

**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
BRNO UNIVERSITY OF TECHNOLOGY

**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**  
**ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ**

FACULTY OF INFORMATION TECHNOLOGY  
DEPARTMENT OF COMPUTER SYSTEMS

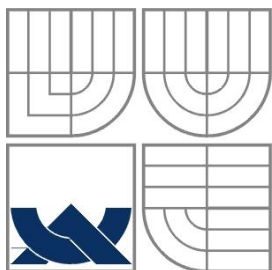
**SERVER PRO SDÍLENÍ TURISTICKÝCH TRAS**

**BAKALÁŘSKÁ PRÁCE**  
BACHELOR'S THESIS

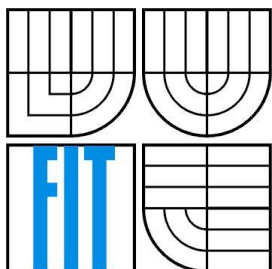
**AUTOR PRÁCE**  
AUTHOR

**ADRIAN MIKULA**

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ  
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ  
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY  
DEPARTMENT OF COMPUTER SYSTEMS

## SERVER PRO SDÍLENÍ TURISTICKÝCH TRAS TOUR ROUTES SHARING SERVER

BAKALÁŘSKÁ PRÁCE  
BACHELOR'S THESIS

AUTOR PRÁCE  
AUTHOR

ADRIAN MIKULA

VEDOUCÍ PRÁCE  
SUPERVISOR

Ing. MICHAL KAJAN

BRNO 2012

## **Abstrakt**

Tato bakalářská práce popisuje návrh a implementaci aplikace pro sdílení turistických tras. Jsou zde vysvětleny související pojmy a technologie použité během implementace. Výsledná aplikace umožňuje vizualizaci tras v mapových podkladech, včetně všech dostupných informací. Trasy lze vyhledávat podle různých kritérií, spojovat a exportovat do KML souborů. Aplikace je dále obohacena o možnost hodnocení nebo plánování nových tras. Systém ke své činnosti využívá funkce rozšíření PostGIS databáze PostgreSQL, algoritmus Haversinovy formule a některé vlastní algoritmy.

## **Abstract**

The bachelor thesis describes design and implementation of the application which can share hiking routes between several users. Thesis also explains related terms and technologies used during the implementation. Final application can present routes including all available informations visualized by using multiple map canvases. Routes can be searched by various criterias, merged and exported into KML files. The system uses functions of extension PostGIS of the PostgreSQL database, algorithm of the Haversine's formula and some own algorithms.

## **Klíčová slova**

Google Maps API, turistická trasa, KML, GPS, WGS-84, PostGIS, PostgreSQL, PHP, Nette Framework, ShadowBox API

## **Keywords**

Google Maps API, tour route, KML, GPS, WGS-84, PostGIS, PostgreSQL, PHP, Nette Framework, ShadowBox API

## **Citace**

Adrian Mikula: Server pro sdílení turistických tras, bakalářská práce, Brno, FIT VUT v Brně, 2012

# Server pro sdílení turistických tras

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Kajana.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....  
Adrian Mikula  
16. května 2012

## Poděkování

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Michalu Kajanovi za cenné rady a připomínky při tvorbě této bakalářské práce.

© Adrian Mikula, 2012

*Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.*

# Obsah

Obsah.....	1
1 Úvod.....	3
2 Teoretický úvod a použité technologie .....	4
2.1 GPS.....	4
2.1.1 Určování polohy .....	4
2.2 Výpočet vzdáleností.....	5
2.2.1 Great Circle.....	5
2.2.2 Haversinova formule.....	6
2.3 Systém WGS-84 .....	6
2.4 Turistická trasa.....	6
2.4.1 Turistické značení.....	6
2.5 Google Maps API .....	7
2.6 Formát KML.....	7
2.7 Databáze PostgreSQL.....	8
2.7.1 Nástroj phpPgAdmin .....	9
2.8 Rozšíření PostGIS.....	9
2.9 Framework Nette .....	10
3 Návrh aplikace .....	11
3.1 Návrh databáze .....	11
3.2 Grafické uživatelské rozhraní .....	12
3.2.1 Navigace webu.....	12
3.2.2 Informační panel.....	13
3.3 Uživatelské funkce.....	13
3.4 Slučování tras.....	14
3.5 Plánování tras.....	15
3.6 Dodatečné funkce .....	15
3.6.1 Informace o počasí.....	15
3.6.2 Vrstva Panoramio .....	16
3.7 Administrátorský účet.....	16
3.8 Zobrazení tras .....	16
3.9 Umístění fotografií do trasy.....	17
4 Implementace .....	18
4.1 Vývojové prostředí .....	18
4.2 Načítání trasy ze souboru.....	18
4.2.1 Doplnující výpočty .....	18
4.3 Hodnocení trasy.....	20
4.4 Navigace do místa začátku trasy.....	20
4.5 Vyhledávání tras .....	21
4.6 Zobrazení obrázků .....	22
4.7 Oznámení nevhodné trasy.....	22
4.8 Generování KML souborů.....	22
5 Testování.....	24
5.1 Testování databáze.....	24
5.2 Testování webového rozhraní.....	24

5.3	Testování aplikace .....	24
6	Závěr .....	26
6.1	Přínos práce .....	26
6.2	Další možná rozšíření .....	26
	Literatura .....	28
	Seznam obrázků.....	30
	Seznam rovnic .....	31
	Seznam příloh .....	32
	Příloha A – ukázka souboru KML.....	33

# 1 Úvod

Turistika je jednou z nejstarších zájmových činností člověka. Kdysi byla spojována s dalekým putováním za potravou, náboženskými cíli nebo objevováním nových území, oproti tomu dnes jde spíše o koníček nebo sport. Ať už kdysi nebo dnes, vždy bylo její nedílnou součástí značení, poznámky a další poznatky z cest. K uložení těchto údajů byla využívána různá média, nejčastěji papír, který byl všem dostupný. Bohužel pro větší zaznamenávanou oblast, bylo potřeba změnit měřítko mapy, nebo použít více listů, což dosti komplikuje následná vyhledávání. Další nevýhodou tohoto záznamu bylo sdílení těchto údajů, které bylo dosažitelné pouze duplikací dokumentu. S rozvojem techniky a pozdější digitalizací, byly tyto problémy vyřešeny, neboť papírová forma byla nahrazena interakčními mapami, ve kterých lze jednoduše vyhledávat, prohlížet, ukládat a editovat trasy. Také záznam trasy se zjednodušil a zpřesnil, díky zaznamenávajícím přístrojům disponujícími GPS moduly. Ačkoli webové aplikace jakými jsou například google maps nebo mapy.cz, umožňují sdílet a vyhledávat trasy, stále je omezena možnost filtrování tras pouze na základní kritéria. Dále jsou uživatelé ochuzeni o vkládání dalších doplňujících informací, jakými jsou například fotografie, informace o povrchu trasy, nebo barevné označení tras.

Cílem této práce bylo vytvořit webovou aplikaci, která umožňuje uživatelům vkládání turistických tras včetně dalších užitečných informací a jejich následné sdílení. Také bude umožňovat vyhledávání podle nejrůznějších kritérií, jakými jsou například obtížnost, převýšení, délka nebo oblíbenost trasy. Aplikace bude podporovat import i export trasy do souboru kompatibilního s většinou GPS navigací.

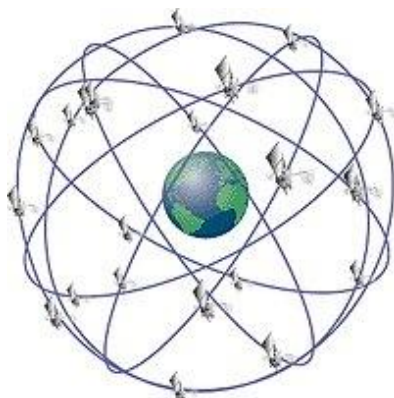
V kapitole 2 se budu věnovat teoretickým znalostem potřebným k pochopení problematik spojených s bakalářskou prací, také popíšu základní technologie použité při implementaci aplikace. Kapitola 3 popisuje návrh aplikace, jsou zde uvedeny možné metody řešení různých problémů a popsány použité postupy. Některé implementační postupy jsou popsány v kapitole 4, kapitola 5 se zabývá způsobem a obsahem testování a kapitola 6 obsahuje shrnutí dosažených výsledků a návrhy pro další rozšíření aplikace.

## 2 Teoretický úvod a použité technologie

V této kapitole budou vysvětleny základní pojmy a technologie použité při implementaci. Bude zde objasněn princip fungování systému GPS a následné určení polohy. Také bude vysvětlen obsah standartu WGS-84. Z použitých nástrojů bude přiblížena knihovna Google Maps API, jazyk pro popis geometrických vektorů KML, databázový systém PostgreSQL včetně své nadstavby PostGIS a v závěru kapitoly bude představen Framework Nette.

### 2.1 GPS

GPS (*Global Position System*) je globální družicový polohový systém, s jehož pomocí lze přesně určit polohu a přesný čas kdekoliv na zemi. Systém GPS je tvořen kosmickým, řídicím a uživatelským segmentem. Kosmický segment GPS (viz Obrázek 2.1) představují družice umístěné na šesti kruhových dráhách, ve vzdálenosti 20 190 km od povrchu Země. Každá dráha má pět pozic pro umístění družic, kde pátá pozice je záložní, protože pro dosažení plné operační způsobilosti postačuje 24 družic [1]. Řídicí segment se skládá z pěti monitorovacích stanic, čtyř pozemních vysílačů a hlavního řídicího střediska. Hlavním úkolem řídicího segmentu je sledování družic a provádění korekcí v dráze letu či vysílaném signálu. Uživatelský segment představují GPS přijímače jednotlivých uživatelů. Tyto přijímače mají pasivní funkci, to znamená, že pouze přijímají signály. Díky tomu systém GPS může obsloužit neomezený počet těchto zařízení.



Obrázek 2.1: Kruhové dráhy družic

#### 2.1.1 Určování polohy

Podle [1] systém GPS pracuje na principu rádiového dálkoměrného systému<sup>1</sup> kde přijímač pro určení 2D polohy musí znát pozici tří satelitů, ve 3D pak nejméně čtyř. Každá družice vysílá konstantní rychlostí světla rádiový signál, obsahující informace o své poloze a přesný atomický čas. Přijímač z těchto informací dokáže za pomoci kódových, fázových, nebo Dopplerových měření vypočítat vzdálenost od družice. Ve většině klasických navigací se používá měření kódových, jejichž předností je relativně nízká výpočetní náročnost, což má za následek rychlejší měření. Ostatní metody se

---

<sup>1</sup> Systém, který určuje polohu objektu ze vzdáleností od bodů se známou polohou, využívající k měření rádiových vln.

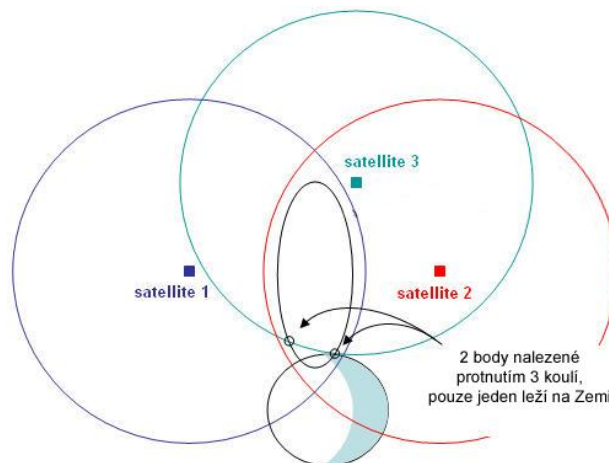
vyznačují vysokou přesností, ale také větší časovou náročností, a proto se používají ve speciálních vědeckých a geodetických aplikacích.

GPS přijímač po přijetí signálu vytvoří pomyslnou kouli o poloměru vypočtených vzdáleností se středem aktuální polohy družice. Když takto zpracuje alespoň tři signály družic, dojde k protnutí průsečíků těchto koulí ve dvou bodech, na základě čehož může určit polohu ve 2D (viz Obrázek 2.2). Pro výpočet trojrozměrné polohy bude zapotřebí ještě alespoň jeden takto zpracovaný signál. V matematické podobě lze toto zapsat pomocí čtyř rovnic koulí o čtyř neznámých (viz Rovnice 2.1), vzorec byl přejat z [1].

$$(X - x_n)^2 + (Y - y_n)^2 + (Z - z_n)^2 = [(T - t_n)c]^2$$

**Rovnice 2.1: Výpočet 3D polohy**

Řešením této rovnice je poloha a čas GPS přijímače. Tyto údaje je ještě potřeba převést do občanského času a zeměpisných souřadnic, nejčastěji popsaných standardem WGS-84, který je dále rozveden níže (viz Kapitola 2.3).



**Obrázek 2.2: Protnutí průsečíků 3 koulí**

## 2.2 Výpočet vzdáleností

Pro určení délky mezi dvěma body určenými zeměpisnými souřadnicemi je třeba použít vhodný algoritmus.

### 2.2.1 Great Circle

Pro delší vzdálenosti, lze použít méně přesný algoritmus Great Circle, který počítá s faktem, že Země je kulatá. Nejkratší spojnici dvou bodů na kulové ploše je ortodroma. Proto vzdálenost lze vypočítat podle rovnice výpočtu délky ortodromy (viz Rovnice 2.2).

$$d = \sigma * r$$

**Rovnice 2.2: Výpočet délky ortodromy**

Kde  $r$  je poloměr koule, v případě Země 6371km (střední poloměr Země) a  $\sigma$  délka krajních bodů ortodromy, která se počítá se středem Země [12]. Tato metoda je ovlivněna velkým zaokrouhlováním, proto nedává dobré výsledky pro krátké vzdálenosti.

## 2.2.2 Haversinova formule

Pro kratší vzdálenosti je lepší použít Haversinovu formuli (viz Rovnice 2.3), u které nedochází k tak velkým zaokrouhlovacím chybám.

$$2r \arcsin \left( \sqrt{\sin^2 \left( \frac{\phi_2 - \phi_1}{2} \right) + \cos(\phi_1) \cos(\phi_2) \sin^2 \left( \frac{\psi_2 - \psi_1}{2} \right)} \right)$$

Rovnice 2.3: Haversinova formule

## 2.3 Systém WGS-84

WGS-84 (*World Geodetic System*) je světový standard, který definuje souřadnicový systém, geoid a referenční elipsoid [18]. Původně byl vyvinut armádou USA, nyní je udržován geodetickým geocentrickým systémem armád NATO.

Geoid je definován jako ideální model Země, který má ekvipotenciální plochu<sup>2</sup> vůči gravitaci při střední hladině moří a oceánů. Tento model se však obtížně matematicky popisuje, a proto je pro geografické a kartografické úlohy nevhodný.

Používá se proto referenční elipsoid systému WGS-84, který je pevně spojený se zemí a definují ho čtyři hlavní parametry (délka hlavní poloosy, převrácena hodnota zploštění Země, úhlová rychlost, součin gravitační konstanty a hmotnosti Země). Je vypočten pomocí satelitních měření a naposledy byl aktualizovaný v roce 2004. Jeho střed je určen na těžiště Země, a je univerzální pro celou planetu, na rozdíl od jiných referenčních elipsoidů (*Airy*, *Bessela* a dalších), které jsou použitelné pouze pro daná území. Díky své univerzální použitelnosti kdekoli na Zemi je používán v systému GPS.

Ze standardu WGS-84 také vychází souřadný systém, kde jednotlivé body jsou definované zeměpisnou délkou, šířkou a výškou. Zeměpisná délka a šířka se obvykle uvádí ve stupních, minutách, vteřinách a výška je uvedena v metrech.

## 2.4 Turistická trasa

Turistická trasa je stezka určena především pro chodce s typickým značením usnadňujícím orientaci v terénu. Tyto trasy se často vyskytují v horských oblastech, v okolí památkových míst nebo dokonce i v městech.

### 2.4.1 Turistické značení

Turistické značení je soustava navigačních bodů, kterou tvoří směrovky, speciální symboly a informační tabule, s jejichž pomocí jsou turisté naváděni, předem určeným směrem. Tento systém je taktéž uváděn v turistických mapách. Toto značení se liší v závislosti na druhu trasy a státu (v některých státech toto značení neexistuje). Rozlišujeme několik druhů turistického značení [19]:

- Turistické pásové – pro pěší turistiku
- Mezinárodní značené cesty – pro delší pěší turistiku
- Naučné stezky – pro pěší nebo cyklisty, s naučným účelem
- Lázeňské trasy – pro lázeňské vycházky

---

<sup>2</sup> Normálová plocha k siločarům silového pole, na ní je potenciál pole konstantní [8].

## 2.5 Google Maps API

Pro zobrazení tras na mapě bylo potřeba zvolit vhodnou knihovnu pro vykreslování dat s vhodnými mapovými podklady. K výběru jsem měl několik dostupných knihoven. Svůj výběr jsem hned na začátku zužil na mně tři nejznámější Google Maps API, OpenLayers a Mapy API [2] obsahují velké množství funkcí s českou dokumentací a podporou. Bohužel jsou vhodné pouze pro projekty zaměřující se na území České republiky, protože obsahují pouze české mapové podklady. Proto přicházely v úvahu již pouze dvě výše uvedené, jelikož společnost Google má obrovskou komunitu a s některými jejími produkty již mám dobrou zkušenost, zvolil jsem tedy jejich knihovnu Google Maps JavaScript API v3.

Google Maps API [16] je založeno na jazyce JavaScript nebo Flash, osobně jsem zvolil JavaScriptovou verzi této knihovny ve verzi 3, která byla nejnovější dostupnou v době vzniku aplikace. Tato knihovna má však jedno svoje omezení pro uživatele, kteří ji chtějí používat zdarma a tím je omezení počtu zobrazení za den. Jelikož se jedná o 25 000 přístupů za den, věděl jsem, že toto číslo v rámci bakalářské práce nebude překročeno. Pokud by byl ovšem systém zařazen do ostrého provozu a byl by úspěšný, co se návštěvností týče, bylo by potřeba používat zpoplatněnou verzi tohoto API.

Google Maps API nabízí mnoho funkcí pro práci s geometrickými vektory (bod, křivka, polygon), dále jsou k výběru různé mapové podklady, které se dají obohatit vlastními vrstvami, a v neposlední řadě je k dispozici Google Street View<sup>3</sup>. Pro knihovnu Google Maps API existuje taky velké množství rozšíření v podobě knihoven třetích stran.

## 2.6 Formát KML

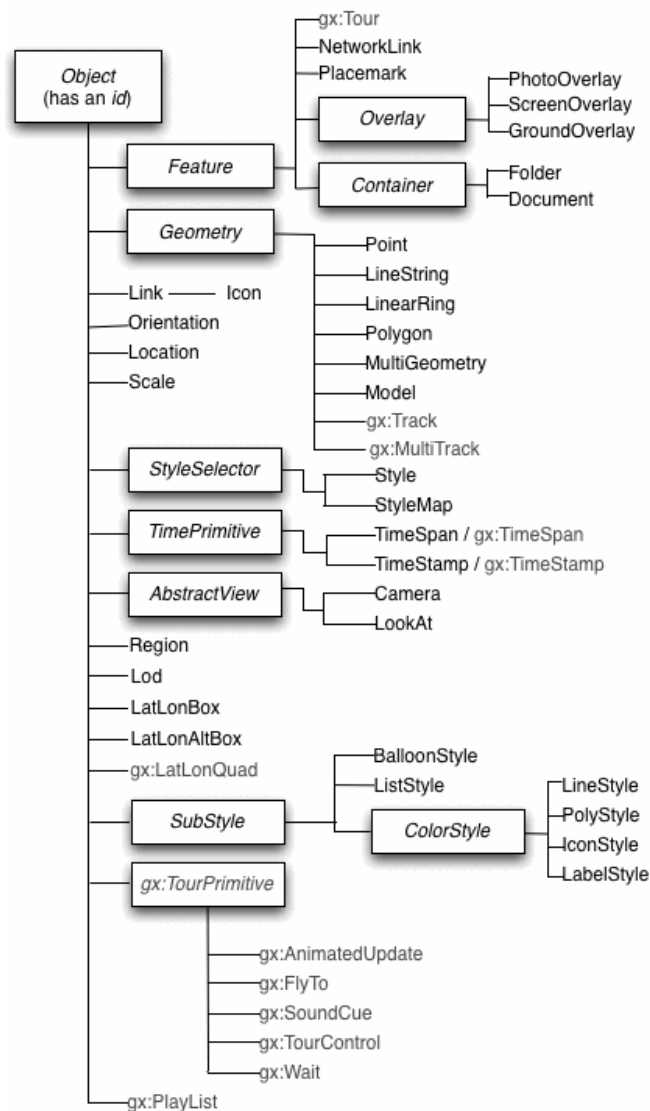
Pro zajištění kompatibility s GPS přístroji a ostatními aplikacemi pracujícími s geografickými daty, bylo potřeba zvolit vhodný formát souboru pro import a export dat. Existuje celá řada takovýchto formátů, jedním z nich je formát KML (*Keyhole Markup Language*). Jedná se o formát udržovaný firmou Google a jeho podporu najdeme dnes ve většině programů komunikujících s GPS přijímači. Pro uživatele vlastníci přístroj GPS, který formát KML nepodporuje, existuje množství aplikací umožňujících převod do tohoto formátu [20].

KML je značkovací jazyk<sup>4</sup> s gramatikou jazyka XML a souborovým formátem pro modelování a ukládání geografických dat. Cílem jazyka KML je reprezentace geografických dat a způsob zobrazení uživatelského rozhraní v aplikacích (např. Google Earth). Výchozím souřadnicovým systémem jazyka KML je systém WGS-84 ve tvaru celých stupňů, více o tomto systému se dozvíte v sekci 2.2. Výšky vztažných bodů využívají vztažný systém EGM-96 [4]. Strukturu a funkce jazyka KML (viz Obrázek 2.3) včetně podrobného popisu geografických prvků, funkcí, syntaxe a příkladů lze najít na webu *Open Geospatial Consortium* [5].

---

<sup>3</sup> Část Google Maps a Google Earth, která nabízí panoramatické pohledy [3].

<sup>4</sup> Jazyk, který vkládá do textu značky vysvětlující význam nebo vzhled jednotlivých jeho částí.



Obrázek 2.3: Struktura a funkce jazyka KML

## 2.7 Databáze PostgreSQL

Všechna data potřebná k chodu aplikace jsou uložena na serveru v databázovém systému PostgreSQL [6]. Jedná se o objektově-relační databázový systém, který je dostupný pod licenci open-source. Jeho hlavní předností díky open-source licenci je hlavně rozšiřitelnost, která uživateli umožňuje nadefinovat si vlastní funkce, datové typy, procedurální jazyky. Právě proto mohlo vzniknout velké množství rozšíření, které dnes existuje, mezi něž patří například PostGIS, Slony-I, pgRouting nebo pgFoundry.

PostgreSQL je primárně vyvíjen pro unixové systémy, ale existují i balíčky pro platformu Win32, díky čemuž lze databázi plnohodnotně využívat i v prostředí Windows. Instalace se nikterak neliší od instalace jiných databázových systémů, jakým je například MySQL, ale pro nové uživatele toto nemusí být zcela triviální záležitost, zvláště díky častým chybám, které se mohou vyskytnout. Osobně doporučuji instalaci balíčku WAPPStack [7], který zároveň obsahuje i instalační soubory PHP a Apache.

## 2.7.1 Nástroj phpPgAdmin

Součástí instalačního balíčku PostgreSQL je taky webová aplikace phpPgAdmin, která slouží k správě databázového serveru. Aplikace vznikla po vzoru phpMyAdmin s cílem poskytnout podobné funkce pro PostgreSQL. Nástroj je napsán v jazyce PHP a lze ho umístit na jakýkoliv webový server, což nám umožňuje přístup k databázi přes webové rozhraní téměř odkudkoli.

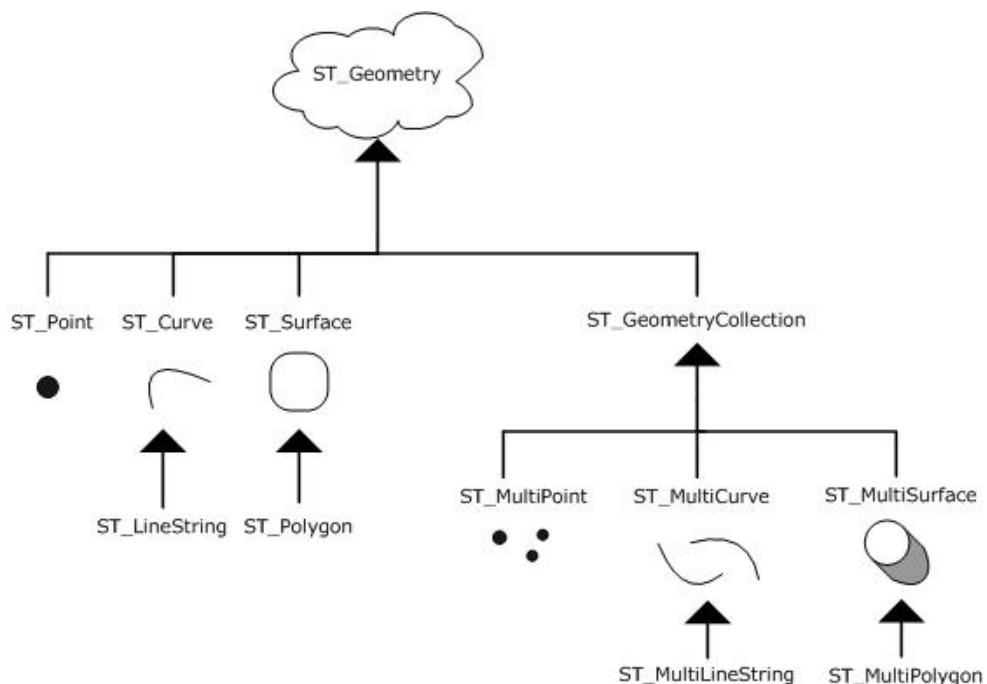
Nástroj phpPgAdmin nabízí základní funkce k práci s tabulkami, pohledy, sekvencemi a funkcemi. Lze taky vytvářet a mazat databáze, nastavovat oprávnění uživatelů nebo provádět export či import dat včetně celé struktury.

## 2.8 Rozšíření PostGIS

Pro uložení geografického objektu do databáze by stačily základní datové typy PostgreSQL, ale veškeré funkce pro práci s nimi, by musely být vykonávané aplikačně. Tento způsob by byl nejen implementačně složitější, ale také náročnost na výkon serveru by vzrostla.

Právě proto vzniklo rozšíření PostGIS, které obohacuje databázi PostgreSQL o nové datové typy a funkce pro práci s geometrickými a geografickými objekty. Jedná se o implementaci OpenGIS standardu a částečně také implementuje SQL/MM. K uložení geometrických dat v databázi se používá jazyk WKB (*Well-known binary*) nebo jeho rozšířenou podobu EWKB (*Extended Well-known binary*) [21], která je rozšířená o třetí dimenzi. V rámci PostGISu existují různé funkce, které tento formát umí převést do textové podoby toho formátu (WKT/EWKT), nebo do fragmentu popsaného jazykem KML. Více o jazycích WKB,EWKB,WKT a EWKT se dozvíte na webu *Open Geospatial Consortium* [5].

V rámci datového typu *geometry* lze rozlišit několik geometrických objektů popsaných jazykem WKT. Tyto prostorové datové typy jsou uspořádány v hierarchii jednotlivých tříd, kde každá podtřída dědí strukturu a chování (metody nebo funkce) své nadřídě (viz Obrázek 2.4).



Obrázek 2.4: Hierarchie datového typu *geometry*

## 2.9 Framework Nette

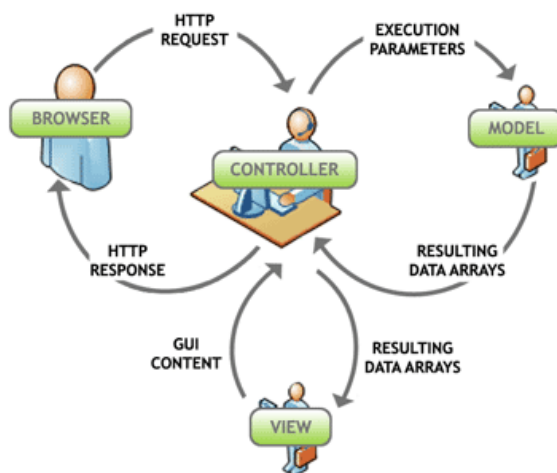
Nette framework je postavený na skriptovacím jazyce PHP s plnou podporou OOP. Jedná se o relativně mladý framework českého vývojáře Davida Grudla. Nette framework je dostupný všem uživatelům jako open-source a svou popularitu si získal díky stále rostoucí komunitě, především českých PHP vývojářů.

Mezi přednosti nette patří především jeho dokonalé zabezpečení a eliminace bezpečnostních děr, velké množství užitečných doplňků a rozšíření, dokonalé ladící nástroje a v neposlední řadě taky jeho vysoký výkon. Podle nezávislého testu [9] je jedním z nejvýkonnějších PHP frameworků vůbec.

Tvorba aplikací v nette vytváří u nezkušených programátorů dobré návyky OOP, také rychlost učení vytvářet webové aplikace je velmi dobrá. Tvorba formulářů a jiných typických webových prvků je v tomto prostředí velmi pohodlná, rychlá i efektivní a to hlavně díky automatickému generování validačních částí v jazyce JavaScript. V nette je také plně podporována technologie Ajaxu.

Architektura frameworku je rozdělena do tří částí (viz Obrázek 2.5), jedná se o tak zvanou MVC architekturu. Tato softwarová architektura rozděluje uživatelské rozhraní, řídicí logiku a datový model do tří nezávislých částí a následná modifikace v každé z nich má minimální dopad na ostatní části. Díky této logice se výborně rozděluje práce v týmu. Dalším plusem nette je oproštění se od roviny URL, místo toho všechny odkazy lze chápat jako zavolání funkce, což je pro programátora určitě přirozenější.

Jedinou nevýhodou tohoto frameworku, je dle mého názoru špatná a často zastaralá dokumentace, což vede k zbytečným chybám a následnému dlouhému řešení těchto chyb a to i když se někdy jedná pouze o triviální záležitosti. Jediným zdrojem informací pak bývá oficiální fórum, což není zrovna nejšťastnější způsob. Informace v něm bývají chaoticky uspořádané a dohledání požadované informace pak zbytečně zdlouhavé.



Obrázek 2.5: Architektura Model-View-Controller

## 3 Návrh aplikace

Tato kapitola popisuje základní návrh databáze, včetně grafického znázornění v podobě ER-diagramu (viz Obrázek 3.1). Také je zde popsáno grafické uživatelské rozhraní a podrobněji rozepsány uživatelské funkce. Dále je zde představena funkcionalita rozšiřující možnosti aplikace pomocí údajů získaných z jiných webových serverů (např. předpověď počasí, fotografie serveru Panoramio). V další části kapitoly jsou popsány funkce plánování a slučování tras a možnosti administrátorského účtu. V samotném závěru kapitoly je vysvětlena možnost ukládání obrázku včetně označování místa jejich pořízení.

### 3.1 Návrh databáze

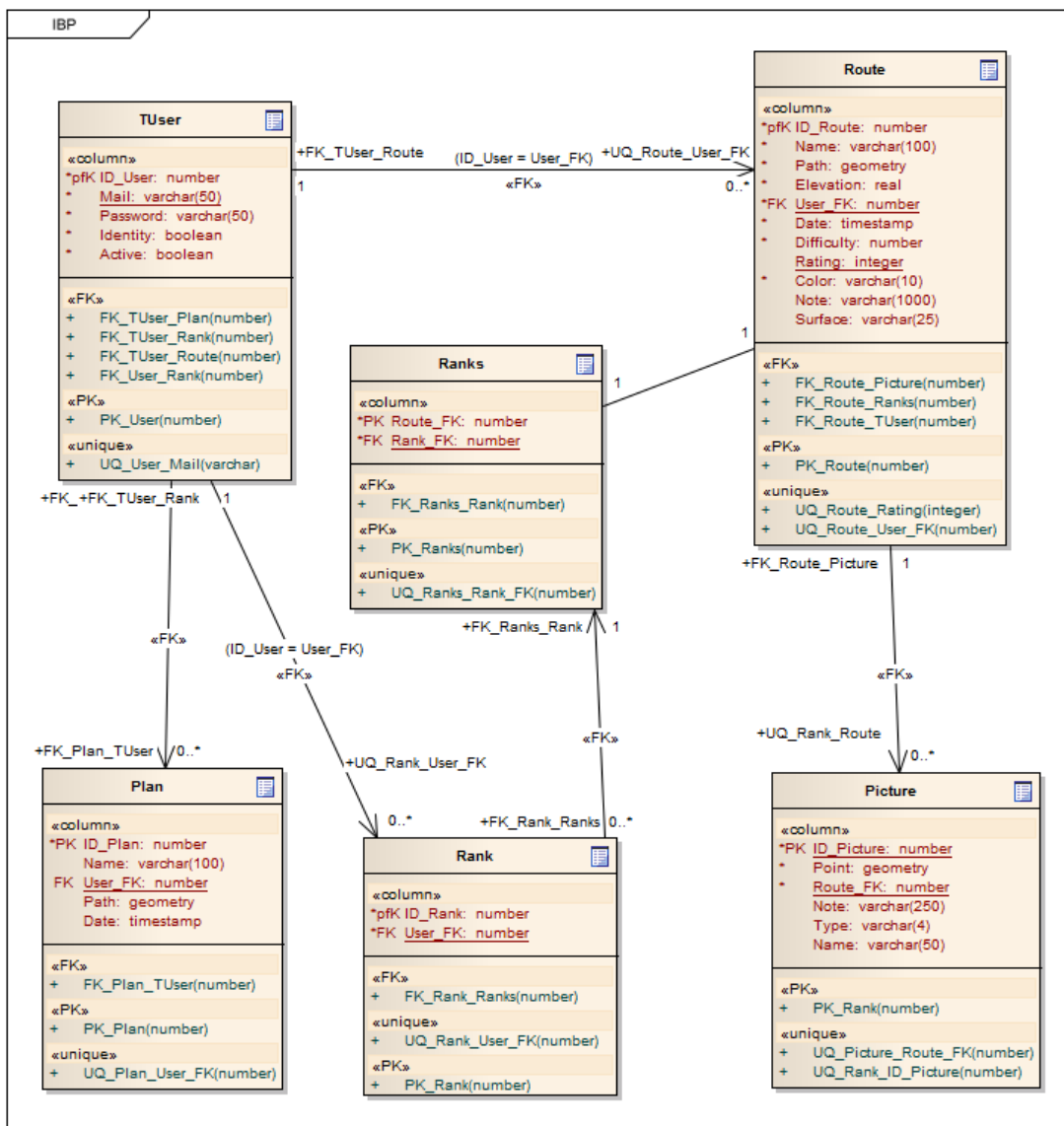
Aplikace pracuje s velkým množstvím dat, proto pro jejich správu je nutné použít databázi. Oproti aplikacím, které pracují s běžnými datovými typy, bylo potřeba počítat taky s typy geometrickými. Zvláště proto na návrh databáze byl kladen velký důraz. K návrhu byl použit klasický ER-diagram, který je znázorněn na schématu níže (viz Obrázek 3.1).

Tabulka `TUser` obsahuje pouze základní sloupce pro jednoznačnou identifikaci uživatele a sloupce obsahující příznaky (`Identity`, `Active`). Sloupec `identity` rozlišuje obyčejné uživatele od uživatelů s právy administrátora a sloupec `active` určuje, zda je účet aktivní, nebo blokován. Jelikož aplikace byla navržena tak, aby nenutila uživatele vyplňovat množství nadbytečných informací, další sloupce nebyly potřeba.

Pro záznam trasy je určena tabulka `Route`. Obsahuje sloupce pro uložení dat popisujících základní údaje trasy (jméno, souřadnice, povrch, barva trasy, atd.) a zároveň je rozšířena o sloupce obsahující předem vypočtená data umožňující rychlejší vyhledávání tras (délka, převýšení, obtížnost). Některé GPS přístroje nebo zvláště aplikace generující KML soubory neposkytují některé dodatečné informace, jako je například převýšení, a proto jsou tyto hodnoty implicitně nastaveny na nulu.

V podobném smyslu je navržena i tabulka `Plane`, která slouží pro ukládání plánovaných tras. Nicméně neobsahuje tolik položek, protože funkce plánování tras slouží pouze k návrhu trasy, nikoli k vytvoření nové trasy v systému a je tudíž viditelná pouze jednomu uživateli. Zamezí se takto vytváření neověřených tras v terénu, nebo alespoň ztíží.

Další entitou je tabulka `Picture`, ve které jsou udržovány záznamy o obrázcích přidaných do trasy. Samotné soubory obrázků jsou umístěny mimo databázi v souborovém systému. Posledními důležitými entitami jsou tabulky `Rank` a `Ranks` sdružující informace o hodnocení tras.



Obrázek 3.1: Návrh databáze

## 3.2 Grafické uživatelské rozhraní

Aby byla aplikace přístupná všem a to bez předchozí instalace, bylo více než vhodné použít webové rozhraní. Při návrhu byl kladen důraz na přehledné a intuitivní ovládání s cílem zobrazit uživateli co největší výsek mapy a zároveň poskytnout co nejvíce užitečných informací. Počítalo se s přístupy zařízení disponujícími obrazovkami s menším rozlišením (např. tablety), ale také i větším (např. Full HD projektory), proto jsou grafické prvky aplikace přizpůsobivé a nedochází tak ke ztrátě potenciálu uživatelského zařízení.

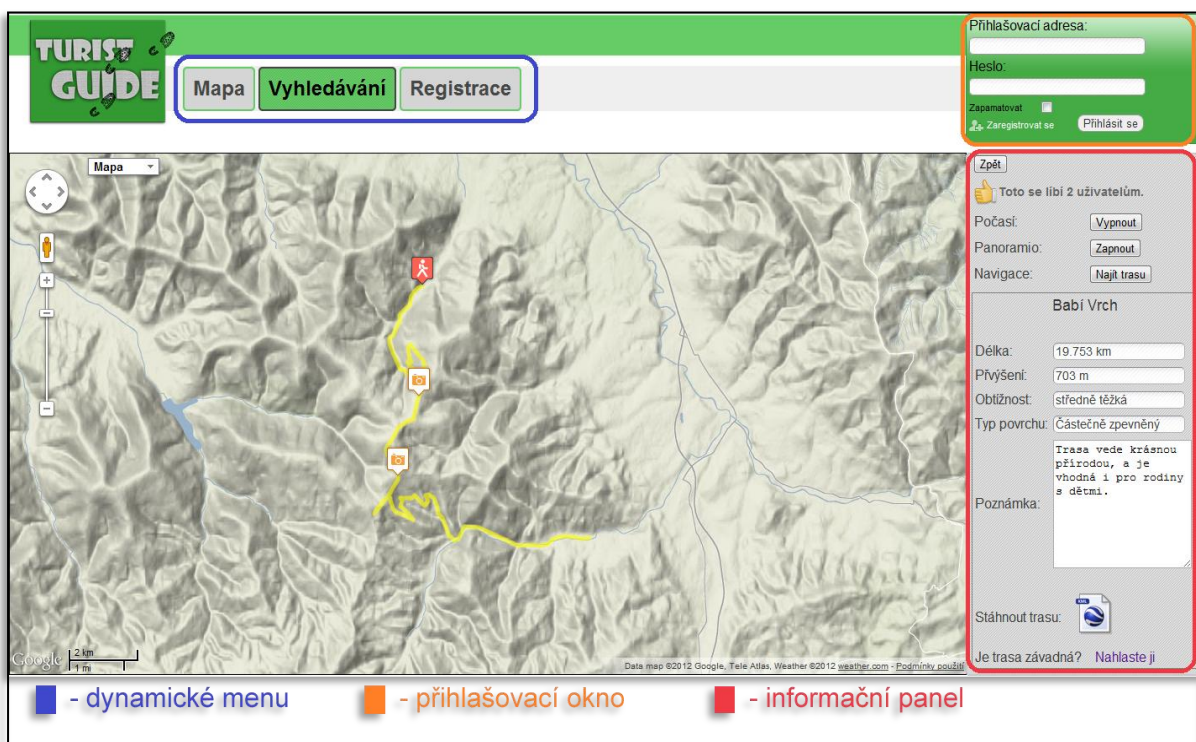
### 3.2.1 Navigace webu

Základním navigačním prvkem aplikace je dynamické menu (viz Obrázek 3.2), které se mění na základě uživatelských práv a obsahuje pouze základní odkazy. Další navigace je pak realizována

v jednotlivých sekcích. K přihlášení, nebo odhlášení ze systému je uživateli poskytnuto přihlašovací okno (viz Obrázek 3.2) dostupné ze všech částí webu, které zároveň jednoduchou ikonou identifikuje identitu uživatele. V rámci obsahujícím mapové podklady jsou pak typické navigační prvky Google Maps API, kterým oproti klasickému zobrazení byly změněny pouze pozice.

### 3.2.2 Informační panel

K vizualizaci doplňujících informací slouží informační panel umístěný v pravé části webu (viz Obrázek 3.2). Jsou v něm zobrazeny především údaje týkající se vykreslené trasy (délka, celkové převýšení, oblíbenost a další). Obsahuje taky některé ovládací prvky mapy, které umožňují zobrazení některých doplňujících vrstev mapových podkladů. V některých případech slouží také jako okno s nápovědou pro lepší orientaci v různých částech webu.

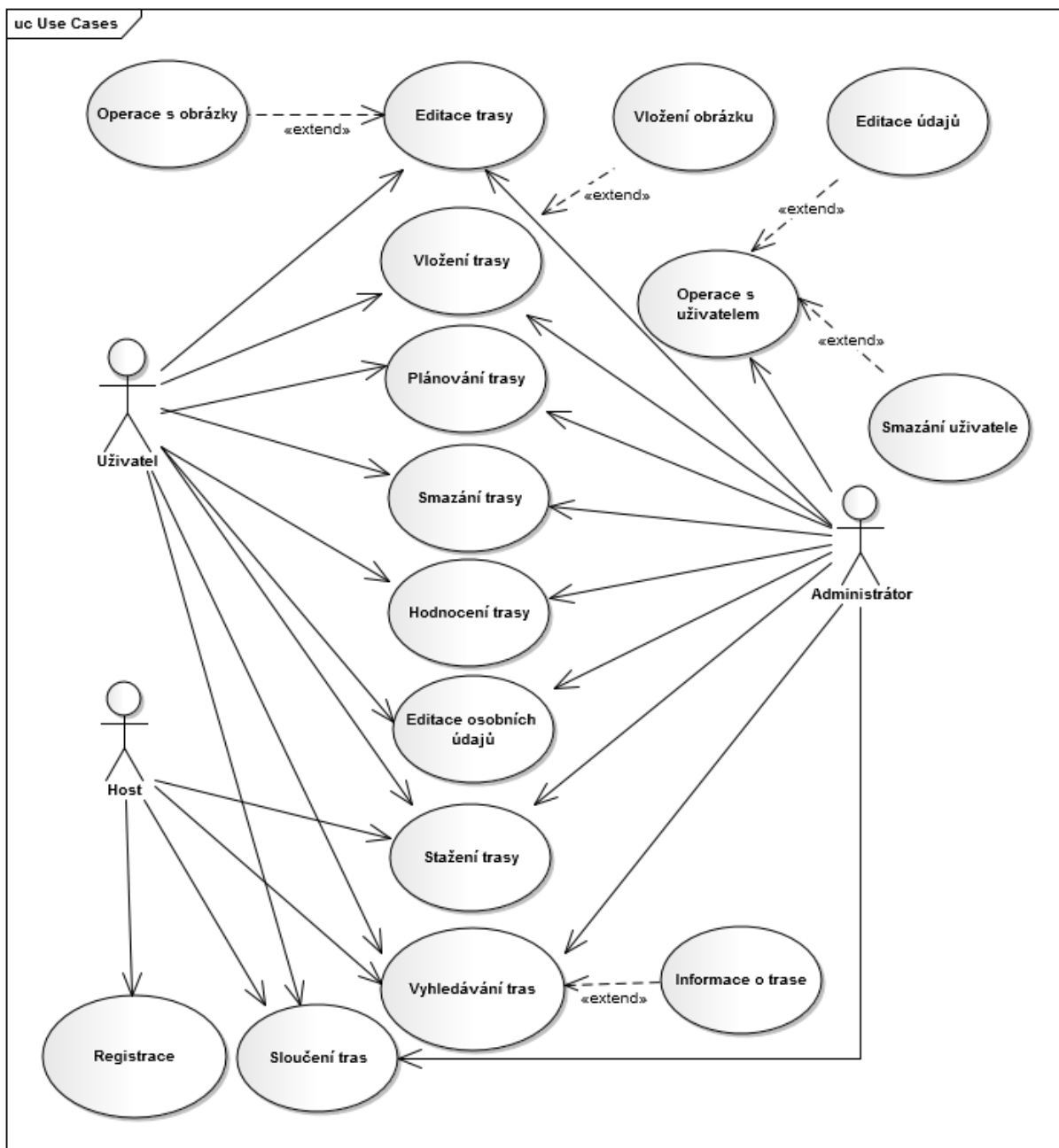


Obrázek 3.2: Grafické uživatelské rozhraní aplikace

## 3.3 Uživatelské funkce

Aplikace rozlišuje 3 typy uživatelů (host, přihlášený uživatel, administrátor). Základním cílem aplikace je umožnění co největšího počtu funkcí i neregistrovaným uživatelům a nenutit tak uživatele k zakládání uživatelského účtu. Uživatelé, kteří chtějí ze systému pouze získat informace v podobě nalezených tras a jejich vlastností, se obejdou bez uživatelského účtu. Nemůžou však trasy, vytvářet, plánovat ani hodnotit, tyto funkce jsou určeny pouze přihlášeným uživatelům. Systém tak zamezuje případné vytváření nesmyslných záznamů, především prostřednictvím automatických robotů, kteří procházejí, a indexují internetové stránky. Nicméně aby i nepřihlášený uživatel mohl nějakým způsobem sdílet názor na trasu, byla do aplikace zakomponována možnost přidávání komentářů prostřednictvím sociální sítě Facebook [10].

Posledním typem uživatelského účtu je typ administrátor. Jedná se o účet, který je určen především správcům aplikace a je obohacen o funkce pro správu tras, obrázků a v neposlední řadě i uživatelů. Možnosti všech uživatelských účtu podrobněji znázorňuje schéma případu užití (viz Obrázek 3.3).



Obrázek 3.3: Diagram případu užití

### 3.4 Slučování tras

Z vlastních zkušeností vím, že náročnější uživatel vyhledávající vhodnou turistickou trasu, nemusí být vždy spokojen pouze s jedinou trasou, ale rád by prošel více na sebe navazujících tras. Na takové uživatele byl brán zřetel, a proto byla aplikace obohacena o funkci sloučení tras. Dohromady lze sloučit až tři trasy. Uživatel si postupně vybere trasy, které a v jakém pořadí chce sloučit. Z těchto je

následně vygenerována jedna celistvá trasa, včetně všech dostupných informací a grafického znázornění výškového profilu. Výhodou spojení tras je hlavně možnost stažení KML souboru obsahujícího výslednou trasu, který lze následně nahrát do GPS navigace a využít přímo v terénu.

## 3.5 Plánování tras

Vytváření nových tras je umožněno pouze prostřednictvím nahrávání dat ze souborů. Manuální tvorba v rámci aplikace není povolena, ale pokud by si uživatel chtěl vygenerovat výškový profil, nebo zobrazit jiné užitečné informace zatím neexistující trasy, může použít funkci pro plánování tras.

Prostým klikáním kurzoru do mapy se umísťují značky, kterými probíhá plánovaná trasa a zároveň se taky dynamicky generuje výškový profil trasy. Výslednou trasu si lze stáhnout v KML souboru, který je dále možno použít v jiném prohlížeči KML souborů (např. Google Earth), nebo GPS přístroji podporujícím tento formát. Pokud by chtěl uživatel aplikaci obelhat a ihned po stažení trasy nahrát zpět do systému jako novou, bez nutných úprav se mu to nepovede. Aplikace si totiž pomocí několika značek zaznamenává, že se jedná o plán vytvořený v systému a neumožní tak uživateli následně trasu uložit.

## 3.6 Dodatečné funkce

Protože je dobré mít co nejvíce informací pohromadě na jednom místě a nenutit tak uživatele dohledávat další užitečné informace (např. informace o počasí) na jiných serverech, je aplikace obohacena o několik funkcí, které ocení náročnější uživatelé. Výhodou je, že tyto funkce jsou dostupné pro všechny, nezávisle na tom zda se jedná o přihlášeného, či ne přihlášeného návštěvníka webu.

### 3.6.1 Informace o počasí

Pro většinu turistů v době plánování výletu, je základním faktorem počasí, proto byla v aplikaci přidána možnost zobrazení vrstvy s těmito údaji (viz Obrázek 3.4). Tato vrstva obsahuje grafické znázornění počasí včetně teploty uváděné v stupních Celsia. Navíc disponuje předpovědí na nejbližší čtyři dny. Všechny údaje jsou poskytovány serverem weather.com, a jsou dostupné pro kteroukoliv oblast na Zemi.

V rámci rozšíření Google Maps API, poskytujícím údaje o počasí existuje taky vrstva obsahující vizuální zobrazení oblačnosti, které bylo taky využito v aplikaci. Pokud by ale jedna z vrstev uživateli překážela lze ji pomocí jednoduchého přepínače deaktivovat.



Obrázek 3.4: Zobrazení informací o počasí

### 3.6.2 Vrstva Panoramio

I přesto že každá trasa může disponovat téměř neomezeným množstvím fotografií vložených autorem trasy, nelze na jejich přítomnost spoléhat. Proto je v systému použita navíc vrstva zobrazující fotografie a obrázky serveru Panoramio.com, kde si každý uživatel může vytvořit účet a vkládat fotky nové. Tato vrstva je primárně vypnutá, protože obsahuje velké množství obrázků a překrývá tak větší část mapy, navíc na přístrojích disponujících pomalejším připojením k internetu dochází k zpomalení načítání mapových podkladů. Její zobrazení lze aktivovat v informačním panelu.

## 3.7 Administrátorský účet

Aplikace je vytvořena tak, aby pracovala s velkým počtem uživatelů a ještě větším počtem tras. U všech těchto záznamů nelze zajistit automatickou kontrolu pravosti nebo duplicity záznamu tras, z tohoto důvodu systém počítá s uživatelem typu administrátor, který tyto informace průběžně kontroluje a spravuje. Aplikace je však navržena tak aby správci jeho činnost co nejvíce usnadňovala.

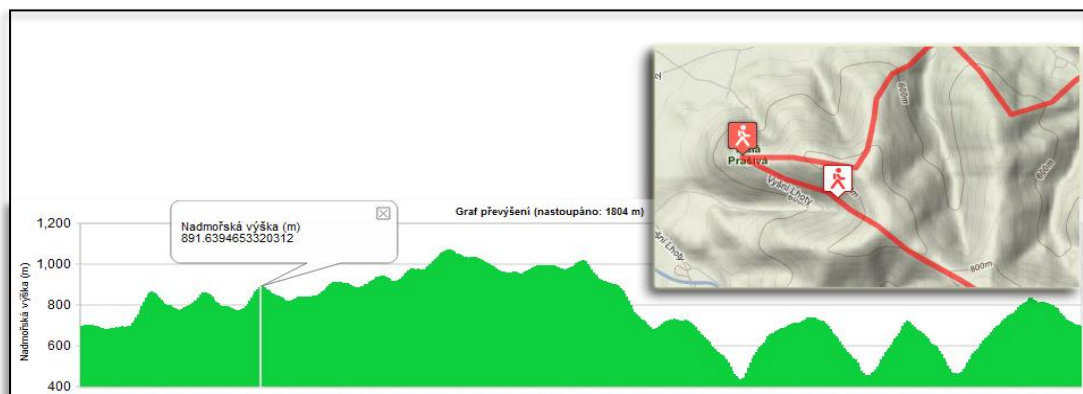
Uživatel přihlášený jako administrátor může využívat všechny standardní funkce a navíc je mu umožněno prohlížení všech uživatelských účtů včetně tras k nim přiřazeným. Záznamy o trasách může jakkoli upravovat a samozřejmě i mazat. Tuto činnost lze provádět jednotlivě nebo jako celek v rámci jednoho uživatele. Smazání určitého uživatele není v aplikaci umožněno, protože by pak nic nebránilo určitému uživateli vytvořit si účet znovu a pokračovat dále v jeho případné nežádané činnosti. Místo toho lze takovému uživateli účet deaktivovat. V rámci systému se už zobrazovat nebude a zároveň dojde k zablokování nové registrace pod jeho emailovým účtem.

## 3.8 Zobrazení tras

Aplikace je koncipována tak aby umožňovala zobrazení turistických tras v barevném provedení ekvivalentnímu turistickému značení trasy v terénu. Tato možnost pak umožňuje lepší orientaci v reálné trase.

V rámci trasy je pak obsažen ukazatel označující její začátek a případné ukazatele určující polohy míst pořízených snímků. Pro označení jednoznačné polohy výškového profilu v trase, je určen dynamický ukazatel, který je v interakci s vygenerovaným grafem převýšení (viz Obrázek 3.5).

K vykreslení tras byla použita třída `google.maps.Polyline` a ukazatelé označující místa na mapě třídou `google.maps.Marker` JavaScriptové knihovny Google Maps API.



Obrázek 3.5: Výškový profil v interakci s ukazatelem na mapě

## 3.9 Umístění fotografií do trasy

Ihned po vytvoření nové trasy je uživateli nabídnutá možnost nahrání fotografií pořízených během absolvování trasy. Aby ostatní uživatelé měli přehled o místech, kde byly snímky pořízeny, je nutné ke každému snímku přiřadit zeměpisné souřadnice. Během návrhu přicházely v úvahu tři způsoby jak tento problém vyřešit.

V současné době existují digitální fotoaparáty s integrovaným GPS modulem, který snímkům přiřazuje GPS souřadnice během zhotovení snímku. Toto je nejjednodušší a nejpřesnější metoda jak tento údaj získat. Bohužel fotoaparát obsahující GPS modul zatím vlastní malé procento lidí, proto jsem tento způsob nevyužil, ale dalo by se ho v budoucnu implementovat.

Další možností bylo přiřazení snímku podle času pořízení. Tento čas by se porovnal s časy zaznamenaných bodů pomocí GPS přijímače a následně přiřadil k souřadnicím s nejméně odpovídajícím časem. Tento způsob by šlo použít za předpokladu, že uživatel má korektně nastaven čas ve fotoaparátu. Protože KML soubor neobsahuje záznamy o časech zaznamenání zeměpisných souřadnic, nemohl být tento způsob použit.

Poslední a zároveň implementovanou možností v rámci aplikace, je manuální označení místa na trase. Aby však nedocházelo k označování míst mimo trasu a nahrávání tak fotografií s ní nesouvisejících, je povoleno získávat souřadnice pouze z bodů nacházejících se na trase (viz Obrázek 3.6).



Obrázek 3.6: Nahrávání fotografií

## 4 Implementace

Kapitola obsahuje některé implementační problémy a zajímavosti. Je zde rozebrána celá problematika načítání tras z externích souborů, včetně výpočtů které se provádějí před uložením nové trasy do systému. Také je vysvětlen princip hodnocení tras a možnost navigace z aktuální uživatelské pozice do začátku trasy. Je zde taky vysvětlen způsob filtrování tras, způsob zobrazení fotografií v mapě, nebo možnost oznámení nevhodné trasy. Na konci kapitoly je pak blíže představeno generování KML souborů.

### 4.1 Vývojové prostředí

Větší část aplikace byla napsána ve Frameworku Nette. Pro tvorbu programů v tomto jazyce je vhodné použít vývojové prostředí NetBeans IDE ve spojení s dostupným doplňkem [16], který je volně ke stažení a výrazně usnadňuje práci v tomto editoru.

### 4.2 Načítání trasy ze souboru

Základním zdrojem dat jsou KML soubory obsahující záznamy tras, více o KML souborech se dočtete v kapitole 2.4. Jelikož se gramatikou nijak neliší od XML souborů, lze pro jejich syntaktické rozebrání použít, některý z mnoha existujících nástrojů. Osobně jsem zvolil rozšíření v rámci PHP 5 SimpleXML, které po aplikování na načtený soubor vytvoří objekt s vlastnostmi ekvivalentními značkám v KML souboru.

Ne všechny aplikace a přístroje generující KML soubory respektují standard zápisu struktury KML souboru [5]. Aplikace je však implementována tak aby se pokusila nejdříve načíst soubor jako validní a při neúspěchu se pokusí načíst typ struktury sice nevalidní, ale taky často se vyskytující.

Rozšíření PostGIS podle dokumentace [11] umí pomocí funkce `ST_GeomFromKML` zpracovat fragmenty KML dokumentu. Bohužel tato funkce nefunguje správně, nepočítá totiž se zápisem souřadnic v pořadí stanoveným standardem (zeměpisná délka, šířka a výška), namísto toho zpracovává data v prohozeném pořadí (zeměpisná šířka, délka a výška). Dochází tak k zápisu nesprávných údajů. Proto bylo nutné před každým uložením trasy do databáze tyto souřadnice prohodit, k tomuto účelu složí metoda `actionReadKml` třídy `CreateRoutePresenter`.

#### 4.2.1 Doplnující výpočty

Pro rychlejší vyhledávání v uložených trasách a zároveň menší zatížení databáze, bylo vhodné některé informace vypočítat během načítání nové trasy.

Jedním z pomocných údajů je výpočet celkového převýšení. Protože informace o zeměpisné výšce v KML souboru jsou nepovinné, není vhodné spoléhat na existenci tohoto údaje v rámci nahrávaného dokumentu. A proto pro výpočet míry převýšení, jsou použity údaje poskytnuté knihovnou Google Maps API. Navíc se takto zamezí nahrávání tras s nedůvěryhodným převýšením. Délky turistických tras jsou většinou v řádech několika kilometrů, proto postačí výsledek převýšení s přesností na jednotky metrů. Aby však nedocházelo k velkému zkreslování údajů při zaokrouhlování, jsou všechny kladné rozdíly údajů o převýšení nejdříve sečteny jako reálná čísla a až konečný výsledek je zaokrouhlen na číslo celé. Dochází tak k nepřesnosti maximálně 0,5m. GPS přístroje

dosahují přesnosti 1–15 m, proto se zaokrouhlovací chybou není třeba dále zabývat. Tento výpočet probíhá na straně klienta, a nedochází tak k zatížení serveru. Úsek kódu popisující tuto činnost je uveden zde:

```

el.getElevationAlongPath(pathRequest, calculateElevation);
function calculateElevation(results, status) {
  if (status == google.maps.ElevationStatus.OK) {
    elevs = results;
    for (var i = 0; i < results.length; i++) {
      if(i > 0){
        var elevationTemp = elevs[i].elevation - elevs[i-1].elevation
      }
      if(elevationTemp > 0){
        elevation += elevationTemp;
      }
    }
    ele = Math.round(elevation);
  }
}

```

Další důležitou operací prováděnou v rámci načítání KML souborů, je dodatečný výpočet celkové délky trasy. K získání tohoto údaje byla prvotně využita funkce nadstavby PostGIS (`ST_Length3D_Spheroid`), které stačí předat jako parametr souřadnice trasy ve formátu EWKT a typ rotačního elipsoidu. Během testování aplikace jsem bohužel odhalil nepřesnosti oproti reálným délkám, které na delších trasách činily až několik set metrů. Více o testování aplikace se dočtete v kapitole 5. Proto jsem se rozhodl tento údaj vypočítat pomocí vlastní funkce, využívající algoritmus Haversinovy formule (viz kapitola 2.).

Posledním výpočtem před uložením celé trasy do databáze, je výpočet obtížnosti trasy. Některé servery udržující základní informace o turistických trasách poskytují tuto informaci na základě předem definované hodnoty uživatelem, který trasu vytvořil. Avšak hranice, kdy je trasa lehká nebo obtížná, může být pro každého jednotlivce individuální, navíc obtížnost bývá často ovlivněna aktuálním počasím nebo věkem turisty. Z těchto důvodů si aplikace obtížnost trasy počítá sama (viz Rovnice 4.1).

$$x = \text{typ povrchu} \times \frac{\text{délka trasy} + (\text{úhrnné stoupání} \times 4)}{1000}$$

**Rovnice 4.1: Výpočet obtížnosti trasy**

Kde délka a úhrnné stoupání trasy jsou uvedeny v metrech a typ povrchu může nabývat jednu ze tří hodnot:

- Zpevněný povrch = 0.5
- Částečně zpevněný povrch = 0.75
- Nezpevněný povrch = 1

Výsledné hodnotě je pak dle následující tabulky (viz Tabulka 4.1) přiřazena míra obtížnosti trasy.

X	0 – 7	7,01 – 18	> 18
Obtížnost trasy	lehká	středně těžká	náročná

**Tabulka 4.1: Přiřazení míry obtížnosti**

## 4.3 Hodnocení trasy

Pro rozlišení kvalitních, nebo oblíbených tras mezi uživateli slouží hodnocení, které je založeno na jednoduchém principu. Každý přihlášený uživatel může hodnotit trasu jenom jednou, prostřednictvím tlačítka `to_se_mi_líbí`, ale kdykoliv může své rozhodnutí vrátit zpět. Součet všech hlasů se automaticky sčítá a následně ovlivňuje vyhledávání. Protože je možné hlasovat pouze kladně, měly by během filtrování procentuálně větší šanci na přednější pozice záznamy existující v systému déle. Oproti tomu nové trasy by se zobrazovaly na konci seznamu. Toto nežádoucí chování je vyřešeno jednoduchým, algoritmem.

Systém si průběžně ukládá i počet zobrazení trasy z každého uživatelského účtu. Z těchto dvou hodnot je průběžně vypočítáván kvocient oblíbenosti (viz Rovnice 4.2), který je směrodatný při vyhledávání tras. To znamená, že i nově vložená trasa může dosáhnout vyšší pozice a naopak neoblíbená trasa, která je uložena v systému déle pozice nižší.

$$Q = 100 \times \frac{\text{počet kladných hlasů}}{\text{počet zobrazení}}$$

Rovnice 4.2: Výpočet kvocientu oblíbenosti

## 4.4 Navigace do místa začátku trasy

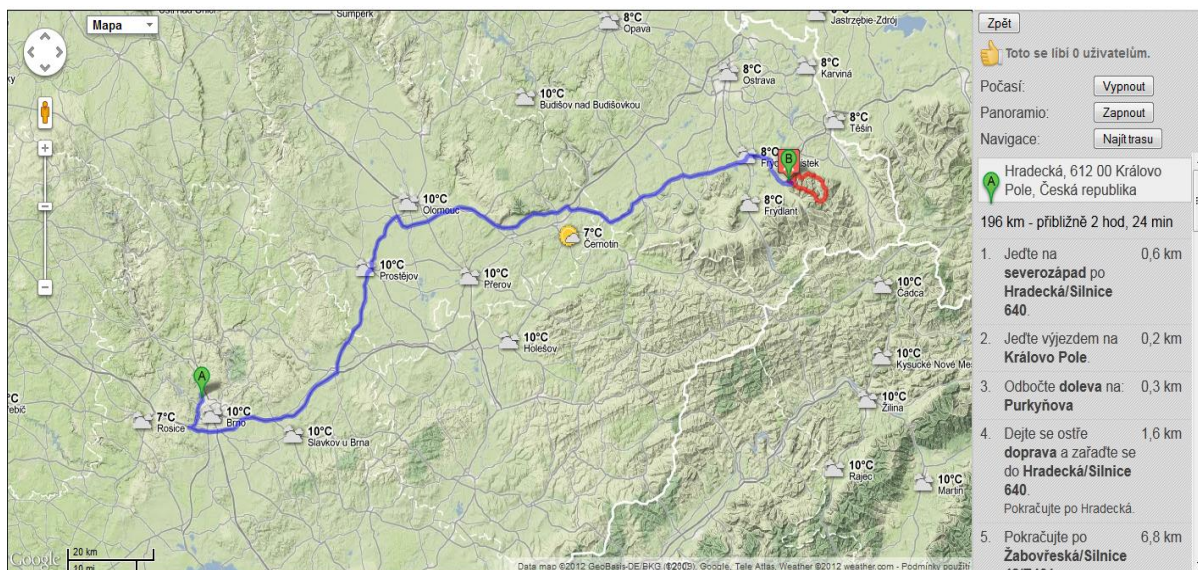
Uživatel, který si vybere vhodnou trasu pro absolvování, musí ještě vyřešit způsob dopravy do místa začátku trasy. Pokud se navíc jedná o lokalitu, kterou nikdy předtím nenavštívil, může mít s orientací problém.

Pro takové uživatele cestující vlastním dopravním prostředkem, je určena funkce pro nalezení optimální trasy z místa kde se uživatel právě nachází až do začátečního bodu trasy (viz Obrázek 4.1). Uživatel se mimo to dozví i vzdálenost a předpokládanou dobu jízdy. K implementaci této funkce byla využita služba Directions Service [14] společnosti Google a Geolocation API [15] které je obsaženo v rámci HTML5.

Pomocí Geolocation můžeme zjistit geografickou polohu uživatele prostřednictvím internetového prohlížeče podporujícího HTML5. Ten může polohu zařízení získat z IP adresy, GPS souřadnic integrovaného GPS modulu, ID buňky GSM/CDMA nebo RFID<sup>5</sup>. Přesnost určení polohy se různí podle způsobu získání této informace. Pokud však uživatel nedisponuje internetovým prohlížečem, který technologii HTML5 podporuje, je o tom informován a následně vyzván aby svou polohu zadal manuálně.

---

<sup>5</sup> Identifikace na rádiové frekvenci.




Obrázek 4.1: Zobrazení navigace do místa začátku trasy

## 4.5 Vyhledávání tras

Aplikace počítá s velkým počtem tras, a proto by se uživatelé bez vyhledávacího mechanismu neobešli. Vyhledávat lze podle mnoha kritérií zároveň a tudíž dostat výsledek co nejbližší uživatelským požadavkům. Pokud by i přesto výsledkem bylo velké množství tras, je možno filtrovat trasy ještě v rámci nalezených výsledků (viz Obrázek 4.2).

Veškerá činnost filtrování probíhá, pomocí specifických dotazů v databázi. Aplikace pouze zpracovává vyhledávací formulář a konkrétně tuto činnost vykonává třída `SearchPresenter`, která své výsledky předává třídě `SearchModel` a ta dále komunikuje s databází.

Vyhledávat lze podle jména, délky, převýšení, oblíbenosti, typu povrchu a obtížnosti trasy. V rámci administrátorského účtu lze pak ještě vyhledávat podle ID trasy.

 <b>NALEZENÉ TRASY</b>				
Název ▾	Délka ▾	Převýšení ▾	Povrch ▾	Operace
Vrch				<b>Použít</b> 🗄
Babí Vrch	19753 km	703 m	Částečně zpevněný	<b>Zobrazit</b>
Nový Vrch	15829 km	729 m	Zpevněný	<b>Zobrazit</b>
Černé Vrchy	23753 km	1125 m	Zpevněný	<b>Zobrazit</b>
Vrchní stráně	10884 km	423 m	Zpevněný	<b>Zobrazit</b>

Položek 1 - 4 z 4 | Zobrazit:  **Změnit** 🔄

Obrázek 4.2: Výsledky vyhledávání, s další možností filtrace a řazení

## 4.6 Zobrazení obrázků

Obrázky přiřazené k trase, bylo potřeba nějakým vhodným způsobem zobrazit. Prvotní myšlenkou byla samostatná galerie, umístěná např. pod oknem obsahujícím mapu. Toto řešení by však při větším počtu obrázků neumožňovalo přehledné zobrazení místa pořízení snímku a nutilo tak uživatele rolovat oknem prohlížeče.

Lepším řešením bylo zobrazení obrázku přímo v mapě pomocí metody `infoWindow`, kterou poskytuje Google Maps API. Tento způsob byl výhodný za předpokladu, že fotografie budou prezentovány v nízkém rozlišení (maximálně 480x360). Proto byla tato možnost ještě dále modifikována. K finálnímu řešení bylo využito ShadowBox API<sup>6</sup>. Každou fotografii pak reprezentuje jedna ikona umístěná na trase, která po odchycení událostí metodou `event.addListener` zavolá třídu knihovny ShadowBox API obsluhující vykreslování obrázku (`Shadowbox.open`). Vše se děje pomocí jazyka JavaScript, který je interpretován na straně klienta a tudíž nedochází k výraznému zatížení serveru.

```
google.maps.event.addListener(picmarker$i, 'click', function() {
    Shadowbox.open({
        player:    'img',
        content:   '../photos/$this->routeId/$name$type',
        title:     '$note'
    });
});
```

## 4.7 Oznámení nevhodné trasy

Systém zatím nijak neověřuje autentičnost modelu trasy, a protože se nelze v tomto ohledu spoléhat na uživatele mohlo by dojít do stádia, že systém bude obsahovat více zavádějících tras než těch užitečných. Tento jev by mohl znehodnotit celou aplikaci a odradit některé uživatele od jeho používání. Z tohoto důvodu je zde možnost nahlášení nevhodné trasy, která formou emailu ohlásí správci upozornění na výskyt nevhodné trasy.

K odeslání emailu byla využita třída `Message` Frameworku `Nette`, která výrazně ulehčuje implementaci. Hlavní výhodou této třídy je využití šablonovacího systému `Latte`, který umožňuje použití předem připravených šablony, stačí pak pouze předat parametry šablony a odeslat email.

## 4.8 Generování KML souborů

Poslední zajímavou částí v rámci implementace, je způsob generování KML souborů. Tato činnost je obsluhována třídou `KmlPresenter`, která rozlišuje tři možnosti zpracování dat a podle toho zvolí odpovídající způsob tvorby KML souboru. Rozdíly ve zpracování dat jsou určeny typem trasy, který má být uložen.

Pokud se jedná o standardní trasu, je situace nejjednodušší. Třída pak očekává jediný povinný parametr (ID trasy), podle kterého vyhledá odpovídající záznam v databázi a pomocí technologie `DOM` (*Document Object Model*) vytvoří jednotlivé objekty KML dokumentu, které naplní vyhledanými informacemi. Kromě dat získaných z databáze, jsou do objektů uloženy styly, které lze

---

<sup>6</sup> JavaScriptová knihovna určena k prezentaci obrázku, videí a html obsahu v okně prohlížeče.

uplatnit v některých prohlížečích zobrazujících soubory ve formátu KML (např. Google Earth). Následně pomocí funkce `saveXML()` vytvoří jeden objekt zapouzdřující všechny předchozí objekty. Pak už jen nastaví hlavičkové meta tagy souboru a nabídne uživateli výsledný KML dokument.

Dalším zpracovávaným typem trasy je turistická stezka, která byla vytvořena v systému a je určena pouze k plánování. Činnost třídy `KmlPresenter` se v takovém případě výrazně neliší od předchozího případu, jediný rozdíl je v obsahu některých značek. Systém si tak značí v několika místech, že se jedná o pouhý plán nikoli o reálnou trasu. Tyto značky jsou pak během vytváření nové trasy detekovány a činnost ukládání je ukončena.

Stezka, která vznikla sloučením několikátí tras dohromady, je posledním typem který umí zpracovat třída `KmlPresenter`. Tento typ trasy není uložen v databázi, a proto musí být vytvořen znovu v rámci této třídy. K získání dat potřebuje pouze seznam ID tras zapsaných v takovém pořadí, v jakém mají být spojeny. Během vytváření souboru dochází také k označení dokumentu, který vznikl spojením tras v rámci aplikace.

Příklad výsledného KML souboru, obsahující trasu a některé další informace (např. název trasy), lze najít v příloze A.

## 5 Testování

Systém byl průběžně testován již během implementace. Některé testy byly prováděny pomocí různých programů, pracujících s geometrickými objekty (např. Google Earth). Aplikace byla taky testována na základě reálných dat získaných mobilním telefonem disponujícím GPS modulem a navigačním přístrojem. Do testování jsem taky zapojil několik přátel, kteří často absolvují turistické výlety a mohli by v budoucnu takový systém využívat.

### 5.1 Testování databáze

V začátcích práce byl kladen důraz hlavně na testování databáze a konkrétně na funkce pracující s typy geometry, které jsou součástí rozšíření PostGIS. Testovány byly především funkce umožňující čtení souborů přímo z formátu KML, kde bylo zjištěno chybné chování, u kterého docházelo k opačnému zápisu zeměpisných souřadnic. Výsledkem pak byly trasy zobrazující se mimo očekávanou oblast.

Další funkcí PostGISu s kterou bylo při návrhu počítáno, byla funkce pro výpočet celkové délky trasy. Po několika testech srovnávání získaných výsledků a reálné délky trasy ve skutečnosti jsem dospěl k závěru, že tato funkce dosahuje relativně velkých odchylek, proto na konec nebyla použita.

Dále byla databáze testována na rychlost vyhledávání ve velkém počtu tras obsahujících větší množství zaznamenaných bodů (řádově stovky až tisíce), zde bylo dosaženo upokojujících výsledků.

### 5.2 Testování webového rozhraní

Účelem testování webového rozhraní bylo dosažení stejného nebo mírně se lišícího vzhledu ve webových prohlížečích, ale také i vlastnosti aplikace. Testy byly prováděny v prohlížečích Firefox 12.0, Chrome 18.0.1025 a Internet Explorer 9.0.1. Prohlížeče Firefox a Chrome dosahovaly velmi podobných výsledků, bohužel pro správné chování v prohlížeči Internet Explorer, musel být napsán nový soubor obsahující stylování, a i přesto se zobrazení mírně liší. Na použití aplikace tato skutečnost neměla žádný vliv, proto jsem se tímto dále nezabýval.

### 5.3 Testování aplikace

Nejvíce testů bylo prováděno nad vrstvou aplikační. Systém byl testován především na neočekávané chování v podobě přístupů do sekcí, které nejsou primárně přístupné jednotlivým uživatelům (např. zobrazení všech uživatelů z nepřihlášeného účtu, smazání cizí trasy, atd.). Také bylo testováno zadáváním html tagů do polí určených k poznámkám, pokusy o obelhání systému hodnocení trasy a nahrávání nevalidních KML souborů.

V rámci testování aplikace, byl také kladen důraz na správnost a účinnost provádění JavaScriptového kódu na straně klienta. K tomuto testování byl použit především doplněk FireBug internetového prohlížeče Firefox. Jedná se o přehledný debugovací nástroj, který automaticky detekuje nedefinované funkce a jiné chyby.

Poslední testy byly prováděny především pomocí nahrávání reálných dat získaných z terénu, a následným porovnáváním se zobrazenými a vypočtenými výsledky v aplikaci.

Měření délky jsem prováděl na bicyklu, kde byl instalovaný tachometr. Jako pomocný nástroj posloužila mobilní aplikace Endomondo a samotný záznam GPS souřadnic do souboru KML prováděla GPS navigace značky Garmin. Samotné výsledky těchto testů byly nad míru uspokojivé, obsahovaly pouze drobné odchylky, ale ty mohly být způsobeny nepřesností navigace. Délka trasy více méně odpovídala realitě, u trasy o délce 24652 m se jednalo o chybu 254 m. Také celkové převýšení se výrazně nelišilo, od naměřených údajů.

Abych otestoval a případně i upravil algoritmus pro výpočet obtížnosti trasy, bylo potřeba měření provádět také pěší chůzí. K tomuto testu jsem využil ochoty mých přátel a poprosil je o účast v testování. Všichni absolvovali tři různě dlouhé trasy, s různými výškovými profily. Pokaždé byl zvolen jiný typ povrchu (asfaltový povrch, šterkový povrch, půda místy s kamením a vystupujícími kořeny). Vždy na konci absolvované stezky se každý zúčastněný vyjádřil, jak náročná pro něj trasa byla. Na základě těchto údajů jsem odvodil výsledný vzorec pro výpočet obtížnosti (viz Rovnice 4.1). Pro zdokonalení této rovnice by bylo potřeba provést ještě více testů a měření. Navíc by bylo vhodné do výpočtu zakomponovat údaje s aktuálním počasím, věkem účastníka a jiné faktory. Pro základní určení obtížnosti, však tento tvar rovnice postačí.

Zatím se mi nepodařilo najít vhodný český cenově výhodný webhosting, který by podporoval všechny mnou využívané služby, zvláště pak nadstavbu PostGIS 1.5.3. Proto po celou dobu testování byl systém včetně databáze provozován na lokálním disku (anglicky *localhost*) počítače z kterého byl spouštěn. Provoz byl realizován pomocí http serveru Apache 2.2.22 a databázového systému PostgreSQL 9.1.1.

## 6 Závěr

Cílem této práce bylo navrhnout a implementovat aplikaci umožňující sdílení turistických tras pomocí webového serveru. V současné době existuje velmi malé procento systémů s podobným zaměřením. Tyto systémy mají několik nedostatků a neposkytují tak uživatelům takové množství funkcí a možností jako výsledná aplikace.

Hlavním principem při návrhu byla jednoduchost a přehlednost výsledné aplikace, která zároveň poskytuje všechny informace potřebné k nalezení a následné absolvování turistické trasy. Systém některé tyto informace poskytuje ze svého zdroje dat (databáze) a zbylé získává z jiných webových serverů.

Uživatelům nabízí základní informace o trase (délka, celkové převýšení, aktuální počasí, obtížnost trasy, oblíbenost), včetně jejího barevného vykreslení, které je ekvivalentní s reálným turistickým značením v terénu. Mezi užitečné funkce lze zařadit spojování tras, export do souboru ve formátu KML, hodnocení tras, rozšířené vyhledávání a možnost navigace do místa začátku trasy. Uživatelům je taky umožněno vkládání fotografií, které lze přiřadit k vybraným bodům na trase.

Aplikace byla vyvíjena pomocí Frameworku Nette, Google Maps API, ShadowBox API, knihovny jQuery, databázové knihovny dibi a databázového systému PostgreSQL včetně nadstavby PostGIS.

Výsledná aplikace byla několikrát testována různými způsoby zaměřujícími se na konkrétní části systému, kde získané výsledky byly relativně upokojivé.

### 6.1 Přínos práce

Práce splnila mé očekávání, protože se mi podařilo implementovat veškerou navrženou funkčnost. Zatím neexistuje žádná aplikace sdružující turistické trasy a zároveň poskytující některé dodatečné funkce jako výsledná aplikace, proto by mohla být přínosem pro některé nadšence turistiky. Osobně mi tato práce přinesla hodně nových zkušeností, rozšiřujících mé znalosti o prvky vyskytující se v oblasti geografických informačních systémů. Nikdy předtím jsem nepracoval s databází PostgreSQL a zvláště ne s její nadstavbou PostGIS, díky které jsem se naučil pracovat s geometrickými datovými typy v rámci databáze. Přínosem byla také práce s Google Maps API, které jsem předtím využíval pouze okrajově nic netušíc o jeho potenciálu. Taky jsem se dozvěděl pár nových informací o turistickém značení a polohovém systému GPS.

Během implementace jsem narazil na řadu problémů, jedním z nich byl chybný výpočet vzdálenosti trasy pomocí vestavěných funkcí PostGISu, proto jsem se problematikou zabýval hlouběji. Dozvěděl jsem se tak, jakými algoritmy se tyto vzdálenosti počítají. Jeden z nich jsem pak následně v rámci aplikace použil.

### 6.2 Další možná rozšíření

I když byla v aplikaci implementována různá rozšíření, které obohacují její funkčnost o jedinečné vlastnosti, během implementace mě napadla řada dalších možností jak aplikaci dále rozšířit. A však pro další taková rozšíření by bylo potřeba více času na návrh a implementaci, nebo alespoň více programátorů pracujících v týmu.

Zde jsou uvedena některá z dalších rozšíření:

- Podpora většího množství formátů – aplikace zatím umí pracovat pouze s formátem KML, který je v dnešní době často používaný, navíc pro převod jiného formátu do KML existuje řada konvertorů. Avšak bylo by více než vhodné rozšířit aplikaci tak aby podporovala i jiné typy souborů, do kterých lze zaznamenávat trasy, například GPX nebo WPT.
- Automatizace spojování tras – slučování tras v rámci aplikace je implementováno pouze manuálně, a proto by šlo aplikaci upravit tak aby v síti tras vyhledávala spojení dvou bodů automaticky, na základě předem stanovených požadavků (např. nalezení nejkratšího spojení)
- Automatické přiřazení zeměpisných souřadnic k fotografiím – souřadnice k snímkům by bylo vhodné přiřazovat automaticky, nebo alespoň nabídnout uživateli předpokládanou pozici pořízení snímku. Tyto souřadnice by mohly být získány, buď ze snímků pořízených fotoaparátem disponujícím GPS modulem, nebo na základě času pořízení snímků a časem zaznamenání bodu trasy. Druhá možnost však vyžaduje první zde uvedené rozšíření a to podporu jiného formátu, který obsahuje zároveň i časy pořízení GPS souřadnic.
- Mobilní aplikace – pro ukládání či prohlížení tras přímo v terénu by přišla vhod aplikace pro některý z chytrých mobilních telefonů s integrovaným GPS modulem. Taková mobilní aplikace by mohla v reálném čase ukládat data přímo do databáze a zároveň informovat uživatele o aktuální absolvované délce, převýšení, nebo i předpovědi počasí na nejbližších pár hodin.

# Literatura

- [1] EL-RABBANY, Ahmed. *Introduction to GPS: the Global Positioning System*. 2nd ed. Boston, MA: Artech House, c2006, 210 s. ISBN 978-159-6930-162.
- [2] API Mapy.cz. *JsDoc Reference* [online]. 2012 [cit. 2012-05-07]. Dostupné z: <<http://api4.mapy.cz/doc/index.html>>
- [3] Street View. *Street View* [online]. 2012 [cit. 2012-05-07]. Dostupné z: <<http://maps.google.com/intl/en/help/maps/streetview/index.html>>
- [4] TREGONING, Paul a C RIZOS. GODDARD SPACE FLIGHT CENTER. *Dynamic planet: monitoring and understanding a dynamic planet with geodetic and oceanographic tools : IAG symposium, Cairns, Australia, 22-26 August 2005*. New York: Springer, c2007, 909 s. ISBN 35-404-9349-2.
- [5] *Open Geospatial Consortium* [online]. © 1994 - 2012 [cit. 2012-05-08]. Dostupné z: <<http://www.opengeospatial.org>>
- [6] WORSLEY, John C a Joshua D DRAKE. *Practical PostgreSQL*. Sebastopol, CA: O'Reilly, c2002, 619 s. ISBN 15-659-2846-6.
- [7] *BitNami :: WAPPStack* [online]. 2012 [cit. 2012-05-08]. Dostupné z: <<http://bitnami.org/stack/wappstack>>
- [8] RAYMOND A. SERWAY, Raymond A. Chris Vuille. *College physics*. 8th ed. Belmont, Calif: Brooks/Cole, 2008. ISBN 978-049-5554-752.
- [9] DANĚK, Petr. Velký test PHP frameworků. [online]. 2008 [cit. 2012-05-09]. Dostupné z: <<http://www.root.cz/clanky/velky-test-php-frameworku-zend-nette-php-a-ror>>
- [10] *Facebook* [online]. © 2012 [cit. 2012-05-10]. Dostupné z: <<http://www.facebook.com>>
- [11] *PostGIS 2.0.0 Manual* [online]. 2012 [cit. 2012-05-10]. Dostupné z: <<http://postgis.refractor.net/documentation/manual-2.0>>
- [12] ADAM WOLSKI, Wojciech Depta. *Loksodroma i ortodroma*. Wyd. 2 popr., rzut 1. Szczecin: Dział Wydawnictw WSM, 1994. ISBN 978-838-6494-033.
- [13] *Math Formulas and Tables*. Boston: MobileReference.com, 2007. ISBN 978-160-5010-816.
- [14] Directions Service. *Google Developers* [online]. 2012 [cit. 2012-05-12]. Dostupné z: <<https://developers.google.com/maps/documentation/javascript/directions>>

- [15] HOLDENER, Anthony T. *HTML5 Geolocation*. Sebastopol, CA: O'Reilly, 2011, 95 s. ISBN 14-493-0472-9.
- [16] *Google Maps JavaScript API v3* [online]. 2012 [cit. 2012-05-14]. Dostupné z: <<https://developers.google.com/maps/documentation/javascript/>>
- [17] Netbeans-plugin. *GitHub* [online]. © 2012 [cit. 2012-05-14]. Dostupné z: <<https://github.com/nette/netbeans-plugin>>
- [18] World Geodetic System. *Wikipedia, the free encyclopedia* [online]. 2012 [cit. 2012-05-15]. Dostupné z: <[http://en.wikipedia.org/wiki/World\\_Geodetic\\_System](http://en.wikipedia.org/wiki/World_Geodetic_System)>
- [19] Turistické značky a značení turistických tras. *AbcTuristiky* [online]. © 2004–2012 [cit. 2012-05-15]. Dostupné z: <<http://turistika.abchistory.cz/turisticke-znaceni.htm>>
- [20] *GPSvisualizer* [online]. ©2003-2012 [cit. 2012-05-15]. Dostupné z: <<http://www.gpsvisualizer.com/>>
- [21] OBE, Regina a Leo HSU. *PostGIS in action*. London: Pearson Education [distributor], c2011, 492 s. ISBN 19-351-8226-9.

# Seznam obrázků

Obrázek 2.1: Kruhové dráhy družic.....	4
Obrázek 2.2: Protnutí průsečíků 3 koulí.....	5
Obrázek 2.3: Struktura a funkce jazyka KML.....	8
Obrázek 2.4: Hierarchie datového typu <i>geometry</i> .....	9
Obrázek 2.5: Architektura Model-View-Controller.....	10
Obrázek 3.1: Návrh databáze.....	12
Obrázek 3.2: Grafické uživatelské rozhraní aplikace.....	13
Obrázek 3.3: Diagram případu užití.....	14
Obrázek 3.4: Zobrazení informací o počasí.....	15
Obrázek 3.5: Výškový profil v interakci s ukazatelem na mapě.....	16
Obrázek 3.6: Nahrávání fotografií.....	17
Obrázek 4.1: Zobrazení navigace do místa začátku trasy.....	21
Obrázek 4.2: Výsledky vyhledávání, s další možností filtrace a řazení.....	21

# Seznam rovnic

Rovnice 2.1: Výpočet 3D polohy .....	5
Rovnice 2.2: Výpočet délky ortodromy.....	5
Rovnice 2.3: Haversinova formule .....	6
Rovnice 4.1: Výpočet obtížnosti trasy.....	19
Rovnice 4.2: Výpočet kvocientu oblíbenosti.....	20

# Seznam příloh

**Příloha A.** Ukázka vygenerovaného souboru KML.

**Příloha B.** CD se zdrojovými texty aplikace a testovacími daty, SQL skript pro vytvoření databáze a programovou dokumentaci

# Příloha A – ukázka souboru KML

```
<?xml version="1.0" encoding="utf-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <name>TuristGuide_Babí- Vrch.kml</name>
    <StyleMap id="msn">
      <Pair>
        <key>normal</key>
        <styleUrl>#sn</styleUrl>
      </Pair>
    </StyleMap>
    <Style id="sn">
      <LineStyle>
        <color>ff5858ff</color>
        <width>5</width>
      </LineStyle>
    </Style>
    <Placemark>
      <name>Babí- Vrch</name>
      <styleUrl>#msn</styleUrl>
      <LineString>
        <tessellate>1</tessellate>
        <coordinates>
          18.63778,49.61217,0
          18.63668,49.61186,0
          18.63611,49.61163,0
          18.63489,49.6108,0
          18.6344,49.61031,0
          18.63401,49.60971,0
          18.63305,49.60744,0
          18.63212,49.60698,0
          18.6291,49.60511,0
          18.62894,49.60488,0
          18.62892,49.60467,0
          18.62898,49.60442,0
          18.62923,49.60388,0
          18.62923,49.60388,0
          18.70674,49.54775,0
        </coordinates>
      </LineString>
    </Placemark>
  </Document>
</kml>
```