážně od nuly a která je zveřejněna pod licencí open-source. K datům OSM se dá přistupovat více způsoby. Prvním způsobem zobrazení OSM-mapy a ostatních dat z OSM je za použití mapových knihoven jako je například Leaflet nebo OpenLayers. Dále se k OSM dá dostat přes různá API. Nejlepším přístupem pro různé vyhledávače a aplikace je třetí způsob, kdy OSM mapa je volně k dispozici v jednom velkém komprimovaném PBF souboru, který má velikost po stažení necelých 62 GB. Ovšem je poměrně zbytečné pracovat s takto velkým souborem, a proto existují různé softwary a webové aplikace, které umožňují z tohoto PBF souboru extrahovat jen vybranou oblast, a tím tak docílit daleko menší velikosti souboru (OpenStreetMap Wiki contributors 2020).

REST API

Rozhraní REST (známé také jako RESTful API) je aplikační programové rozhraní (API nebo webové API), které splňuje omezení architektonického stylu REST a umožňuje interakci s webovými službami RESTful. Zkratka REST znamená Representational State Transfer (přenos reprezentativního stavu) a vytvořil ji počítačový vědec Roy Fielding (Red Hat 2020).

Rest API funguje na základě dotazů od klienta, které jsou zpracovávané a odesílány ke koncovému uživatelovi. Tyhle dotazy jsou prostřednictvím HTTP protokolu doručovány v jednom z několika formátů. Jedná se o JSON (Javascript Object Notation), HTML (Hypertext Markup Language), XLT (Legacy Excel templates), Python, PHP nebo jako samostatný text (Red Hat 2020).

Php/Python

Python je vysokoúrovňový objektově orientovaný programovací jazyk. Má zabudované datové struktury v kombinaci s dynamickým typováním a vazbami, což z něj činí ideální volbu pro rychlý vývoj aplikací. Python také nabízí podporu modulů a balíčků, což umožňuje modularitu systému a opakované použití kódu. Vzhledem k vysoké výpočetní rychlosti a možnosti využívat knihovnu JSON, která jednoduše umí zpracovávat JSON soubory a data, je Python vysoce vyhledávaným programem pro tvorbu multimodálních aplikací (Campbell 2022).

Alternativním jazykem pro tvorbu plánovačů je PHP. PHP je zkratka pro PHP: hypertextový preprocesor. Jedná se o skriptovací jazyk na straně serveru. Používá se k vývoji dynamických webových stránek nebo webových aplikací. Tento jazyk lze snadno integrovat se všemi hlavními webovými servery ve všech hlavních operačních systémech (Campbell 2022).

4 MULTIMODÁLNÍ PLÁNOVAČ

V této kapitole je popsán postup tvorby multimodálního webového plánovače (klienta). Jeho hlavní funkcionalitu má na starost programovací jazyk Python a především open source program OpenTripPlanner 2.

V první řadě bylo zapotřebí nastudovat daný software a rozhodnout se, jakým způsobem bude klient vytvořen.

Na základě rešerše OTP 2 musel být program vhodně nakonfigurován a musela být zajištěna potřebná data GTFS a OSM. Pro ověření správnosti chodu programu slouží již vytvořený klient, který je zabudován uvnitř softwaru.

Po ověření funkčnosti programu a dat následovalo spuštění Apache serveru a připojení se na API OTP 2 z lokálního zařízení.

Pak následovala stěžejní část práce obsahující tvorbu klienta pomocí CGI programování. Nejprve bylo vytvořeno uživatelské rozhraní a následně byla aplikována různá funkcionalita pro vykreslování cesty v mapě a zobrazení detailního popisu jednotlivých úseků ve formě textu.

4.1 Zprovoznění a konfigurace OTP 2

OpenTripPlanner 2 je open source program, který funguje jako server. Jeho jádro je napsáno v programovacím jazyku Java, tudíž je spustitelný na každým operačním systému, který podporuje a má v sobě nainstalovaný JVM (Java Virtual Machine). Prvním krokem pro tvorbu klienta bylo tedy zprovoznit chod programu na základě návodu nacházejícího se na webu OTP 2 v sekci *Usage -> Basic Tutorial*.

4.1.1 Instalace OTP a dat

Nejprve bylo zapotřebí nainstalovat Javu na operační systém. Z oficiálního webu Oracle byl stažen instalační .exe soubor Java SE Development Kit verze 11.0.11 pro Windows 10 s 64bitovým procesorem. OTP 2 je kompatibilní s Javou verze 11 a vyšší, ovšem doporučená je právě verze 11. Následně byla provedena instalace a ověření funkčnosti.

Následovalo vytvoření souboru pro spuštění programu OTP 2. Program je napsán v Javě a distribuován v samostatně spustitelném JAR (Java ARchive) souboru. Tento soubor je volně stažitelný z repositáře na centrále Maven, nebo lze vytvořit naklonováním aktuální GitHub vývojářské verze. Na centrále Maven se nachází pouze verze v oficiálním vydání (nejnovější je 2.1.0), zatímco na GitHub jsou aktuální verze pro vývojáře. V této práci byla použita vývojářská GitHub verze. Pro získání souboru z GitHub bylo zapotřebí mít nainstalované tyto softwary:

- kontrolní systém verzí Git,
- Java Development Kit verze 11 (JDK 11),
- nástroj pro správu, řízení a automatizaci buildů Maven.

JDK 11 již byl nainstalován v předchozím kroku, tudíž proběhla instalace pouze programů Git a Maven. Dále byly v příkazovém řádku spuštěny tyhle příkazy, které naklonovaly aktuální Github repositář OTP 2 do adresáře *git*:

```
mkdir git
cd git
git clone git@github.com:opentripplanner/OpenTripPlanner.git
```

Poté bylo provedeno vytvoření JAR souboru pomocí těchto příkazů:

```
cd OpenTripPlanner
mvn clean package
```

Tyhle příkazy docílily toho, že v adresáři *git/OpenTripPlanner2/target* byl vygenerován spustitelný JAR soubor s názvem *otp-2.1.0-SNAPSHOT-shaded.jar*. Soubor byl pro snadnější přístup překopírován do nově vytvořené složky na disku.

Pro správný chod programu bylo nutné získat dvě datové sady: data ve formátu GTFS, které obsahují informace o jízdních řádech a prostorová data stejné oblasti ve formátu PBF z OpenStreetMap. GTFS data poskytuje od 16. srpna 2021 společnost KORDIS JMK a pokrývají celý Jihomoravský kraj. Data jsou volně k dispozici na webu *data.brno.cz* a jsou aktualizována vždy v neděli ve 12:00 hodin. Data byla stažena a ve výchozí podobě ZIP souboru byla přemístěna do adresáře s JAR souborem OTP 2.

OTP 2 také potřebuje PBF formát OSM stejné oblasti jako GTFS data, což v případě této práce je celý Jihomoravský kraj. Pro extrakci této oblasti byl použit nástroj *OSMConvert*. Nástroj potřebuje vstupní data, ze kterých může vyextrahovat určitou oblast, proto byla ze stránky *download.geofabrik.de* stáhnuta data PBF pro celou Českou republiku. Dále nástroj potřebuje znát souřadnice ohraničujícího rámečku (bounding box) znamenající oblast pro extrakci. Souřadnice Jihomoravského kraje byly získány pomocí webové aplikace *boundingbox.klokantech.com*. Osmconvert byl spuštěn a byl vytvořen PBF soubor, který byl přesunut do adresáře k JAR souboru OTP 2.

4.1.2 Spuštění serveru

Po získání potřebných dat přišlo na řadu spuštění OTP 2 serveru. OTP 2 umožňuje spuštění serveru na lokálním zařízení dvěma způsoby. Prvním způsobem je vytvoření serveru okamžitě jedním příkazem bez nutnosti uložení souborů na disk. Příkaz má dva důležité parametry *build* a *serve* a celý zápis vypadá takto:

```
java -Xmx4G -jar otp-2.1.0-SNAPSHOT-shaded.jar --build --serve C:/home/username/otp
```

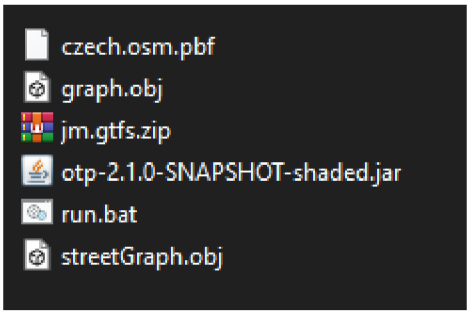
Druhý parametr příkazu *-Xmx4G* znamená alokaci operační paměti pro server. V tomto případě to jsou 4 GB. Poslední pozice je pak cesta k adresáři s JAR souborem. Hlavní nevýhodou tohohle způsobu je dlouhá načítací doba serveru. Po přibližně třech minutách načítání je v příkazovém řádku zobrazena zpráva *„Grizzly server running“*, což znamená úspěšný start serveru.

Druhým, lepším způsobem, jak spouštět server, je vytvoření graphs objektů a jejich uložení na disk. Graphs eliminují dlouhou načítací dobu serveru při opakovaném spouštění. Použitím parametrů *build* a *save* je v adresáři s JAR souborem vytvořen nový soubor *graph.obj*. Následujícími nepovinnými parametry jsou *buildStreet* a *loadStreet,* pomocí nichž je vytvořen objektový soubor *streetGraph.obj*. Poté ve spouštěcím příkazu stačí nahradit parametry za příkaz *load* a server bude načten během několika sekund. V této práci byl využíván druhý způsob spouštění a celý příkaz byl zabalen do spouštěcího souboru *run.bat*. Obsah souboru byl v tomto zápisu:

```
java -Xmx4G -jar otp-2.1.0-SNAPSHOT-shaded.jar --load C:\Users\twina\otp\otpdev3
```

Adresář (Obr. 4.1) otpdev3 v této chvíli obsahoval 6 souborů:

- JAR soubor OTP 2,
- GTFS data JM kraje,
- PBF data,
- graphs.obj,
- nepovinný soubor streetGraph.obj (finální verze tento soubor neobsahuje),
- spouštěcí run.bat.



Obr. 4.1 Adresář se soubory pro správný chod OTP.

4.1.3 Konfigurační soubory

Chod a funkčnost serveru OTP může být konfigurován přes tři dodatečné JSON soubory, které se ukládají do stejného adresáře ke spouštěcímu JAR souboru. Soubory musí mít specifický název:

- build-config.json,
- router-config.json,
- otp-config.json.

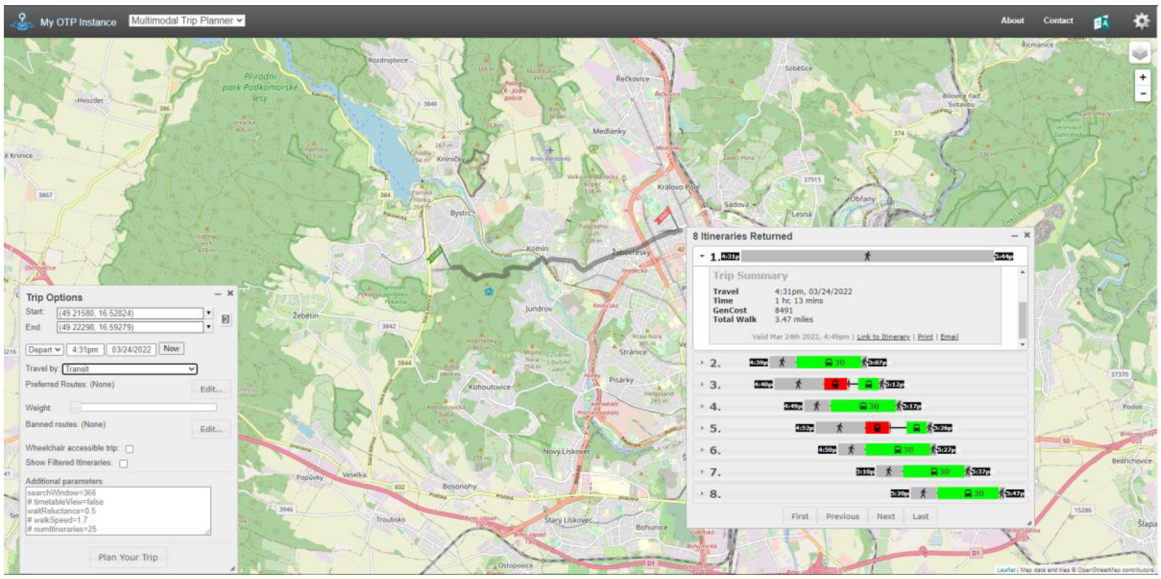
Každý z nich má jinou funkcionalitu a do každého jsou zapisovány jiné parametry. *Build-config.json* obsahuje volby a parametry, které ovlivňují proces sestavování graphů při startu serveru a nejde je později měnit při již zapnutém serveru. Jedná se o nastavení, například zda zahrnout všechny vstupní soubory ze skenovaného adresáře nebo nastavení maximálního počtu uzlů pro danou oblast. Parametry v *Router-config.json* zohledňují samostatný výpočet cesty a mohou být změněny i při chodu serveru. Například průměrná rychlost chůze, průměrná rychlost na kole nebo maximální počet přestupů. Pomocí souboru *otp-config.json* mohou být povoleny nebo zakázána podporovaná rozhraní API a různá programová rozšíření.

Pro tuhle práci nebyl žádný z těchto konfiguračních souborů vytvořen a použit, protože se jedná pouze o nepovinné soubory a jejich absence zapříčiní nastavení všech konfiguračních hodnot na defaultní, což bylo při této tvorbě klienta dostačující.

4.1.4 OTP 2 testovací klient

Po zapnutí souboru *run.bat* je server během několika sekund načten a na adrese *http://localhost:8080/* je k dispozici testovací JS (JavaScript) klient aplikace. Jedná se

o interaktivní Leaflet mapu, která slouží pro ověření funkčnosti a správnosti OTP 2 serveru a vstupních dat. Klient obsahuje formulářové okno, do kterého uživatel zadává různé nastavení cesty. Jsou to souřadnice počátečního a koncového bodu cesty, čas odjezdu a příjezdu, typ přepravy, preferované cesty, zakázané cesty a možnost zapnutí nebo vypnutí bezbariérového přístupu (Obr. 4.2).



Obr. 4.2 OTP 2 testovací klient.

Klient generuje cesty kliknutím pravým tlačítkem myši do mapy a vybráním počátečního a koncového bodu společně s nastavením cesty. Vzápětí je zobrazena nově vygenerovaná cesta ve formě linie. Současně je také zobrazeno nové dialogové okno obsahující sumarizační statistiky cesty a detailní popisy kroků cesty. Některé prvky uživatelského rozhraní jsou inspirací pro funkcionalitu finální verze klienta, který je vytvořen v rámci této práce. Například výběr bodů v mapě pomocí pravého tlačítka myši.

Hlavní nevýhodou je nemožnost ovládat klient, když je webový prohlížeč v českém jazyce. Tento problém byl eliminován přepnutím jazyka do anglického. Další nevýhodou je nevhodné řešení výpisu statistik a kroků cest. Klient je tak pro uživatele nepoužitelný a slouží spíše pro vývojáře na ověření správnosti dat a chodu serveru.

4.1.5 Prvotní napojení na API

Po konfiguraci a zprovoznění serveru OTP 2 přišlo na řadu připojení na jeho aplikační rozhraní (API). API OTP 2 funguje na základě REST webových dotazů, což umožňuje přístup mnoha způsoby. Na základě rešerše byl pro uložení dotazů a zpracování výsledků vybrán programovací jazyk Python.

V první řadě bylo zapotřebí zjistit, jak se připojit na API pomocí Pythonu. Na základě dokumentace OTP 2 (OTP 2 Developers nedatováno) a pomocí Python knihoven *Requests* a *JSON* byl napsán krátký testovací skript (Obr 4.3), který pomocí parametrů generoval JSON soubor obsahující veškeré informace o výsledné cestě. Skript byl napsán pomocí Python verze 3.8 v programu PyScripter.

```python
import requests
import json

para = {"fromPlace": "49.23320, 16.57638", "toPlace": "49.18260, 16.64744", "mode": "TRANSIT,WALK"}

r = requests.get("http://localhost:8080/otp/routers/{ignoreRouterId}/plan", params=para)
data = json.loads(r.text)

print(json.dumps(data, indent=2))
```

Obr. 4.3 Testovací skript pro připojení na API a generaci JSON souboru obsahující informace o cestě.

Do proměnné *para* byly přiřazeny parametry reprezentující dotaz z webového klienta. Jednalo se o souřadnice počátečního a koncového bodu cesty a mód přepravy (v tomto případě *TRANSIT,WALK* znamená přepravu veřejnou dopravou s chůzí). Veškeré možné parametry jsou k dispozici v dokumentaci OTP 2 (OTP 2 Developers nedatováno). Stěžejní částí tohoto skriptu byl řádek s proměnou *r* obsahující dotaz na adresu: *http://localhost:8080/otp/routers/{ignoreRouterId}/plan*

společně s předchozími parametry. Tento řádek odesílal dotaz s parametry do spuštěného programu OTP 2. Na základě toho byla vygenerována JSON struktura obsahující veškeré informace o nově vzniklé cestě. Struktura byla uložena do proměnné a vypsána v přehledném tvaru (Obr. 4.4). Tímto způsobem byly později informace o nově vzniklé cestě zpracovávány a zobrazovány v klientovi.


```
2381            },
2382            {
2383              "distance": 864.5459999999999,
2384              "relativeDirection": "LEFT",
2385              "streetName": "Pod Leskounem",
2386              "absoluteDirection": "EAST",
2387              "stayOn": false,
2388              "area": false,
2389              "bogusName": false,
2390              "lon": 16.3475785,
2391              "lat": 49.012923900000004,
2392              "elevation": ""
2393            },
2394            {
2395              "distance": 1188.106,
2396              "relativeDirection": "CONTINUE",
2397              "streetName": "Znojemsk\u00e1 vina\u0159sk\u00e1 stezka, Krumlovsko - Jevi\u016lovicko",
2398              "absoluteDirection": "EAST",
2399              "stayOn": false,
2400              "area": false,
2401              "bogusName": false,
2402              "lon": 16.358389000000003,
2403              "lat": 49.014315,
2404              "elevation": ""
2405            },
2406            {
2407              "distance": 1021.958,
2408              "relativeDirection": "HARD_RIGHT",
2409              "streetName": "track",
2410              "absoluteDirection": "SOUTHEAST",
2411              "stayOn": false,
2412              "area": false,
2413              "bogusName": true,
2414              "lon": 16.372288100000002,
2415              "lat": 49.0176327,
2416              "elevation": ""
2417            }
2418          ],
2419          "rentedBike": false,
```


Obr. 4.4 Ukázka části JSON souboru z API OTP 2 vygenerovaného pomocí Python skriptu.

4.1.6 Python CGI

V této práci je předchozí skript aplikován do programového kódu klienta a při každém odeslání dotazu uživatelem je nastavení cesty uloženo do parametrů. Z vygenerovaného JSON souboru jsou postupně zpracovávány výsledky ve formě textu a linie v mapě. Pro odeslání dotazů a jejich uložení do Python proměnných byla využita knihovna CGI, která funguje stejný způsobem jako populární jazyk PHP. Common Gateway Interface neboli CGI je sada standardů, které definují způsob výměny informací mezi webovým serverem a vlastním skriptem (Tutorials Point 2006). Knihovna pro správný chod potřebuje spuštěný webový server, který podporuje CGI programování. Proto byl stažen a nainstalován softwarový webový HTTP server Apache, který je součástí balíčku MS4W (GatewayGeo 2008).

CGI programování a konkrétní Python knihovna CGI umožňuje kombinovat programovací jazyk Python společně s webovými jazyky jako je například HTML, CSS nebo JavaScript. Pomocí této možnosti byl vytvořen klient, který na základě uživatelského HTML formuláře odesílá dotazy ve formě Python proměnných. Proměnné vstupují jako parametry pro vygenerování JSON souboru ze spuštěného OTP 2 programu a následně jsou informace z JSONu zpětně zobrazovány v klientovi ve formě textu a linie v mapě. Ve finále je klient funkční jen na zařízení, na kterém je spuštěn software OTP 2 a server s podporou CGI.

Klient byl vytvářen na lokálním zařízení, proto výchozí adresář pro spouštění Python souborů pomocí CGI byl umístěn při defaultní instalaci MS4W na adrese *C:\ms4w\Apache\cgi-bin*. V tomto adresáři byl vytvořen spouštěcí Python soubor klienta *index.py*, do kterého později byl psán veškerý kód plánovače. Na prvním řádku musela být definovaná cesta k Python kompilátoru serveru MS4W. Do prvního řádku byl tedy zapsán tento kód: `#!"C:\ms4w\Python\python.exe"`.

Následovalo ověření funkčnosti CGI. Do souboru byl vepsán jednoduchý skript pro vypsání testovací věty „Hello world!“ (Obr. 4.5). Následně byl spuštěn Apache server a do webového prohlížeče byla zapsána adresa *http://localhost/cgi-bin/index.py*, pomocí které byl script automaticky spuštěn a byla zobrazena jeho testovací věta na webové stránce. Tímto způsobem byl celý klient naprogramován a spouštěn.

Pro připojení CSS (Cascading Style Sheets) k spouštěcímu souboru index.py musel být CSS soubor umístěn do složky *C:\ms4w\Apache\htdocs\bakacss* a v hlavičce HTML kódu souboru *index.py* nalinkována relativní cesta *../bakacss/style.css*.

Hlavní nevýhodou CGI programování byla nemožnost rozdělit hlavní soubor klienta na více menších a tím zvětšit přehlednost kódu. Proto veškerý kód musel být zapsán v souboru *index.py*.

```
#!"C:\ms4w\Python\python.exe"


print("Content-Type: text/html")
print()

print("Hello world!")
```

Obr. 4.5 Script pro vypsání textu „Hello world!“.

4.2 Tvorba klienta

V první fázi tvorby klienta bylo navrženo uživatelské rozhraní. Poté byla postupně přidávána nezbytná funkcionalita a klient byl nakonec vhodně otestován.

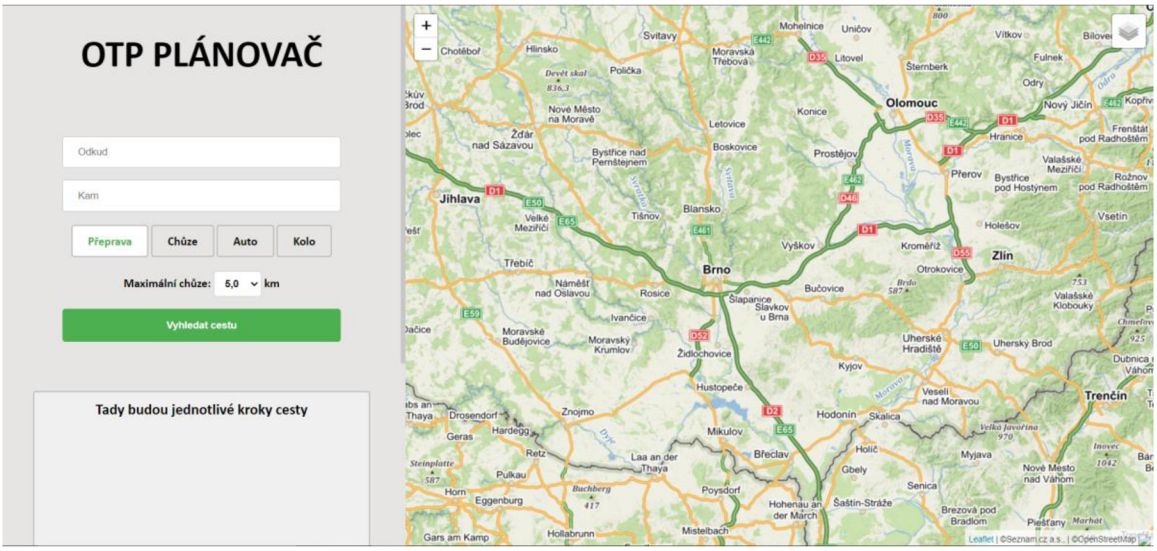
4.2.1 Uživatelské rozhraní

Do spouštěcího souboru *index.py* byla přidána HTML struktura (například meta tagy, title, body), která byla vypisována pomocí víceřádkového „print“ zápisu: `(''' kód ''')`. Stránka byla rozdělena na dva oddíly (divy) ležící vedle sebe. Levý oddíl byl nastaven na šířku 35 % stránky a pravý na zbylých 65 %, tudíž oba oddíly dohromady vyplňovaly celkovou plochu.

V levém oddílu byl vytvořen provizorní uživatelský formulář. Jeho styl byl vytvořen za pomoci návodu na stránce W3Schools (W3Schools 1999). Formulář obsahoval dvě textové pole pro zadávání počátečních a konečných souřadnic bodů ve formátu WGS 84, čtyři seskupené „radio“ tlačítka pro vybrání typu přepravy, výběrové pole pro maximální možnou chůzi v kilometrech a potvrzovací tlačítko na odeslání formuláře. Výběrové pole pro maximální chůzi bylo později nahrazeno za výběr datumu a času odjezdů nebo příjezdů. Pod formulářem byl nachystán oddíl na zobrazování jednotlivých kroků cesty. Na konec byla přidána možnost levý div posunovat nahoru a dolů pomocí posunovacího baru nebo kolečka na myši.

Do pravé části stránky byla vložena interaktivní Leaflet mapa (Agafonkin 2010), která zabírala celkovou plochu oddílu. Mapa byla vložena přes externí CSS a JS soubory a do HTML struktury byla vepsán základní kód, který mapu na stránce zobrazoval. Mapa byla defaultně vycentrována a přiblížena na zájmovou oblast (Jihomoravský kraj). Na závěr byla přidána možnost měnit mezi třemi podkladovými mapy pomocí ikony v pravém horním rohu.

Takovéto navržené uživatelské rozhraní (Obr. 4.6) bylo podkladem pro funkcionalitu klienta a bylo různě vylepšováno a měněno v závislosti na potřebách a možnostech následného programování a tvorby.



Obr. 4.6 Návrh uživatelského rozhraní klienta.

4.2.2 Zpracování dotazu a prvotní vykreslení cesty

Klient funguje na základě uživatelského formuláře, který odesílá uživatelské dotazy s hodnotami (odkud, kam, typ přepravy atd.). Proto bylo zapotřebí do kódu klienta naimportovat knihovnu CGI, která umí hodnoty z HTML formuláře vhodně zpracovat. Pomocí vepsání modulu `import cgi` na začátek kódu klienta byla knihovna aktivována. Knihovna nemusela být instalována, protože již byla součástí Python adresáře v MS4W (MapServer for Windows). Následovalo vytvoření proměnných, které uchovávaly hodnoty z formuláře (Obr. 4.7).

```
form = cgi.FieldStorage()

odkudd = form.getvalue("odkud") #Odkud
kamm = form.getvalue("kam") #Kam
modd = form.getvalue("radioo") #Mód cesty
```

Obr. 4.7 Uložení hodnot z formuláře do proměnných.

Po odeslání formuláře byly do proměnných uloženy hodnoty podle názvu elementů formuláře (Například do proměnné *odkudd* byla uložena hodnota zapsaná do textového pole *odkud*). Pro ověření funkčnosti bylo pracováno jen s formulářovými elementy odkud, kam a typ (mód) přepravy. Následovalo uložení těchto proměnných do parametrů pro vygenerování JSON souboru pomocí OTP 2. Jak je zmíněno v kapitole 4.1.5, pro připojení na API OTP 2 bylo nutné aktivovat python knihovny *Requests* a *JSON*. Skript z kapitoly byl předělán do podoby, kdy místo „na pevno“ daných hodnot byly aplikované dynamicky se měnící hodnoty proměnných na základě dotazu z formuláře (Obr. 4.8). V tomto momentě byl po odeslání dotazu vygenerován JSON soubor obsahující veškeré informace o cestě na základě zapsaných hodnot ve formuláři. Informace z JSON souboru byly uloženy do proměnné *data*.

```
para = {"fromPlace": odkudd, "toPlace": kamm, "mode": modd} #Parametry

r = requests.get("http://localhost:8080/otp/routers/{ignoreRouterId}/plan", params=para)
data = json.loads(r.text)
```

Obr. 4.7 Uložení proměnných do parametrů.

Pro prvotní vykreslení cesty v mapě a ověření funkčnosti tohoto přístupu, bylo zapotřebí nalézt vygenerované souřadnice cesty v JSON souboru. Vygenerovaný JSON soubor je rozdělen do tzv. itineraries znamenající jednotlivé cesty. Jeden JSON může mít i více itineraries, ale v tomto ověřovacím případě bylo pracováno pouze s první nalezenou cestou. Každá cesta má údaje o souřadnicích na pozici *legs* -> *legGeometry* -> *points*. Ovšem jak lze vidět na obrázku 4.8, souřadnice jsou pro zmenšení velikosti souboru zakódovány pomocí formátu Algoritmu kódovaných polylinií od firmy Google (Google 2022).

```
},
"legGeometry": {
  "points": "{atkHcqeeBtBdEdD`GdB|CDH|AnCl@dBT^v@b@v@h@p@x@~@|@NrB\\jBNV`@p@fA`BhA|ArAbBp@~@~@g@DTXxADbA?RFhCBXHXv@PZLNJ\\tBV`CNhA@P@F@JD^F??F",
  "length": 40
},
```

Obr. 4.8 Ukázka zakódovaných souřadnic v JSON souboru.

Pro dekódování souřadnic musela být použita python knihovna *Polyline*. Jelikož knihovna nebyla součástí instalace MS4W, musela být dodatečně nainstalována pomocí pip instalačního balíčku (The pip developers 2008). Do příkazového řádku byl zadán tento příkaz: `pip install --target= C:\ms4w\Python\Lib polyline` pomocí kterého byla knihovna nainstalována do adresáře s knihovnami v MS4W. Následně mohla být aktivována pomocí modulu *import*.

Souřadnice byly uloženy do proměnné a následně pomocí knihovny dekódované. Jelikož dekódované souřadnice mají jiný formát závorek, než je podporovaný pro vykreslení cesty v Leaflet mapě, musely být kulaté závorky nahrazeny za hranaté (Obr. 4.9).

```
cesta = (data['plan']['itineraries'][0]['legs'][0]['legGeometry']['points'])

souradnice = str(polyline.decode(cesta))
s1 = souradnice.replace("(","[") #Nahrazení kulatých závorek za hranaté
s1 = s1.replace(")","]")
```

Obr. 4.9 Dekódování souřadnic a nahrazení kulatých závorek za hranaté.

Na závěr byla proměnná se souřadnicemi aplikována do proměnné *polyline* vykreslující linii v mapě. Zároveň s vykreslením linie byla mapa automaticky přiblížena na celou cestu (Obr. 4.10).

```
if s1:   #Zobrazení cesty v mapě
    print("var latlngs =" + s1 +";")
    print("var polyline = L.polyline(latlngs, {color: 'red', weight: 5}).addTo(map);")
    print("map.fitBounds(polyline.getBounds());") #Zoom na polyline
```

Obr. 4.10 Zobrazení a zoom na cestu v mapě.

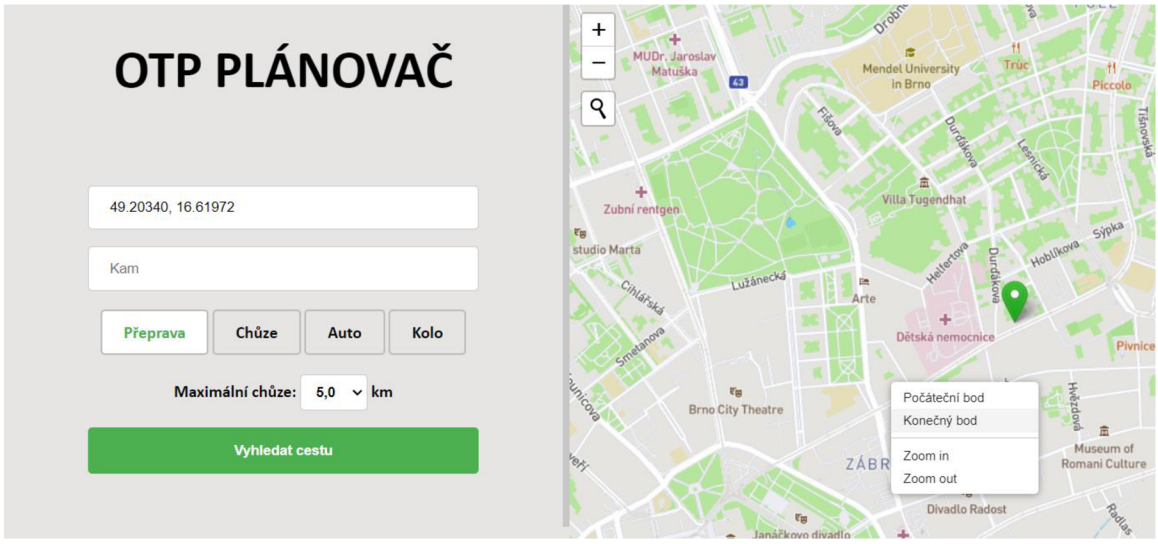
Po zadání souřadnic, vybrání typu přepravy a odeslání dotazu byla v mapě zobrazená červená linie znázorňující výslednou cestu. Ovšem takto vykreslená cesta nebyla ve vhodné podobě a byla později upravena. Sloužila zatím pouze pro ověření funkčnosti.

4.2.3 Zadávání souřadnic a geolocator

V této chvíli byly souřadnice zadávány překopírováním nebo zapsáním do formuláře ručně. Tento způsob byl ovšem pro uživatele nevhodný a časově náročný, proto byl aplikován způsob zadávání souřadnic přes výběr bodů v mapě. Přes Leaflet plugin Context menu (GITHUB 2019) byla přidána možnost kliknutím pravým tlačítkem myši do mapy zobrazit kontextovou nabídku. Plugin musel být nejprve nalinkován do hlavičky HTML struktury a poté správně zapsán do Leaflet proměnné mapy. Nabídka byla z výchozího stylu upravena do podoby zobrazující tyto čtyři prvky:

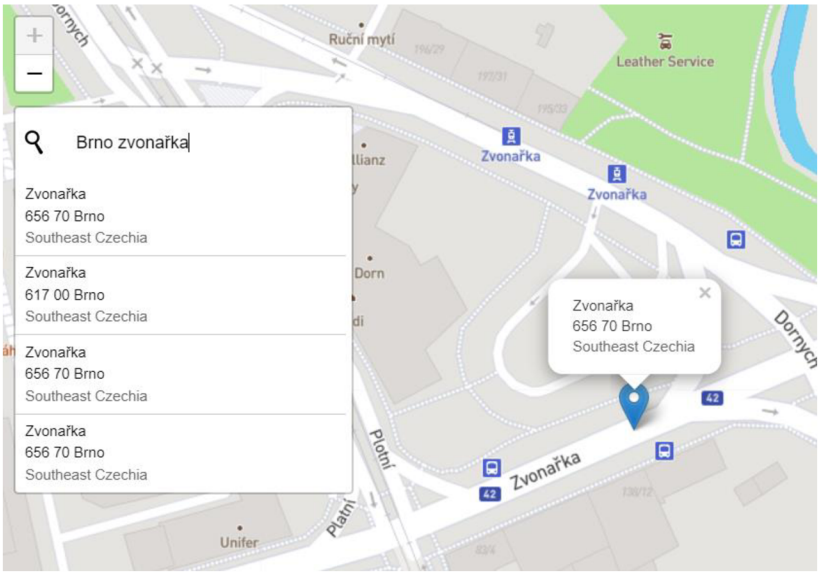
- Počáteční bod,
- Konečný bod,
- Zoom in,
- Zoom out.

Po vybrání možnosti *Zoom in* nebo *Zoom out* byla mapa na kliknuté místo přiblížena nebo oddálena. Pro možnosti počátečního a konečného bodu byla vytvořena JavaScriptová funkce. Funkce byla upravena do podoby, kdy při výběru jedné možnosti byl na vybraném místě vytvořen bod a jeho souřadnice byly automaticky přeneseny do textového pole formuláře *Odkud* nebo *Kam* (Obr. 4.11). Vytvořené body byly volně uchopitelné a vždy po přemístění bodu byly souřadnice aktualizované. Nakonec byl počáteční a konečný bod rozlišen barvami (počáteční zelený a konečný červený) a bylo ošetřeno mizení obou bodů při vygenerování cesty. Zadávání souřadnic již nemuselo být prováděno ručním překopírováním.



Obr. 4.11 Kontextová nabídka pro vytvoření bodů v mapě a přenesení jejich souřadnic do formuláře.

Pro ještě větší usnadnění práce s klientem byla přidána možnost vyhledávat adresní místa pomocí tlačítka lupy v mapě. Tuhle funkci umožňuje Leaflet plugin Control Geocoder (GITHUB 2020a), který ve výchozí podobě vyhledává adresy na základě vyhledávače Nominatim od OpenStreetMap. Po kliknutí na tlačítko lupy v mapě vlevo nahoře je zobrazeno textové pole pro zadání jakéhokoli adresního místa. Výběrem adresy je mapa přiblížena na danou oblast a na konkrétní adrese je automaticky vytvořen bod (Obr. 4.12). V pozdějších fázi tvorby klienta byl podobný vyhledávač adres aplikován i na textové pole formulářů pro zadávání souřadnic.



Obr. 4.12 Leaflet plugin Control Geocoder.

4.2.4 Obarvení cest a generování kroků cesty

Formulářový element pro výběr typu dopravy obsahuje čtyři možnosti: Přeprava, chůze, auto a kolo. OTP 2 u typu přepravy chůze, auta a kola generuje právě jednu nejrychlejší možnou cestu, která ve většině případů je tvořena pouze jedním krokem cesty. Ovšem u možnosti přeprava (přeprava veřejnou dopravou) generuje hned několik možných cest, které jsou složeny z více kroků. Například začátek cesty je pěší a následuje přestup na autobus nebo vlak. Proto zobrazování cesty veřejnou dopravou muselo být upraveno do přehledné a lepší podoby.

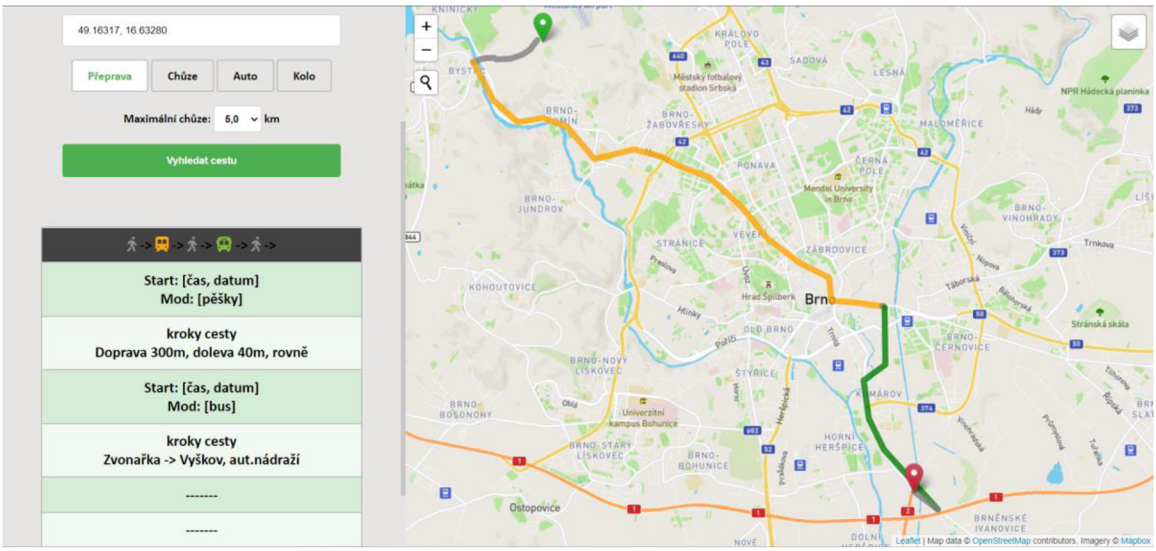
Nejprve v kódu klienta byla ošetřena detekce módu přepravy a následné rozdělení souřadnic do více proměnných znázorňující jednotlivé kroky cesty. Byly vytvořeny pomocné slovníky (dictionary), do kterých byly uloženy souřadnice kroků cesty (Obr. 4.13). Poté pomocí *for* cyklu byly dynamicky vypisovány Leaflet proměnné obsahující linie kroků s odpovídajícími souřadnicemi.

```
#cyklus, který do dictionary uloží souřadnice jednotlivých kroků
for iti in data['plan']['itineraries'][0]['legs']:
    dict_souradnice["promenna%s" %pocet_souradnice] = str(polyline.decode(iti['legGeometry']['points']))
    dict_souradnice["promenna%s" %pocet_souradnice] = dict_souradnice["promenna%s" %pocet_souradnice].replace("(","[")
    dict_souradnice["promenna%s" %pocet_souradnice] = dict_souradnice["promenna%s" %pocet_souradnice].replace(")","]")
    pocet_souradnice = pocet_souradnice + 1 #pomocna prommena pro pocet
```

Obr. 4.13 Generování slovníků se souřadnicemi kroků cesty.

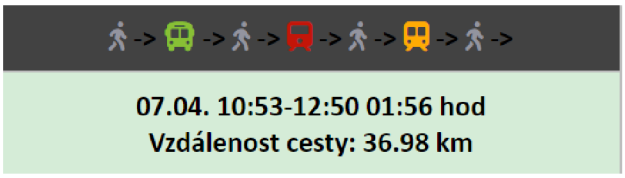
Ve výsledku v mapě bylo zobrazeno několik na sebe navazujících linií spojených v celkovou výslednou cestu. Tento nový způsob zobrazování cesty umožnil jednotlivé linie obarvit podle typu přepravy nebo dopravního prostředku (Obr. 4.14). Po zjištění nejčastějších možných typů přepravy pro Jihomoravský kraj byly vybrány tyto barvy: šedá pro chůzi, zelená pro autobus, oranžová pro tramvaj, červená pro vlak, žlutá pro auto, modrá pro kolo a černá pro případ možnosti nespecifikovaného typu přepravy.

Následně pod formulářovým oknem byl vytvořen oddíl obsahující dynamicky se měnící znázornění jednotlivých kroků cesty pomocí Font Awesome ikon (Fonticons 2022). Pod tento oddíl byl přichystán prostor pro výpis statistik a kroků cesty v podobě textu (Obr. 4.14). Klient v tomto momentě vhodně a přehledně vykresloval cestu v mapě, ovšem bylo zapotřebí přidat textovou informaci v podobě statistik a detailů kroků cesty.



Obr. 4.14 Náhled vykreslené a obarvené cesty v klientovi.

Pod ikonami kroků cesty bylo vytvořeno pole pro celkové shrnutí cesty v podobě základních údajů. Jednalo se o datum, čas odjezdu a příjezdu, celkový čas cesty a vzdálenost cesty v kilometrech. Pro vypsání časů musely být do kódu naimportovány knihovny *Datetime* a *Time*, jelikož údaj o čase ve vygenerovaném JSON souboru OTP 2 byl ve formátu časové značky Unix (Dan's Tools 2014). Čas byl konvertován do vhodné podoby a společně se vzdáleností cesty vypsán v přichystaném divu (Obr. 4.15).



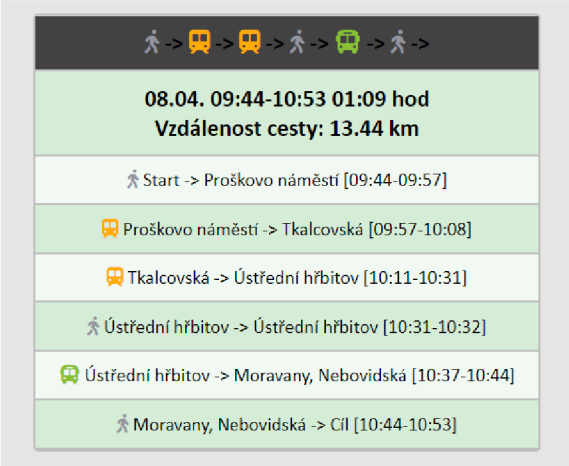
Obr. 4.15 Vypsání základních statistik cesty.

Pod sekcí se statistikami bylo vytvořeno automatické generování polí (divů) pro výpis kroků cesty ve formě textu. Při každém odeslání dotazu a zobrazení cesty v mapě byly vždy vygenerovány pole na základě počtu jednotlivých kroků cesty. Nejdůležitější částí kódu pro tuto operaci byl *for* cyklus:

```
for iti in data['plan']['itineraries'][0]['legs']:
```

který vykonával operace na základě počtu kroků cesty (legs). Jednotlivé divy byly pro lepší přehlednost odlišeny barevně podle sudého nebo lichého pořadí. Pomocí JS knihovny jQuery (The jQuery Foundation 2006) bylo umožněné každý div rozkliknout a zobrazit novou sekci, do které byly později přidány detailnější informace o daném úseku cesty. Do výchozích rozklikávacích divů byly přidány názvy počátečního a konečného místa a doba trvání úseku (Obr. 4.16). Sekce s detailnějšími informacemi musely být generovány dvěma způsoby, jelikož zápis informací v generovaném JSON souboru byl odlišný pro veřejnou dopravu (autobus, tramvaj, vlak) od ostatních typů „jednokrokových" typů přepravy (auto, kolo, chůze). Detailnější informace veřejné dopravy vždy obsahovaly: Název počáteční a konečné zastávky, jejich časy odjezdu a příjezdu, hlavní zastávky spoje a název společnosti s odkazem na webové stránky. Do úseků chůze, autem a na kole byla aplikována podrobná textová navigace ukazující směr, název ulice a vzdálenost kudy se vydat. Pokud se nejednalo o název ulice, bylo

vypsáno v anglickém jazyce, o jaký typ vozovky/chodníku se jedná. Později byly tyto názvy přeloženy do češtiny. Nakonec pro ještě větší přehlednost byly k informacím přidány ikony určující typ přepravy.

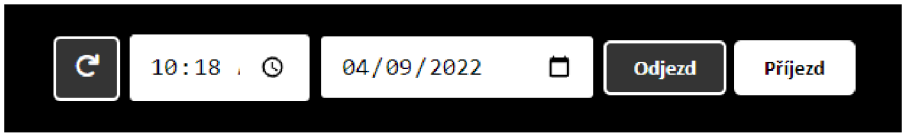


Obr. 4.16 Kroky cesty v dynamicky generovaných divech.

4.2.5 Více možností cesty, změna formuláře a design klienta

Při výběru kratších cest veřejnou dopravou klient často vykresloval trasu pouze pěšky. Tento problém je způsoben řazením nejkratších cest na první pozici ve vygenerovaném JSON souboru. Proto byla vytvořena možnost vybírat si až z třech různých cest, čímž byl problém eliminován.

V první řadě bylo omezeno maximální počet možných cest (itineraries) na tři a tím došlo k výraznému navýšení výpočetní rychlosti, neboť OTP 2 nemusel vypočítávat velké množství cest a velikost JSON souboru byla zmenšena. Do parametrů pro výpočet cesty byl přidán prvek *numItineraries* a jeho hodnota nastavena na 3. Následně ve formuláři byla odstraněna možnost výběru maximální vzdálenost chůze, protože vývojáři pozměnili fungování parametru *maxWalkdistance* a již nebylo možné s ním dále pracovat. Ovšem maximální chůze byla nahrazena novým formulářovým elementem pro výběr času a datumu odjezdu nebo příjezdu z vybraného místa (Obr. 4.17). Pomocí JS funkce byla vytvořena funkcionalita, kdy při každém načtení stránky je nastaven datum a čas na aktuální. Poté vedle okna času bylo přidáno resetovací tlačítko pro možnost obnovení data a času do aktuální podoby bez nutnosti obnovovat celou stránku.



Obr. 4.17 Formulářový element pro výběr času a datumu odjezdu nebo příjezdu.

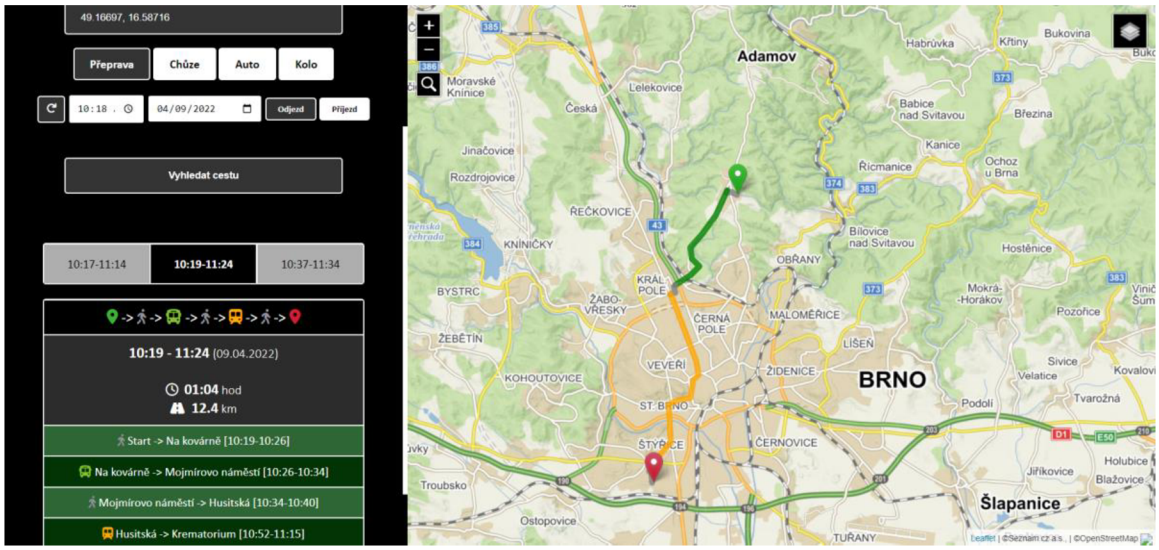
Mezi formulář a vypisování kroků cesty byly přichystány tři tlačítka pro přepínání vygenerovaných cest a dvě nové totožné sekce pro vykreslení kroků cesty. Těmto tlačítkům byla přiřazena jQuery funkce, která po aktivaci jednoho z tlačítek vykreslila odpovídající jednu sekci a ostatní dvě byly schovány. Pro možnost zobrazit odlišné cesty v každé sekci musely být třídy divů vygenerovány s unikátními názvy. Názvy byly generovány s číselnou příponou odpovídající sekce. Názvy první sekce byly bez přípony, druhé sekce s příponou 1 a třetí sekce s příponou 2. Tím se tak názvy tříd jednoduše

roztřídily. Jednotlivé statistiky a textové informace o krocích cesty pak byly přiřazeny do odpovídající sekce.

Zároveň muselo být ošetřeno změnění linie v mapě po překliknuti tlačítka. Souřadnice cest byly rozřazeny do unikátních proměnných stejně jako u názvů tříd. Pomocí podobného principu byla dynamicky zobrazována vždy jen jedna cesta a ostatní dvě byly z mapy odstraňovány. Do tlačítek byl na závěr přenesen čas odjezdu a příjezdu dané cesty.

Po několika zkušebních pokusech vykreslení cest byl zjištěn problém vyskytující se při vygenerování pouze jedné nebo dvou cest. V tomto případě celý kód nemohl být spuštěn a klient přestal fungovat. Do kódu bylo implementováno ošetření pro eliminaci tohoto problému. Na začátku generování názvu sekcí a zobrazování textových informací kroků cesty bylo pomocí zápisu `len(data['plan']['itineraries'])` zjištěno, kolik bylo vygenerovaných cest. Poté byly jednotlivé sekce kódu rozděleny na základě všech možností (mohly nastat tři možnosti, jelikož maximální počet cest byl omezen na 3). Sekce kódů byly upraveny, aby vypisovaly právě jen daný počet cest stejně jako vygeneroval program OTP 2. Velikost programového kódu klienta se tímto problémem téměř ztrojnásobil.

Pro lepší přehlednost a orientaci v mapě byla aplikována interaktivita kroků cest. Po najetí myší na některý z oddílů s informacemi o úseku cesty byl daný úsek v mapě zvýrazněn žlutou barvou. Barva zvýraznění úseku cesty pomocí auta byla nastavena červenou, protože ve výchozí podobě byla cesta autem žlutá a nedošlo by k danému efektu. Na závěr byl vylepšen celkový design klienta a byl aplikován tmavý motiv (Obr 4.18).



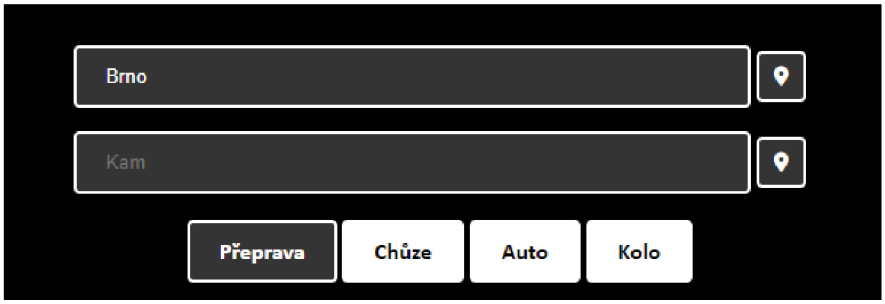
Obr. 4.18 Nový design a tmavý motiv klienta s tlačítky pro přepínání cest.

4.2.6 Překlad názvů a Nominatim geolocator

Pokud je cesta generována po komunikacích nebo chodnících, které nemají název, je namísto názvu ulice v textové navigaci zanesena výchozí hodnota znázorňující daný typ povrchu, po kterém má uživatel jít. Například místo názvu ulice je v textu uveden název chodník nebo silnice. Ovšem tyhle názvy jsou ve výchozí podobě v anglickém jazyce, tudíž musely být přeloženy do češtiny.

Samotný OTP 2 obsahuje lokalizační soubory pro překlad, ovšem v tomto případě byly názvy přeloženy uvnitř kódu klienta, neboť názvy ulic pro navigaci byly ukládány do proměnných, čímž byl umožněn jejich překlad. K překladu bylo přistupováno způsobem: když se obsah proměnné rovnal určitému anglickému názvu, tak byl nahrazen odpovídajícím českým názvem. Po zjištění všech výskytů možností typu komunikací byly vybrány a přeloženy tyto názvy: *Sidewalk* a *path* na chodník, *road* a *service road* na silnici, *track* na cestu, *bike path* na cyklostezku, *ramp* na nájezd, *platform* na nádraží a název *link* na napojení. Veškerá textová navigace tak byla v českém jazyce.

Úplně poslední funkcionalitou přidanou do klienta byla geolokace a vytvoření počátečního a koncového bodu pomocí formulářového elementu pro zadávání souřadnic. Pomocí vyhledávacího nástroje Nominatim od OpenStreetMap (OpenStreetMap contributors 2022) a JS knihovny jQuery byla přidána možnost vyhledat jakoukoliv adresu nebo adresní místo v textovém poli *Odkud* nebo *Kam* pro zadávání souřadnic. Po zadání názvu adresy do některého pole a zmáčknutí nově vytvořeného tlačítka ležícího vedle (Obr. 4.19), je vygenerován počáteční nebo konečný bod cesty v mapě a název adresy je nahrazen souřadnicemi bodu. Tento způsob výrazně ulehčuje vybírání počátečního a koncového místa. Jedinou nevýhodou nástroje je nemožnost vyhledat adresy s diakritikou. Nicméně zapsáním adresy bez diakritiky je daná lokalita bez problému vyhledána. Tento nástroj v podstatě nahradil celou funckionalitu Leaflet pluginu Control Geocoder popsaný v kapitole 4.2.3, ovšem pro lepší vizuální a funkční naplněnost byl plugin v klientovi ponechán.



Obr. 4.19 Geolokace adresních míst pomocí vedlejšího tlačítka s obrázkem markeru. Po kliknutí na toto tlačítko bude v mapě vytvořen bod v centru Brna a název ve formuláři bude nahrazen jeho souřadnicemi.

Plánovač byl v tomto momentě plně funkční a obsahoval veškeré nezbytné prvky pro jednoduché zadávání cesty. Výsledná cesta byla v přehledné podobě a měla velkou informační hodnotu. Pro ověření správné koncepce a funkčnosti klienta bylo provedeno testování od uživatelů a srovnání s již vytvořenými plánovači běžících pomocí softwaru OTP 2.

5 TESTOVÁNÍ KLIENTA

Po vytvoření a naprogramování celého klienta muselo být provedeno testování od uživatelů pro ověření správné koncepce a funkčnosti. Následně byl vytvořený plánovač porovnán s existujícími vyhledávači.

5.1 Testování od uživatelů

Jelikož klient prozatím fungoval jen na lokálním zařízení, bylo pozváno sedm respondentů, kteří podnikli několik vybraných úkolů, podle kterých se vyhotovily závěry a opravily, popřípadě zapsaly některé nedostatky klienta, na které respondent upozornil. Čtyři respondenti byli odborníci na GIS (geografický informační systém) technologie a zbylí tři nebyli s GIS problematikou vůbec seznámeni.

Respondentům byly postupně zadávány tyhle úkoly:

- Vyhledat cestu z Blanska do Židlochovic veřejnou dopravou,
- Vyhledat cestu z Brno Kroftova do Brno Trnkova 5,
- Vyhledat jakoukoli cestu s odjezdem 15.4.2022 ve 13:00 hodin,
- Přepnout podkladové mapy,
- Říct název konečné lokace při vyhledání cesty na kole z Hodonína do Břeclavi.

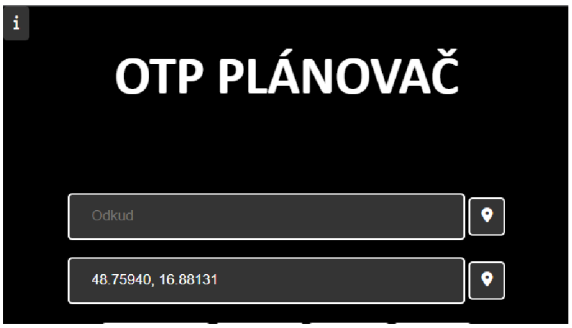
Tab. 5.1 Testování klienta od uživatelů na základě úkolů

Respondent	Úkol 1	Úkol 2	Úkol 3	Úkol 4	Úkol 5
Respondent 1	splnil	splnil	splnil	splnil	splnil
Respondent 2	splnil	splnil	splnil	splnil	splnil
Respondent 3	splnil	splnil	splnil	splnil	splnil
Respondent 4	splnil	splnil	splnil	splnil	splnil
Respondent 5	splnil	splnil	splnil	splnil	splnil
Respondent 6	splnil	splnil	splnil	splnil	splnil
Respondent 7	splnil	splnil	splnil	splnil	splnil

První respondent splnil bez výraznějších problémů všech pět úkolů. Vytknul pouze nemožnost potvrdit geolokaci adresy v textovém elementu formuláře klávesou enter a nelíbil se mu příliš jednoduchý design klienta. Druhý respondent měl pouze problémy u zadávání adresy do textového pole, protože nebyl obeznámen se psaním adres bez diakritiky. V klientovi by uvítal možnost nápovědy (našeptávače) při vyhledávání adres. Třetí i čtvrtý odborný respondent neměli se splněním úkolů výraznější problémy a všechny zadání zvládli.

U respondentů, kteří neměli s geoinformatickými technologiemi žádné zkušenosti byly výsledky lehce odlišné. Veškeré úkoly splnili, ovšem musela jim být více podrobněji vysvětlena funkčnost klienta. Pro uživatele bylo nejtěžší pochopit, jak zadávat adresy a vytvářet počáteční a koncový bod cesty. Ovšem po zjištění nutnosti zadat bod kliknutím do mapy nebo zapsáním adresy do textového pole ve formuláři, nebylo pro ně problémem splnit veškeré zbylé úkoly.

Na základě výsledků bylo do klienta zabudované informační tlačítko, po jehož aktivaci je v plánovači zobrazena nová sekce obsahující základní informace o vyhledávání cesty v mapě a funkčnosti (Obr. 5.1). Toto informační okno může uživatelům pomoci, pokud si nebudou vědět rady s vyhledáváním cesty, konkrétně jak zadat počáteční a konečný bod.



Obr. 5.1 Informační tlačítko v levém horním rohu. Po jeho zmáčknutí jsou uživatelovi zobrazeny základní informace o klientovi.

5.2 Srovnání s existujícími vyhledávači

Ve světě existuje několik veřejně dostupných plánovačů generující cesty pomocí programu OTP 2. Pro srovnáni klienta s existujícími vyhledávači byl vybrán Norský národní plánovač Entur (Entur AS 2022), plánovač města Valencie ve Španělsku (EMT Valencia 2022), francouzský plánovač města Grenoble (Mobilités-M 2021) a český multimodální plánovač Bileto (Bileto 2018), který ale nevyužívá OTP 2.

Do tabulky 5.2 byly zapsány a hodnoceny prvky, na základě kterých byly jednotlivé plánovače porovnávány. Jednalo se o design a přehlednost plánovače, funkcionalita, informační hodnota a zadávání cesty. Jednotlivé prvky byly ohodnoceny známkami 1 až 5 (1 nejlepší, 5 nejhorší). Pro lepší popis byl klientovi, vytvořeného v této bakalářské práci, dán název OTP plánovač

Tab. 5.2 Hodnocení prvků na vybraných plánovačích

Název plánovače	**Design a přehlednost**	**Funkcionalita**	**Informační hodnota**	**Zadávání cesty**
OTP plánovač	1	1	2	1
Entur	2	3	3	2
Valencie	3	2	1	1
Grenoble	3	3	2	3
Bileto	2	4	2	3

Plánovač města Valencie je kromě designové stránky téměř na stejné úrovni jako OTP plánovač. Jeho informační hodnota a schopnost zadávat cestu pomocí geolokace adres i kliknutím na dané místo do mapy odpovídá prvkům OTP plánovače. Plánovač Valencie v mapě zobrazuje veškeré autobusové i tramvajové zastávky. Ze zastávek lze vybrat vhodný spoj.

Stejně jako u plánovače Bileto a plánovače Grenoble, má klient této bakalářské práce vysokou informační hodnotu. Jednotlivé kroky cesty jsou u těchto plánovačů

rozděleny na úseky a každý úsek má možnost rozkliknutí detailnějších informací. Navíc OTP plánovač má pro lepší přehlednost možnost zvýraznit úsek, což ostatní plánovače nemají.

Nevýhodou francouzského plánovače je absence možnosti přeložení plánovače do anglického jazyka a vybrat počáteční a koncový bod z mapy. Adresy lze vyhledat jen přes textové pole ve formuláři. Plánovač je pro zahraniční uživatele obtížné používat. Unikátním prvkem u plánovače Entur je možnost zakoupení lístků pro danou cestu přímo v plánovači.

Oproti plánovače Entur a Bileto má OTP plánovač daleko propracovanější funkcionalitu. V Biletu není možnost interagovat v mapě s nově vykreslenou cestou a jednotlivé spoje jsou spojeny vzdušnou čarou. V OTP plánovači je cesta v mapě nalinkovaná na odpovídající cesty a silnice.

Velká diverzita u plánovačů je ve výběru interaktivní mapy pro zobrazení cest. Pro vykreslení cest v mapě je pro tento klient použita JS knihovna Leaflet. Plánovač Bileto používá interaktivní mapu od Google, norský plánovač Entur používá online mapu od poskytovatele Mapbox a zbylé dva plánovače měst Grenoble a Valencie pak používají knihovnu pro zobrazování map OpenLayers.

6 VÝSLEDKY

Výsledkem bakalářské práce je multimodální plánovač umožňující vyhledávat a plánovat cesty různými dopravními prostředky i veřejnou dopravou po celém Jihomoravském kraji. Serverová část používá vhodně nakonfigurovaný program pro generování cest OpenTripPlanner 2. Klientská část umožňuje uživateli zadávat různé nastavení cesty a vybírat si počáteční a konečný bod cesty. Webový klient je napsán a funguje pomocí Pythonu a CGI (Common Gateway Interface) protokolu pro propojení externích aplikací s webovým serverem.

Program OpenTripPlanner 2 generuje cesty za pomocí dvou datových formátů: GTFS formát jízdních řádů pro Jihomoravský kraj a prostorový PBF formát stejné oblasti z OpenStreetMap. Klient funguje na základě odesílání dotazů do spuštěného programu OTP 2 na serveru, který vygeneruje JSON strukturu obsahující veškeré informace o nově vzniklé cestě (viz podkapitoly 4.1.5 a 4.2.2). Za pomocí jazyku Python jsou v klientu informace zobrazovány ve formě linie v mapě a textu jako kroky cesty s detailnějšími informacemi.

Uživatelské rozhraní klienta je rozděleno na dvě části. V levé části se nachází uživatelský formulář a sekce pro zobrazování textových informací o cestě. V pravé části je interaktivní mapové pole, ve kterém se zobrazuje cesta ve formě linie (viz podkapitola 4.2.1). Pro zobrazení cesty si uživatel nejprve vybere počáteční a koncový bod cesty kliknutím pravým tlačítkem do mapy. Oba body jdou také vytvořit pomocí napsání adresních míst do textových elementů formuláře (viz podkapitola 4.2.6). Uživatel si dále vybere typ přepravy a čas a datum odjezdu nebo příjezdu. Po kliknutí na potvrzovací tlačítko jsou vygenerovány až tři možné cesty spojující dva vybrané body. Linie v mapě je vhodně obarvena podle typu přepravy v jednotlivých úsecích cesty. Paralelně je v klientovi vygenerována sekce obsahující detailnější textové informace o krocích a statistikách cesty (viz podkapitola 4.2.4). Klient je zasazen do přehledné, jednoduché, ale velice informativní podoby a také obsahuje pomocné dynamické prvky pro zjednodušení orientace ve čtení jednotlivých cest. Například zvýrazňování jednotlivých úseků cesty po najetí myší na odpovídající textovou část.

Závěrem byl klient porovnán s již existujícími vyhledávači, generující cesty za pomocí programu OpenTripPlanner 2 a s jedním, fungující jiným způsobem (viz podkapitola 5.2). V porovnání s ostatními vyhledávači je tento klient vhodně sestrojen a jeho funkcionalita se podobá i velice používaným plánovačům v zahraničí. Toto tvrzení podpořilo i několik vybraných respondentů, kteří klient vhodně otestovali za pomoci splnění několika zadaných úkolů (viz podkapitola 5.1).

Webový klient byl vytvořen a testován na lokálním zařízení, ovšem pro pohodlnější přístup byl později nainstalován na školní server. Plánovač je dostupný na adrese:

- **http://158.194.229.229/cgi-bin/index.py**

Programový kód klienta je dostupný na GitHub centrále na adrese:

- **https://github.com/MichalPoty/ClientOTP2**

7 DISKUZE

Ačkoliv je webový klient funkční a správně vypočítává a zobrazuje cesty, lze v něm najít některé nedostatky, které z určitých důvodů nemohly být vyřešeny.

Prvním problémem je, když uživatel vybere počáteční nebo konečný bod na odlehlém místě bez přístupu, například v lese, tak občas nastane, že program OTP 2 nenalezne žádnou cestu, jelikož neexistuje. Tento problém by pravděpodobně šel eliminovat automatickým vytvořením bodu na nejbližší možné cestě, ovšem tohle řešení by bylo příliš komplikované a náročné. Dalším podobným problémem je nemožnost vyhledat dlouhé pěší cesty. Pokud uživatel zadá cestu pouze pěšky, například přes celý Jihomoravský kraj, program nenalezne žádnou cestu. Maximální délka pěší chůze se pohybuje okolo 10 km. Vývojáři programu OTP 2 možnost zadávat dlouhé pěší cesty omezili z důvodu lepší výpočetní rychlosti, protože nepočítali s tím, že multimodální plánovač by byl používán tímto způsobem. V budoucnu by tento problém šel eliminovat vydáním nové verze programu s možností vyhledávat delší pěší cesty.

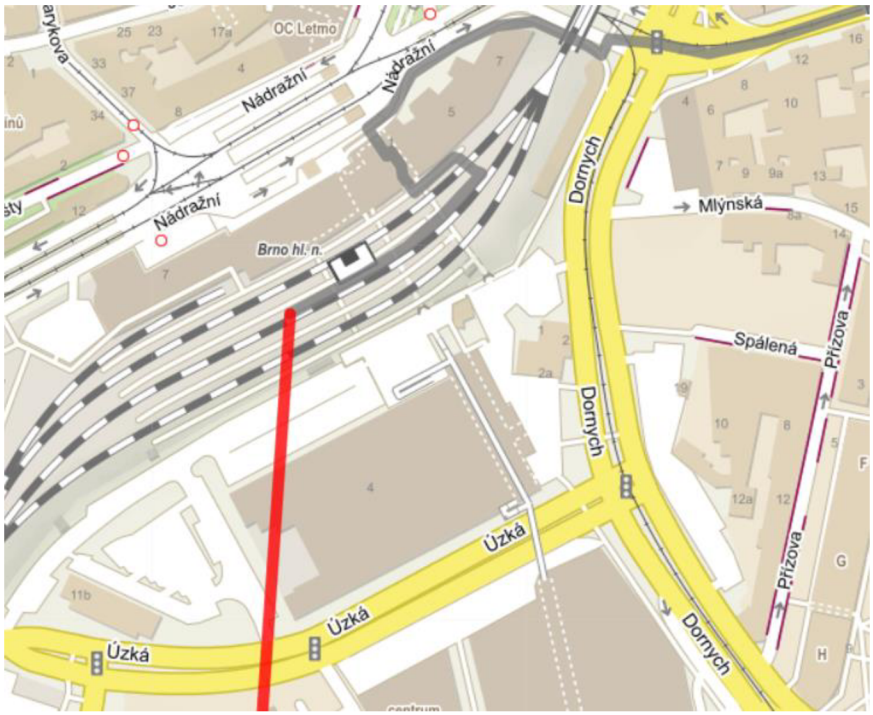
Dalším omezením pro správný chod klienta je také nutnost aktualizace GTFS dat jízdních řádů vstupujících do programu OTP 2. Jízdní řády jednotlivých spojů se neustále mění. Například nastane výluka u vlakové přepravy, nebo kvůli opravě vozovky autobusy jezdí objízdnou trasou. Proto je zde nutnost provádět aktualizaci jízdních řádů. Data Jihomoravského kraje použitá v tomto klientovi jsou společností KORDIS JMK aktualizována jednou týdně. Nicméně pro jejich fungování v klientovi musí být vždy ručně stažena a nahrazena za data současná. Kdyby data v klientovi nebyla aktualizována delší dobu, tak může nastat dokonce i možnost, že cesty veřejnou dopravou nebudou vyhledány, jelikož jedny GTFS data jsou platná zhruba na 2 měsíce dopředu. V budoucnu by mohl být tento problém eliminován vytvořením automatizovaného systému, který by každý týden data aktualizoval.

U již zmíněných GTFS dat Jihomoravského kraje nastal problém při vygenerování souřadnic cesty veřejnou dopravou. Jak je možné vidět na obrázku 7.1, linie veřejné dopravy nejsou správně „nalinkovány“ na podkladovou mapu. Místo toho, aby červená linie (vlak) vedla po kolejích, je vygenerována spojením od zastávky k zastávce vzdušnou čarou. Tento problém nastal pouze u veřejné přepravy GTFS dat Jihomoravského kraje. Například na datech Pražské integrované dopravy nebo datech města Oregon je nalinkování správné. Na vyřešení tohoto problému bylo věnováno dost času, ovšem bez jakéhokoli úspěchu. Data jsou relativně nově dostupná veřejnosti (od 16. srpna 2021) a proto se došlo k závěru, že jsou pro program OTP 2 nevhodně publikována. Ovšem pro základní funkcionalitu není tenhle problém omezující, jedná se pouze o vizuální nedostatek, a proto jsou data nadále používána.

Na základě uživatelského testování bylo zjištěno trochu neintuitivní zadávání bodů do mapy pomocí souřadnic. Uživatelé často mačkali klávesu enter místo kliknutím na tlačítko pro vytvoření bodu a tím odeslali nevyplněný formulář. Zároveň použitý nástroj Nominatim od OpenStreetMaps nemá v základní podobě možnost „našeptávače“ (nápověda, která se zobrazí při zadání například tří písmen). Aplikování obou funkcí by zabralo velké množství času.

Funkcionalita klienta by mohla být rozšířena o daleko více věcí. Program OTP 2 nabízí možnost aplikovat pro vygenerování cest také údaje o převýšení a nadmořských výškách, čímž by se výpočet zpřesnil a cesta by více odpovídala realitě. Dále by šla přidat možnost generace cest s bezbariérovým přístupem. Nejlepší inovací by bylo předělat klient do mobilní aplikace a přidat uživatelovi informaci o aktuální dopravě

nebo zpoždění a na základě toho, vypočítávat nejrychlejší a nejoptimálnější cesty. Aplikace by byla dostupná na všech mobilních zařízení a usnadňovala by pohyb obyvatel po městech, mezi městy nebo dokonce i po celém státě. Výhoda mobilní aplikace by spočívala v možnosti používat plánovač i ve venkovním prostředí a plánovat cesty v reálném čase.



Obr. 7.1 Ukázka vizuálního nedostatku u linií veřejné dopravy. Linie značící vlak (červená) spojuje vzdušnou čarou sousedící zastávky místo toho, aby vedla po kolejích.

8 ZÁVĚR

Cílem bakalářské práce bylo vytvořit webový plánovač spojení veřejnou dopravou pro vybraný region ČR za pomocí programu OpenTripPlanner 2. Výsledkem práce je webový klient rozdělený na formulářovou část pro zadávání dotazů a mapovou část pro zobrazování cesty ve formě linie. Plánovač zobrazuje cesty v oblasti celého Jihomoravského kraje. Ve formulářové části se nachází sekce pro zobrazení detailních textových informacích o výsledné cestě.

V první části tvorby plánovače proběhla konfigurace programu a jeho zprovoznění na lokálním zařízení. V následující části byl vytvořen samostatný webový klient, který na základě uživatelských dotazů generuje informace o cestě a vhodně je zobrazuje. Klient funguje na základě programovacího jazyka Python společně s jazyky pro tvorbu webu.

Klient byl vhodně otestován vybranými odbornými i neodbornými uživateli a na základě jejich připomínek dodatečně upraven. Nicméně uživatelé celkově hodnotili klient jako velice povedený a použitelný.

V budoucnu by mohl být klient předělán do kompaktnější mobilní verze a lze předpokládat, že by mohl být využíván i širokou veřejností.

POUŽITÁ LITERATURA A INFORMAČNÍ ZDROJE

AGAFONKIN, Vladimir, 2010. *Leaflet* [online] [vid. 2021-09-18]. Dostupné z: https://leafletjs.com/SlavaUkraini/

BILETO, 2018. *Bileto* [online] [vid. 2021-12-08]. Dostupné z: https://bileto.cz/

BOTEA, A, M BERLINGERIO, S BRAGHIN, E BOUILLET, F CALABRESE, B CHEN, Y GKOUFAS, R NAIR, T NONNER a M LAUMANNS, 2016. Chapter 16 - Docit: an integrated system for risk-averse multimodal journey advising. In: Mohammad S OBAIDAT a Petros B T - Smart Cities and Homes NICOPOLITIDIS, ed. [online]. Boston: Morgan Kaufmann, s. 345–359. ISBN 978-0-12-803454-5. Dostupné z: https://doi.org/10.1016/B978-0-12-803454-5.00016-X

BOTEA, Adi, Evdokia NIKOLOVA a Michele BERLINGERIO, 2013. *Multi-Modal Journey Planning in the Presence of Uncertainty* [online]. Dostupné z: https://users.ece.utexas.edu/~nikolova/papers/BoteaNikolovaBerlingerio-ICAPS13.pdf

CAMPBELL, Steve, 2022. *Python Vs PHP* [online] [vid. 2021-12-12]. Dostupné z: https://www.guru99.com/python-vs-php.html

CHAPS, 2007. *IDOS jízdní řády* [online] [vid. 2021-12-10]. Dostupné z: https://idos.idnes.cz/

DAN'S TOOLS, 2014. *Unix Time Stamp* [online] [vid. 2022-01-25]. Dostupné z: https://www.unixtimestamp.com/

EMT VALENCIA, 2022. *Valencia Journey Planner* [online] [vid. 2022-04-13]. Dostupné z: https://geoportal.emtvalencia.es/visor?lang=en

ENTUR AS, 2022. *Entur* [online] [vid. 2022-04-13]. Dostupné z: https://entur.no/

FONTICONS, 2022. *Font Awesome 6* [online] [vid. 2022-02-20]. Dostupné z: https://fontawesome.com/

GATEWAYGEO, 2008. *MapServer for Windows* [online] [vid. 2021-09-18]. Dostupné z: https://ms4w.com/download.html

GITHUB, 2019. *Leaflet Context menu* [online] [vid. 2021-10-21]. Dostupné z: https://github.com/aratcliffe/Leaflet.contextmenu

GITHUB, 2020a. *Leaflet Control Geocoder* [online] [vid. 2022-01-16]. Dostupné z: https://github.com/perliedman/leaflet-control-geocoder

GITHUB, 2020b. *OpenTriPlanner 2* [online] [vid. 2021-08-24]. Dostupné z: https://github.com/opentripplanner/OpenTripPlanner

GOOGLE, 2022. *Google Encoded Polyline Algorithm Format* [online] [vid. 2021-10-20]. Dostupné z: https://developers.google.com/maps/documentation/utilities/polylinealgorithm

GRACZYK-RACZYŃSKA, Monika, Mateusz OLSZEWSKI, Kacper GÓWCZEWSKI a Damian DEREBECKI, 2020. Multimodal journey planners. GREATER LONDON AUTHORITY, 1999. *Transport for London* [online] [vid. 2021-12-10]. Dostupné z: https://tfl.gov.uk/plan-a-journey/

INSPIROCK, 2022. *Inspirock* [online] [vid. 2021-12-10]. Dostupné z: https://www.inspirock.com/

KISIO DIGITAL AND CONVEYAL, 2018. *OpenTripPlanner and Navitia comparison* [online] [vid. 2021-12-12]. Dostupné z: https://github.com/CanalTP/navitia/wiki/OpenTripPlanner-and-Navitia-comparison

MAGISTRÁT MĚSTA BRNA, 2021. *data Brno* [online] [vid. 2021-12-11]. Dostupné z: https://data.brno.cz/datasets/jízdní-řád-ids-jmk-ve-formátu-gtfs-gtfs-timetable-data/about

MHD.SK, 2000. *Imhd.sk* [online] [vid. 2021-12-10]. Dostupné z: https://imhd.sk/ba/

MOBILITÉS-M, 2021. *M, la marque des mobilités sur l'Aire Grenobloise* [online] [vid. 2022-04-13]. Dostupné z: https://www.mobilites-m.fr/

MOBILITYDATA, 2010. *GTFS* [online] [vid. 2021-12-10]. Dostupné z: https://gtfs.org/

OPENDATA, 2015. *Opendata Praha* [online] [vid. 2021-08-25]. Dostupné z: https://opendata.praha.eu/dataset/jizdni-rady-pid

OPENSTREETMAP CONTRIBUTORS, 2022. *Nominatim* [online] [vid. 2022-04-05]. Dostupné z: https://nominatim.openstreetmap.org/ui/search.html

OPENSTREETMAP WIKI CONTRIBUTORS, 2020. Main Page. *OpenStreetMap Wiki* [online] [vid. 2021-09-18]. Dostupné z: https://wiki.openstreetmap.org/w/index.php?title=Main_Page&oldid=2013332

OTP 2 DEVELOPERS, nedatováno. *OTP PlannerResource* [online] [vid. 2021-08-24]. Dostupné z: http://dev.opentripplanner.org/apidoc/2.0.0/index.html

RAC, 2019. *RAC Route Planner* [online] [vid. 2021-12-10]. Dostupné z: https://www.rac.co.uk/route-planner/

RED HAT, 2020. *Rest API* [online] [vid. 2021-12-12]. Dostupné z: https://www.redhat.com/en/topics/api/what-is-a-rest-api

SEZNAM.CZ, 1998. *Mapy.cz* [online] [vid. 2021-12-10]. Dostupné z: https://mapy.cz/

THE JQUERY FOUNDATION, 2006. *jQuery* [online] [vid. 2022-02-08]. Dostupné z: https://jquery.com/

THE PIP DEVELOPERS, 2008. *Pip package installer for Python* [online] [vid. 2021-08-25]. Dostupné z: https://pip.pypa.io/en/stable/

TOMARAS, Dimitrios, Vana KALOGERAKI, Thomas LIEBIG a Dimitrios GUNOPULOS, 2018. Crowd-based ecofriendly trip planning. In: *2018 19th IEEE International Conference on Mobile Data Management (MDM)*. B.m.: IEEE, s. 24–33. ISBN 153864133X. TUTORIALS POINT, 2006. *Tutorials Point - CGI programming* [online] [vid. 2021-09-18]. Dostupné z: https://www.tutorialspoint.com/about/index.htm

W3SCHOOLS, 1999. *W3Schools Css Forms* [online] [vid. 2021-09-18]. Dostupné z: https://www.w3schools.com/css/css_form.asp

ZOGRAFOS, K G a K N ANDROUTSOPOULOS, 2008. Algorithms for Itinerary Planning in Multimodal Transportation Networks. *IEEE Transactions on Intelligent Transportation Systems* [online]. 175–184. Dostupné z: doi:10.1109/TITS.2008.915650

ZOZNAM, nedatováno. Mapa.sk [online]. Dostupné z: https://mapa.zoznam.sk/

PŘÍLOHY

SEZNAM PŘÍLOH

Volné přílohy

Příloha 1 CD

Příloha 2 Poster

Popis struktury CD

Adresáře:

- data_server_OTP2
- kod_aplikace
- text_prace
- web