Digitálne mapy

Dokumentácia k tímovému projektu

**Tím číslo:** 2

**Vedúci tímu:** Ing. Peter Bartalos

**Odbor:** Softvérové inžinierstvo

**Šk. rok:** 2009/2010

Bc. Abelovský Peter

Bc. Bartal Kamil

Bc. Basár Peter

Bc. Hraško Marián

Bc. Torda Dušan

Bc. Zdechovan Ján

# Analýza problému

## Digitálne mapy

Digitálne mapy, ich spracovanie a poskytnutie používateľom je v súčasnosti veľmi obľúbená, ale zároveň aj progresívna oblasť, pretože zatiaľ neboli využité všetky možnosti, ktoré digitálne mapy ponúkajú. Digitálne mapy majú čoraz širšie využitie a to v rôznych oblastiach. Prostredníctvom mobilných zariadení k nim máme prístup prakticky kdekoľvek, a preto by nám mali poskytovať čo najviac informácií o danej oblasti a objektoch. Existujúce riešenia sú však často obmedzené len na to, čo im poskytuje používané Google API a neponúkajú žiadnu pridanú hodnotu. Taktiež v nich absentuje dostatočná personalizácia digitálnej mapy pre jednotlivých užívateľov. Mapa sa javí pre všetkých užívateľov rovnaká, neprispôsobuje sa vlastnostiam a charakteristike užívateľa. Užívateľ nemá možnosť využiť celý potenciál digitálnych máp, ktorý je podľa nás veľmi veľký.

## Prínos riešenia – motivácia

Cieľom našej práce je priniesť do tejto oblasti nové nápady, ktoré sme získali pri používaní digitálnych máp pre vlastné účely a máme pocit, že väčšina z nich by priniesla používateľom nové možnosti využitia digitálnych máp a pomohla by im cítiť sa pri práci s nimi komfortnejšie. Priniesť nové myšlienky a nápady do problémovej oblasti digitálnych máp považujeme za veľkú výzvu, rovnako ako vylepšenie existujúcich riešení. Členovia nášho tímu disponujú všetkými znalosťami potrebnými na realizáciu všetkých fáz projektu podľa požiadaviek zadávateľa a dokážu poskytnúť vlastné zaujímavé nápady. Chceme sa zamerať hlavne na nasledujúce nedostatky existujúcich riešení:

* Nedostatok identifikovaných objektov s GPS súradnicami. Chýbajúce metadáta k nim. Potreba získania dát z viacerých zdrojov.
* Za geo-objekt sa považuje jeden bod charakterizovaný dvoma súradnicami. Existuje však viac typov a možností značenia geo-objektov(napr. polygón, trasa).
* Vyhľadávanie objektov a informácií na mape je slabo personalizované, ak vôbec. Potreba väčšieho prispôsobenia sa užívateľovi.
* Geo-objekty často nie sú zatriedené do kategórií. Ak aj sú, nie sú zaznamenané vzťahy medzi nimi.
* Chýbajúca integrácia s hromadne obľúbenými službami ako sociálne siete typu Facebook, Twitter. Možnosť využitia informácií o užívateľoch, ich pozície v sociálnej sieti a popularizácie medzi ďalšími užívateľmi.

# Špecifikácia

## Požiadavky na funkcionalitu

Na základe stanovených cieľov sme určili nasledovné procesy v systéme:

### Prehľad procesov

* BP01 Manažment používateľských účtov
* BP02 Navigácia po mape
* BP03 Manažment geografických objektov po mape
* BP04 Profily ovplyvňujúce správanie systému
* BP05 Vyhľadávanie geografických objektov
* BP06 Navigácia po konceptuálnej mape
* BP07 Manažment geografických objektov cez konceptuálnu mapu
* BP08 Manažment trás
* BP09 Manažment fotogalérií

### BP01 Manažment používateľských účtov

Ako už vyplýva z názvu, proces zahŕňa vedenie účtov používateľov systému. Rozhodli sme sa integrovať náš projekt so sociálnou sieťou Facebook, preto sa budeme snažiť využiť jej možnosti. K aplikácii sa bude môcť pristúpiť dvomi spôsobmi, od ktorých sa odvíja manažment používateľov. Prvým spôsobom je pristúpiť k aplikácii cez Facebook, tým pádom musí byť používateľ prihlásený. Pre naše potreby budeme pracovať s používateľovým uid, ktoré získame cez Facebook API.

Druhým spôsobom je priame zadanie adresy do prehliadača. Tu máme možnosť zistiť stav prihlásenia používateľa na Facebook, čo nám umožní automatické prihlásenie. Ak nie je žiadny používateľ aktuálne prihlásený na Facebook, môže sa na neho prihlásiť priamo cez našu aplikáciu. Ak nemá konto alebo záujem, bude sa môcť registrovať alebo prihlásiť cez účet nášho systému. Pri využití Facebook-u teda odpadá manažment používateľov ako taký, ten je nutný implementovať pre ostatných používateľov.

Takisto budú dva spôsoby pri práci s objektami. Facebook používateľ si bude môcť zvoliť svojich priateľov alebo skupiny, ktorým priradí práva na prácu s objektom, ktorého je autorom. Ostatný používatelia budú mať veľmi obmedzené možnosti - práva na operácie s objektom bude mať len samotný autor.



Obrázok 1 - Diagram prípadov použitia manažmentu používateľských účtov.

Nasledujúce sekvenčné diagramy podrobnejšie opisujú priebeh jednotlivých prípadov použitia.



Obrázok 2: Sekvenčný diagram založenia používateľa.



Obrázok 3: Sekvenčný diagram prihlásenia používateľa.



Obrázok 4: Sekvenčný diagram prihlásenia používateľa pomocou Facebook účtu.



Obrázok 5: Sekvenčný diagram odhlásenia používateľa.

Zmena údajov a zrušenie účtu používateľa prihláseného cez Facebook účet je možná iba cez aplikáciu Facebook.



Obrázok 6: Sekvenčný diagram zmeny účtu používateľa.



Obrázok 7: Sekvenčný diagram zrušenie účtu používateľa.

### BP02 Navigácia po mape

Navigáciu po mape poskytuje užívateľské rozhranie zložené z mapy a ďalších ovladacích nástrojov. Je zabezpečené intuitívne ovládanie mapy pomocou myši a kláves. Po výbere objektu, ktorý má byť zobrazený sa mapa automaticky presunie a vycentruje na daný objekt.



Obrázok 8 - Diagram prípadov použitia pre navigáciu po mape.

Na nasledujúcich sekvenčných diagramoch je podrobnejšie znázornený proces manuálneho posúvania sa po mape, približovania objektu a nakoniec prezeranie objektu.

Obrázok 9 – Sekvenčný diagram posúvania sa po mape.



Obrázok 10 – Sekvenčný diagram priblíženia na mape.



Obrázok 11 – Sekvenčný diagram prezerania objektu.



### BP03 Manažment geografických objektov po mape

V rámci manažmentu geografických objektov po mape, je zabezpečené pridanie nového objektu. Užívateľ zadá základné dáta o objekte alebo ich vyznačí priamo na mape, doplní prípadne ďalšie dáta a objekt uloží. Následne má možnosť údaje o objekte editovať. Ku každému objektu môžu užívatelia zadávať poznámky v podobe komentárov, prípadne ratingu objektu.



Obrázok 12 – Diagram prípadov použitia pre manažment geografických objektov po mape.

Sekvenčné diagrami znázorňujú podrobnejšie prípady použitia *Pridanie nového objektu*, *Editácia atribútov objektu* a *Poznámky používateľov*.



Obrázok 13 – Sekvenčný diagram pridania objektu.



Obrázok 14 – Sekvenčný diagram editácie objektu.



Obrázok 15 – Sekvenčný diagram pridania poznámky k objektu.

### BP04 Profily ovplyvňujúce správanie systému

Prihlásený používateľ môže jednotlivé časti rozhrania aplikácie personalizovať. Teda prispôsobiť si ich vzhľad resp. spôsob používania. Personalizácia mapy zahŕňa prispôsobenie spôsobu jej ovládania a prispôsobenie prezentácie (vzhľadu) geo-objektov (trás a bodových geo-objektov). Personalizácia vyhľadávania umožňuje prispôsobiť prehľadávané koncepty a koncepty ktorých sa týka automatické dopĺňanie rozpísaných slov. Podobným spôsobom je možné prispôsobiť aj prácu s objektmi. Či už bodové alebo trasové geo-objekty je možné odporučiť inému používateľovi.



Obrázok 16 – Diagram prípadov použitia pre profily ovplyvňujúce správanie systému.

### BP05 Vyhľadávanie geografických objektov

Používateľ využíva pre vyhľadanie objektu alebo konceptu to isté vyhľadávacie pole. Prípad použitia *Vyhľadávanie* následne zabezpečuje rozlíšenie medzi objektami a konceptami, z ktorých užívateľ vyberie hľadaný geo-objekt resp. koncept.



Obrázok 17 – Diagram prípadov použitia pre vyhľadávanie geografických objektov.



Obrázok 18 – Sekvenčný diagram vyhľadávania.

### BP06 Navigácia po konceptuálnej mape

Používateľ sa naviguje po konceptuálnej mape prehliadaním grafu a následným výberom (konkretizáciou alebo zovšeobecnením) konceptu. Týmto výberom vzniká požiadavka na server, ktorý ju spracuje a poskytne príslušnú časť konceptuálnej mapy. Pri výbere konceptu je súčasné notifikovaná mapa, ktorá zobrazí objekty v kontexte s daným konceptom.



Obrázok 19 – Diagram prípadov použitia pre navigáciu po konceptuálnej mape.



Obrázok 20 – Sekvenčný diagram navigácie medzi konceptami.

### BP07 Manažment geografických objektov cez konceptuálnu mapu

Konceptuálna mapa bude zobrazená grafom, kde jednotlivé body budú predstavovať body záujmu - koncepty. Výberom daného bodu na grafe je možné editovať jeho vlastnosti rovnako ako v BP03.



Obrázok 21 – Diagram prípadov použitia pre manažment geografických objektov cez konceptuálnu mapu.

### BP08 Manažment trás

Manažment trás umožňuje vytvorenie novej trasy, vyhľadanie existujúcej trasy a ohodnotenie trasy. Pri vytváraní trasy je aktuálne zobrazovaná do mapy. Podobne sú znázornené alternatívy a ich hodnotenie. Vyhľadávanie trasy pozostáva z vyhľadania dvoch (prípadne viacerých) geografických objektov na mape.



Obrázok 22 – Diagram prípadov použitia pre manažment trás.

### BP09 Manažment fotogalérií

Fotogalérie musia byť naviazané na objekt na mape, preto proces vytvorenia fotogalérie využíva proces vyhľadania objektu. Fotogaléria má tieto kľúčové špecifické parametre:

* Miesto, z ktorého sa fotilo
* Miesto (objekt), ktoré je na fotografii

Fotogalérie sa môžu voľne spájať, pretože môžeme kompozíciou vytvoriť novú virtuálnu fotogalériu. Napríklad, ak je objekt A viditeľný z dvoch rôznych miest, tak nová virtuálna galéria (nájdená požiadavkou „všetky fotografie objektu A“) bude obsahovať obidve tieto fotky. Používateľ preto nemusí prehľadávať dve nezávislé fotogalérie.



Obrázok 23 – Diagram prípadov použitia pre manažment fotogalérií.

# Návrh

## Architektúra systému a prehľad použitých technológií

Pre navrhovaný systém digitálnych máp je použitá klient-server architektúra, pričom klient beží v prostredí webového prehliadača. Klient aj server sa skladajú z viacerých komponentov, ich zloženie a komunikácia medzi nimi spolu s použitými technológiami je znázornená na obrázku 15. Internetové aplikácie majú oproti desktopovým aplikáciám mnoho výhod. Sú dostupnejšie, nie je potrebná inštalácia a ich vývoj je jednoduchší. Hlavný nedostatok internetových aplikácií, ktorým je chudobnejšie používateľské rozhranie v prípade aplikácie digitálnych máp rieši použitie technológie AJAX.

V ďalších podkapitolách sú bližšie špecifikované jednotlivé komponenty systému, pričom je zároveň vysvetlená komunikácia medzi nimi, pri ktorej sú použité viaceré technológie a programovacie jazyky.

### Server

Strana servera sa skladá z dvoch častí:

* Úložisko údajov
* Webová služba

Obojsmerná komunikácia medzi uvedenými časťami je zabezpečená pomocou technológie ADO .NET, ktorá je samozrejme súčasťou frameworku .NET. Platforma .NET poskytuje veľké množstvo tzv. data providers, pričom každý je špecializovaný pre komunikáciu so špecifickým databázovým systémom (SQL Server, Oracle, MySQL, atď). Technológia ADO .NET môže byť použitá dvomi základnými spôsobmi. V prvom prípade (ConnectedLayer) je pripojenie k vytvárené a ukončované explicitne. V druhom prípade (DisconnectedLayer) je po získaní dát z RDBS spojenie implicitne ukončené a pre manipuláciu s dátami je použitá ich kópia na strane klienta. Táto možnosť podporuje škálovateľnosť riešenia, keďže používané zdroje sú rýchlo uvoľnené.

### Úložisko údajov

Ako úložisko údajov je použitý relačný databázový systém PostgreSQL. Jeho úlohou je uchovávanie všetkých potrebných údajov pre beh aplikácie. Prvá skupina údajov sa týka správy používateľských kont. Druhá, omnoho väčšia časť je tvorená na jednej strane dátami, ktoré pridali v priebehu používania aplikácie jej používatelia, na strane druhej dátami, ktoré sú dodávané spolu s aplikáciou (trasy, objekty). Vo finálnom produkte bude v databáze uchovaných zhruba 10.000 objektov na Slovensku, 200.000 českých a okolo 140.000 svetových.

### Webová služba

Webová služba tvorí biznis vrstvu aplikácie digitálnych máp. Odpovedá na volania klientskej časti aplikácie (prezentačná vrstva) a riadi prístup k úložisku dát. Samotná webová služba je implementovaná pomocou frameworku .NET. Výhodou je možnosť použiť pre vývoj ľubovoľný frameworkom podporovaný jazyk (C#, C++, VB .NET, Delphi .NET, atď.).

### Klient

Klient predstavuje prezentačnú vrstvu aplikácie. Keďže prvok mapy tvorí významnú časť používateľského rozhrania, je architektúra prezentačnej vrstvy v diagrame rozdelená na časť tvorenú samotnou mapou a časť tvorenú zostávajúcimi časťami používateľského rozhrania. Komunikácia jednotlivých prvkov gui v rámci klienta prebieha pomocou jazyka JavaScript. Prezentačná vrstva je implementovaná pomocou technológie ASP .NET a CSS.

### Server Google Maps

Poskytovateľom samotnej mapy a jej základnej funkcionality (približovanie, pohyb v mape, kreslenie do mapy) je server Google Maps. Využívanie možností, ktoré poskytuje Google pre prácu s mapou prebieha pomocou volaní v jazyku JavaScript prostredníctvom techniky AJAX. Mapu je možné popri s ňou dodávaných dátach obohatiť aj o nové dáta, v tomto prípade dodané našou aplikáciou (trasy a bodové objekty).

### Klient-server komunikácia

Navrhnutá architektúra poskytuje možnosť flexibilnej komunikácie medzi stranou klienta a stranou serveru. Udalosti spojené s prvkami používateľského rozhrania je podľa potreby možné obslúžiť buď na strane klienta, resp. v prípade potreby dotazov na úložisko dát aj na strane serveru. Komunikácia prebieha asynchrónne pomocou techniky AJAX, takže používateľ ju neregistruje.

Architektúra sa skladá z centrálneho úložiska, ktoré tvorí databázový server, webovej aplikácie na servri a klientskej časti aplikácie.

Klientskú časť tvorí niekoľko základných komponentov a to používateľské rozranie, ktoré obsahuje všetky komponenty potrebné na interakciu s používateľom a rozhranie mapy. Rozhranie mapy je nadstavbou nad základnou funkcionalitou Google mapy a poskytuje hlavne možnosť komunikácie so servrom. Okrem toho sa stará o manažment geografických objektov v klientskej časti aplikácie a posielanie príbuyných údajov servru a všetkými ostatnými funkciami súvisiacimi s objektom mapy. Oba tieto komponenty komunikujú so serverom prostredníctvom technológie AJAX a medzi sebou prostredníctvom rozhraní v implementovaných v JavaScripte.

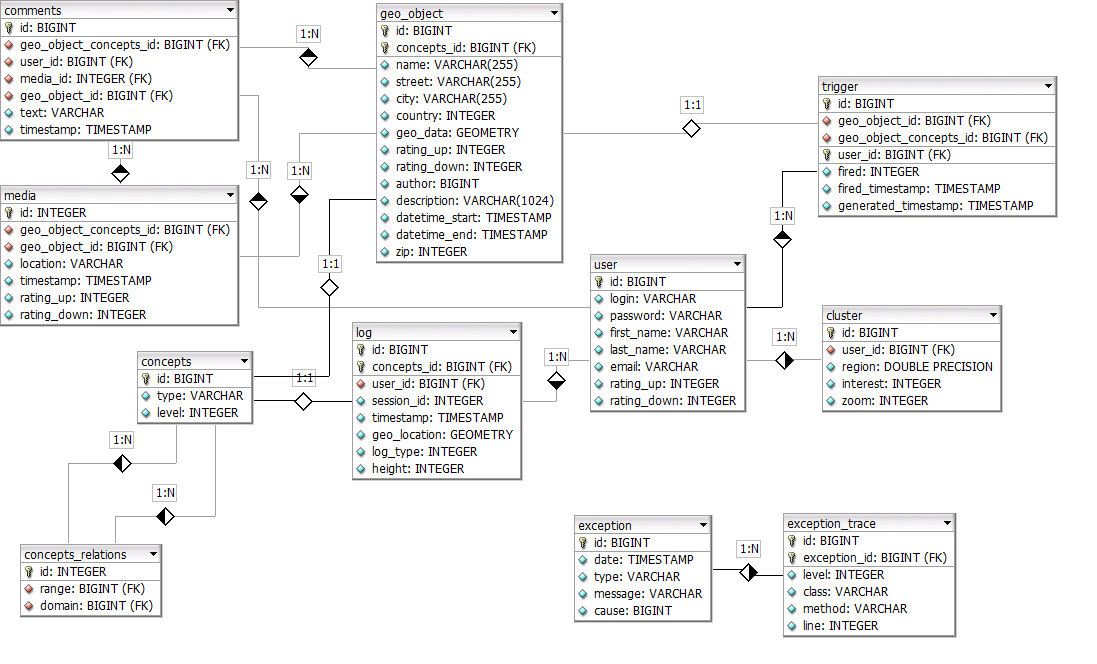


Obrázok 24 – Diagram komponentov aplikácie digitálnych máp.

## Dátový model

Vo veľkej miere budeme využívať databázu umiestnenú na strane servera. Je potrebná na zabezpečenie manažmentu používateľov, zaznamenanie ich činností ako pohyb po mape, obľúbené lokácie, preferované typy objektov. Tiež sa v nej nachádzajú geografické údaje o  objektoch, ktoré sú zatriedené do kategórií. Databáza nám taktiež zabezpečuje zaznamenávanie vzťahov medzi jednotlivými kategóriami.

V projekte použijeme databázu PostgreSQL, ktorá nám umožňuje pohodlnú prácu s geo-dátami, a disponuje veľkým množstvom zabudovaných funkcií, ktoré výrazne uľahčujú prácu na dátovej vrstve. Návrh tabuliek s príslušnými atribútmi je znázornený v dátovom modeli na obrázku 16.



Obrázok 25 – Dátový model údajov.

# Prototyp

Kapitola obsahuje opis prototypu vytvoreného v zimnom semestri. V prvej časti kapitoly sú opísané ciele prototypovania a množina špecifikovaných služieb systému zahrnutých do prototypu.

Ďalšia časť opisuje jednotlivé vrstvy systém z pohľadu použitých technológií a ich implementácie.

## Ciele prototypovania

Medzi najdôležitejšie ciele procesu prototypovania vytváraného systému patrí určenie náročnosti procesu implementácie prostredníctvom špecifikovaných technológií a možnosti implementovať technologicky najnáročnejšiu funkcionalitu. Na uvedený cieľ nadväzuje overenie schopnosti dodržať vytýčený projektový plán a jeho následná úprava.

Prototyp musí overiť zhodu medzi vytvoreným návrhom a požiadavkami zadávateľa prostredníctvom použitia vybranej funkcionality v praxi. Ďalším cieľom procesu prototypovania je upresnenie špecifikácie vytváranej funkcionality a teda zlepšenie celkového návrhu systému. Prostredníctvom prototypu budú lepšie pochopené zadávateľove požiadavky resp. identifikované chýbajúce služby systému.

## Množina prototypovanej funkcionality

Podkapitola opisuje množinu špecifikovaných služieb (prípadov použitia) zahrnutých do prototypu a ich opis z hľadiska skutočného fungovania. Podmnožina špecifikovanej funkcionality zahrnutá do prototypu bola vyberaná s cieľom overenia správnosti návrhu z hľadiska používateľských požiadaviek a overenia technických možností jej realizácie.

### Prihlásenie používateľa

Služba systému *Prihlásenie používateľa* zabezpečuje prihlásenie jednak používateľa registrovaného prostredníctvom aplikácie digitálnych máp a jednak používateľa registrovaného prostredníctvom sociálnej siete Facebook. Pri vyvolaní požiadavky na prihlásenie prostredníctvom Facebook je vyvolané samostatné externé prihlasovacie okno s využitím ktorého sa používateľ prihlási. Služby pre prihlásenie používateľa sú združené v triede *UserManagement*.

### Približovanie mapy

Služba *Približovanie mapy* zabezpečuje približovanie mapy prostredníctvom posuvného ovládača nachádzajúceho sa v priestore geografickej mapy a prostredníctvom dvojitého kliknutia do priestoru geografickej mapy.

### Odďaľovanie mapy

Služba *Odďaľovanie mapy* zabezpečuje odďaľovanie geografickej mapy prostredníctvom posuvného ovládača nachádzajúceho sa v priestore geografickej mapy.

### Posúvanie sa po mape

Služba *Posúvanie mapy* zabezpečuje posúvanie geografickej mapy prostredníctvom štvorsmerového ovládača a prostredníctvom potiahnutia geografickej mapy.

### Pridanie nového objektu

Služba *Pridanie nového objektu* zabezpečuje pridanie geografického objektu a uloženie jeho údajov do databázy. Súradnice geografického objektu sú určené prostredníctvom jednoduchého kliknutia na geografickú mapu po prepnutí do módu pridávania nových objektov.

### Vyhľadávanie objektov a konceptov

Jednoduchý prístup k objektom a konceptom v databáze umožňuje služba *Vyhľadávanie objektov a konceptov.* Vyhľadávanie podľa názvu objektu je dostupné prostredníctvom vyhľadávacieho formuláru nachádzajúceho sa nad geografickou mapou. Vyhľadávací formulár je rozšírený o ponuku automatického dopĺňania vloženého textu na základe zadaného textu na vyhľadávanie. Dopyt získaný z vyhľadávania, je ďalej analyzovaný na strane servera. Cieľom je zistenie, o aký druh geografického objektu sa jedná, prípadne, či to nie je koncept. Na základe analyzovaného dopytu je notifikovaná mapa, vykoná príslušný pohyb mapou, zobrazenie objektov na nej a zmení sa graf. Žiadané objekty sú zobrazované len v aktuálne zobrazenej mape, preto vždy pred zobrazením objektov sú získané krajné body zobrazenej oblasti, tieto body vytvárajú ohraničenie pre práve zobrazované objekty.

V prípade, že zobrazovaných objektov je priveľa, pred zobrazením objektov je spúšťaný klastrovací algoritmus, ktorý zabraňuje priveľkému zaťaženiu pri zobrazení väčšieho počtu objektov. Klastrovací algoritmus analyzuje polohu objektov v zobrazovanej oblasti a vytvára klastre, v ktorých sú objekty sústredené. Pri každom klastri je orientačne uvedené číslo, ktoré vyjadruje počet objektov zaradených do daného klastra

### Navigácia medzi konceptmi

Služba *Navigácia medzi konceptmi* je zabezpečená prostredníctvom konceptuálneho grafu. Konceptuálny graf zobrazuje časť konceptuálnej mapy pričom centrom je koncept na ktorý je používateľ práve zameraný. Okolo centra sú zobrazení potomkovia centra v konceptuálnej mape a jeho rodič. Rodič, potomkovia a centrum sú farebne odlíšené.

V grafe konceptov je možné prepínať medzi dvoma módmi. V jednom sú zobrazovaní potomkovia práve zobrazovaného centrálneho uzlu, kým v druhom móde sú zobrazované uzly na rovnakej úrovni (súrodenci).

## Prezentačná vrstva

Prezentačná vrstva web aplikácií vytvorených prostredníctvom technológie ASP .NET je tvorená dynamicky generovanými web stránkami. Vzhľad stránky je definovaný na jednej strane prostredníctvom klasických HTML tagov, na druhej strane prostredníctvom tagov rozoznávaných a interpretovaných webovým serverom (Web server controls). Serverom rozoznávané prvky stránky majú pre jednotlivé udalosti priradené metódy triedy predstavujúcej samotnú web stránku na strane serveru (Code Behind) vyvolávané pri vzniku danej udalosti.

Štandardná množina prvkov pre tvorbu používateľského rozhrania bola rozšírená o prvky poskytované knižnicou Ajax Control Toolkit. Ajax Control Toolkit je open-source projekt postavený nad ASP .NET poskytujúci širokú škálu prvkov (tzv. controls) používateľského rozhrania pre ASP .NET webové aplikácie. Ďalšou použitou knižnicou je Google Maps .NET Control zabezpečujúca vykreslenie geografickej mapy.

Diagram tried na obrázku 26 zobrazuje najdôležitejšie triedy použité pri vytváraní používateľského rozhrania tvoreného dynamicky generovanou web stránkou.

Trieda *ScriptManager* zabezpečuje AJAX funkcionalitu aplikácie. Umožňuje stiahnutie skriptov knižnice Microsoft AJAX Library prehliadačom a koordinuje vykonávanie zmien častí stránky s využitím triedy *UpdatePanel*. Pomocou metódy *RegisterClientScriptBlock()* triedy *ScriptManager* je prehliadaču v prípade potreby (napr. pridanie objektu do geografickej mapy) posielaný blok skriptu určený na okamžité vykonanie.

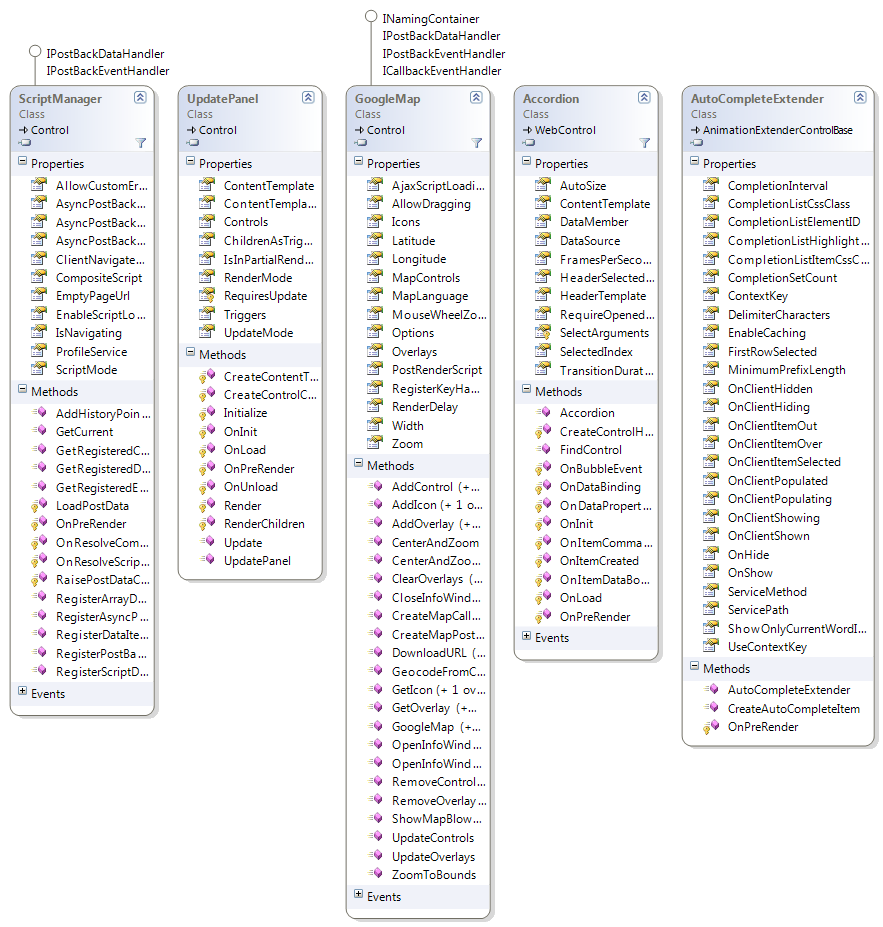
Trieda *UpdatePanel* frameworku .NET je používaná na vykonanie prekreslenia častí stránky bez spustenia synchrónneho postback-u. Obaľuje server tag triedy *Accordion* a teda umožňuje asynchrónnu aktualizáciu jeho obsahu.

Pre prácu s geografickou mapou je použitá knižnica Google Maps .NET Control. Samotné zobrazovanie geografickej mapy zabezpečuje trieda GoogleMap. Jej metódy generujú bloky skriptu, ktorý je následne prostredníctvom metódy *RegisterClientScriptBlock()* triedy *ScriptManager* odoslaný prehliadaču na vykonanie. Metódy triedy GoogleMap teda obaľujú priame volania nad objektom Google mapy existujúcom na klientovi.

Trieda *Accordion* zabezpečuje generovanie kódu pre vykreslenie vysúvateľných panelov na strane klienta. Panely obsahujú prvky používateľského rozhrania pre navigáciu po konceptuálnej mape, manažment používateľských účtov resp. pre manažment geo-objektov. Server tag triedy *Accordion* je obalený prostredníctvom tagu triedy *UpdatePanel* a teda jeho obsah je obnovovaný asynchrónne prostredníctvom technológie AJAX.

Proces vyhľadávania vyžaduje automatické dopĺňanie zadávaného výrazu. Potrebné dáta sú získavané s využitím webovej služby *AutocompleteService*. Generovanie skriptu do klientskej stránky pre prístup k API pre zabezpečenie komunikácie medzi prehliadačom a službou je vykonané triedou *AutoCompleteExtender* pri jej vytváraní. Asynchrónne volania webovej služby serveru sú vykonávané s využitím objektu XMLHTTP, pričom časť vrstvy zabezpečujúcej komunikáciu lokalizovanej na klientovi predstavujú stiahnuté skripty.

Konceptuálna mapa je vytvorená prostredníctvom objektu platformy Flash. Prístup k objektu a zachytávanie udalostí je obalené prostredníctvom knižnice GraphGear. Knižnica ošetrovanie vznikajúcich udalostí presúva do funkcie *nodeNotify(id)* s potrebnou hlavičkou(stále na strane klienta), ktorej parametrom je identifikátor vybraného uzlu grafu. Trieda *ScriptManager* zabezpečuje v prípade potreby vykonanie skriptu volajúceho funkcie knižnice GraphGear pre aktualizáciu (prekreslenie) objektu grafu.



Obrázok 26 Diagram najdôležitejších tried zabezpečujúcich funkcionalitu prezentačnej vrstvy

## Logická vrstva

Prepojenie medzi prezentačnou a dátovou vrstvou zabezpečuje logická vrstva. Logická vrstva obsahuje kód vykonávaný na serveri, v ktorom sa vhodne spracovávajú dáta poskytnuté prezentačnou vrstvou resp. dátovou vrstvou pri vyhľadávaní, zobrazovaní mapy a objektov na nej a pri zmene grafu. Reprezentácia logickej vrstvy je znázornená na obrázku 26 a podrobne opísaná nižšie.

Webová služba *AutocompleteService* a jej funkcie zabezpečujú napĺňanie poľa priebežných výsledkov vyhľadávania. Text vo vyhľadávacom poli vstupuje ako parameter do funkcie *getSuggestion.* Tá vracia v poli zoznam výsledkov nájdených v databáze, pri ktorých je vyznačené o aký druh geografického objektu sa jedná.

Trieda *SearchAnalyzer* vykonáva vo svojich funkciách analýzu vyhľadávania. Funkcia *AnalyzeString* analyzuje reťazec, ktorý bol odoslaný vo vyhľadávacom formulári. Funkcie identifikuje jednotlivé výrazy v reťazci a rozlíši, či sa jedná o objekt, koncept, ulicu, miesto, prípadne ich označí ako nerozoznané. Následne sú výsledky analýzy odovzdávané funkcii *Search*, ktorá pre rôzne kombinácie rozoznaných výrazov, ktoré dostane ako vstup, vykoná príslušnú akciu. Podľa počtu objektov, ktoré majú byť zobrazené sú notifikované funkcie tried *MapPainter* resp. *MarkerClusterer*.

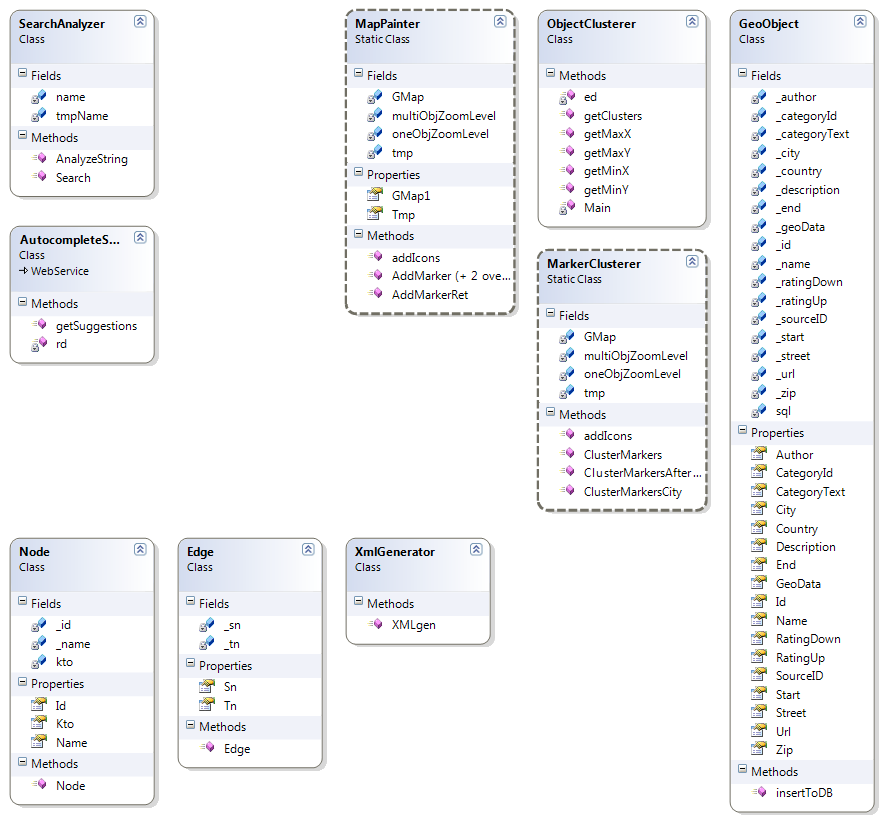
Funkcie triedy *MarkerClusterer* sú volané v prípade, že počet objektov ktoré majú byť zobrazené je väčší ako 70. Funkcia *ClusterMarkers* zabezpečuje rozdelenie mapy na klastre, podľa rozloženia objektov a vykreslenie graficky zvýraznených klastrov. Na výpočet klastrov je zavolaná funkcia *getClusters* triedy *ObjectClusterer*, ktorá na vstupe dosáva súradnice objektov a vracia pole klastrov s príslušným počtom objektov v nich.

V prípade, že počet objektov, ktoré majú byť zobrazené je menší ako 70, sú volané metódy triedy *MapPainter*. Konkrétne funkcia *AddMarker* zabezpečuje vykreslenie objektu a vyznačenie polohy pomocou markera. Každému zobrazenému objektu sú vo funkcii priradené dostupné informácie o ňom z našej databázy.

Každý geografický objekt je v prípade, že sa ním pracuje reprezentovaný ako inštancia triedy *GeoObject,* ktorá nám zabezpečuje prístup k jeho príslušným poliam.

Zobrazenie a prekreslovanie grafu je zabazpečené triedou *XmlGenerator*. Jej funkcia *XMLgen* je v potrebnom prípade notifikovaná ostatnými komponentmi a na základe momentálneho stavu ostatných komponentov zabezpečí prekreslenie grafu. Graf je vždy vytvorený na základe vygenerovaného *XML* súboru funkciou *XMLgen*. Pri generovaní súboru sú využívané triedy *Node* a *Edge*, ktoré reprezentujú vrchol a hranu. Ich metódy vracajú potrebné hodnoty, ktoré sú následne zapísane do *XML* súboru slúžiaceho na vytvorenie grafu.

Funkcie triedy *Hints* sú volané v prípade, že majú byť zobrazené odporúčania objektov alebo konceptov, história alebo obľúbené. Odporúčania sa vyvolajú metódou rôznymi udalosťami – napr. vyhľadávaním, pohybom po mape, prezeraním objetku, atď. Na ich získanie a zobrazenie slúži metóda *addHints.* História sa zobrazí po kliknutí na záložku história, na ich získanie a zobrazenie slúži metóda *addHistory*. Obľúbené sa zobrazia po kliknutí na záložku história, na ich získanie a zobrazenie slúži metóda *addFavorites*.



Obrázok 27 – Logická vrstva prototypu.

## Dátová vrstva

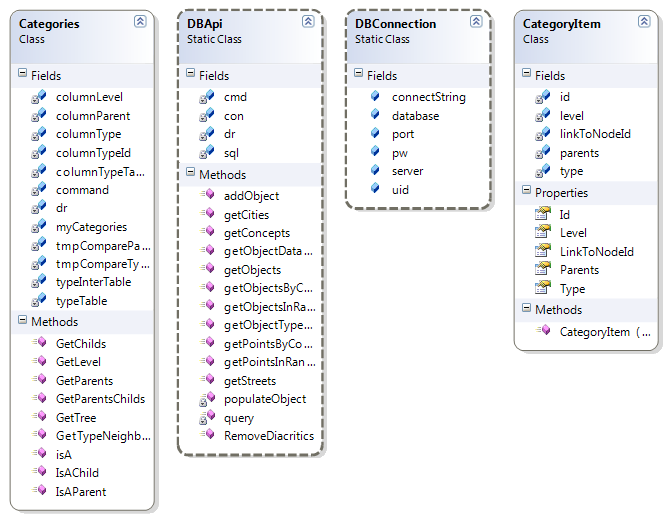
Dátová vrstva systému umožňuje prístup k databázovému systému PostgreSQL na vyššej úrovni. Databázové rozhranie je rozčlenené na triedu *Categories* atriedu *DBApi*.

Trieda *Categories* zabezpečuje prístup ku konceptuálnej mape pomocou metód zabezpečujúcich navigáciu v strome konceptov pomocou získania rodičov, potomkov resp. súrodencov daného konceptu.

Trieda *DBApi* zabezpečuje prístup ku geo-objektom uložených v databázovom systéme pomocou metód zabezpečujúcich naplnenie údajov geo-objektu, získanie geo-objektu podľa kľúčového slova, konceptu alebo podľa oblasti.

Obrázok 28 znázorňuje časť diagramu tried vytvoreného prototypu týkajúcu sa databázovej vrstvy.

Obrázok 28 znázorňuje časť diagramu tried vytvoreného prototypu týkajúcu sa databázovej vrstvy.



Obrázok 28 – Dátová vrstva prototypu.

## Dosiahnuté výsledky

Z Cieľov prototypovania uvedených v kapitole 5.1, boli aspoň čiastočne implementované všetky. V rámci prototypu sú plne funkčné približovanie/oddaľovanie mapy, posúvanie sa po mape, vyhľadávanie objektov, vyhľadávanie konceptov a navigácia medzi konceptmi. Čiastočne funkčné je prihlasovanie používateľa a pridávanie objektov, oba ciele prototypovania sú však ďalej realizovateľné.

Keďže všetky stanovené ciele boli technicky realizovateľné, a väčšina z nich plne realizovaná, v rámci implementácie prototypu bola potvrdená správnosť návrhu, ktorý bol takto overený.

## Overenie prototypu

V rámci overenia prototypu boli vykonané dva odlišné testy. Na overenie výkonu prototypu, technických možností a záťaže klientskej a serverovej časti aplikácie bol vykonaný záťažový test. Na overenie poskytnutej funkcionality boli vykonané akceptačné testy.

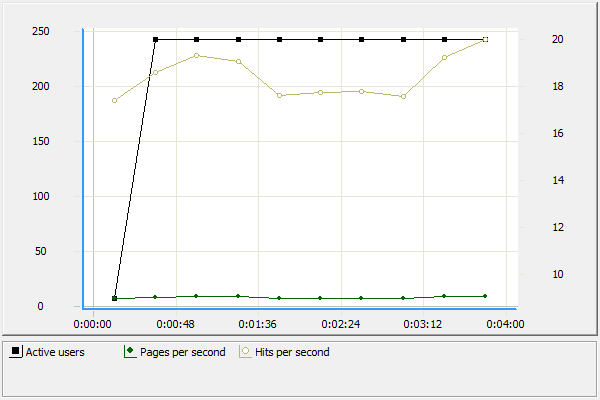
### Záťažový test

Cieľom záťažového testu bolo overiť realizovateľnosť priamočiareho notifikovania jednotlivých komponentov podľa návrhu. Záťažový test bol vykonaný pomocou testovacieho nástroja WAPT, ktorého trial verzia je voľne dostupná na <http://www.loadtestingtool.com/>. V trial verzii, je možné simulovať záťaž s dvadsiatimi virtuálnymi používateľmi. Pred začatím testu bola nahratá komunikácia, ktorú potom virtuálny užívatelia súčasne viackrát po sebe vykonávali. Nahratá komunikácia konkrétne v tomto teste obsahovala nasledovné kroky:

1. opakované vyhľadanie konceptu, zobrazenie objektov, pohyb po mape, prezeranie informácií o objektoch
2. opakované kliknutie na iný koncept cez graf
3. zobrazenie objektov, pohyb po mape, prezeranie informácií o objektoch
4. opakované vyhľadanie konkrétneho objektu podľa názvu.
5. opakované vyhľadanie ulice a mesta, pohyb po mape, prezeranie informácií o objektoch

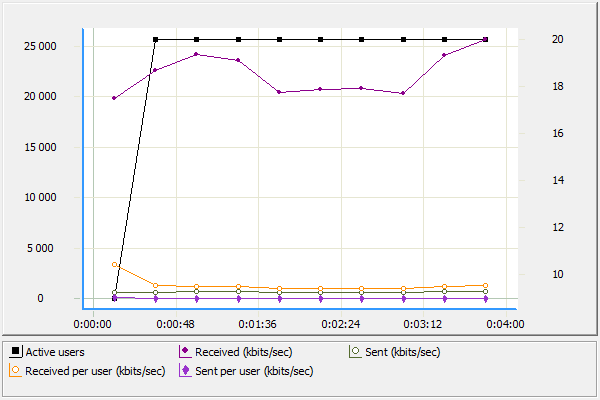
Výstup testu, ktorý poskytol nástroj sú grafy a tabuľky, zachytávajúce správanie sa aplikácie v závislosti od priebehu testu, pričom boli sledované tri javy – počet interakcií, objem prenesených dát a čas odozvy.

#### Počet interakcií



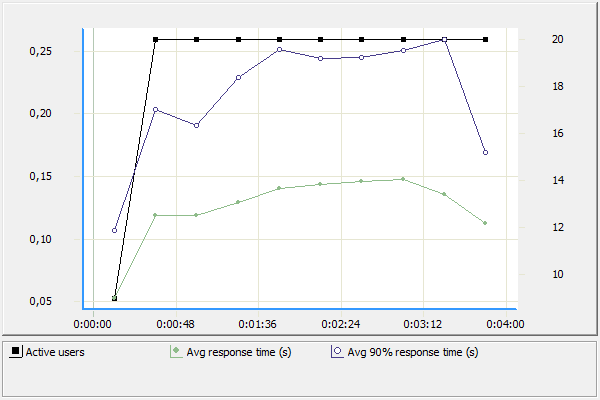
Obrázok 29 – Počet interakcií.

#### Objem prenesených dát



Obrázok 30 – Objem prenesených dát.

#### Čas odozvy



Obrázok 31 – Čas odozvy.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Čas**  **[sek]** | **00:00- 00:24** | **00:24- 00:48** | **00:48- 01:12** | **01:12- 01:36** | **01:36- 02:00** | **02:00- 02:24** | **02:24- 02:48** | **02:48- 03:12** | **03:12- 03:36** | **03:36- 04:00** |
| **Min** | 0,01 | 0,02 | 0,02 | 0,03 | 0,01 | 0 | 0,01 | 0,02 | 0,02 | 0,03 |
| **Avg** | 0,05 | 0,12 | 0,12 | 0,13 | 0,14 | 0,14 | 0,15 | 0,15 | 0,14 | 0,11 |
| **Avg90** | 0,11 | 0,2 | 0,19 | 0,23 | 0,25 | 0,24 | 0,24 | 0,25 | 0,26 | 0,17 |
| **Max** | 0,31 | 0,42 | 0,41 | 0,78 | 0,55 | 0,5 | 0,57 | 0,5 | 1,0 | 0,38 |

Tabuľka 1 - Odozva v sekundách

### *Vyhodnotenie záťažového testu*

Na základe výsledkov záťažového testu je možné skonštatovať, že navrhnutý prístup priamočiareho notifikovania komponentov, je technicky realizovateľný. Najdôležitejšie kritérium – čas odozvy – zostáva nízke aj pri maximálnom vyťažení. Možným problémom však môže byť veľký počet prenesených dát pri maximálnej premávke v rámci testu. Z tohto dôvodu je potrebné záťažový test po nasadení prototypu na server zopakovať, s cieľom overenia schopnosti servera zvládať prenos dostatočne veľkého objemu dát pri maximálnej prevádzke.

# Revízia návrhu

## Technická dokumentácia klienta

V tejto časti sú uvedené princípy a konvencie používané pri vývoji na strane klienta. Hlavné použité technológie na tomto mieste boli: Javascript, CSS a JQuery ktorých dokumentácia je dostupná na webe.

Klientská časť aplikácie zahŕňa najmä klienské skripty. Jednotlivým elementom sú priradené DOM identifikátory prostredníctvom ktorých sa elementy referencujú. Základné delenie prvkov na stránke je nasledovné:

* Bočný panel
* Panel vyhľadávania
* Panel prihlásenia a registrácie
* Panel pre navigovanie
* Lišta s fotkami

Každý z prvkov má definovaný vlastný súbor v ktorom sú formou JS triedy zapúzdrené súvisiace operácie. Špeciálnym prvkom je inicializačný, ktorý obsahuje postupnosť inicializácie ostatných prvkov.

Všetky triedy sú umiestnené v adresári /jscript/, každá v separátnom súbore. Jednotlivé súbory sú:

* Init.js – obsahuje inicializačné rutiny, definíciu globálnych premenných a volanie inicializácie ostatných prvkov
* Panel.js – súbor obsahujúci funkcionalitu vyžadovanú bočným panelom, tj. Jeho skrývanie a podobne.
* Photos.js - súbor obsahujúci funkcionalitu vyžadovanú panelom fotiek
* Searchbox.js – súbor obsahujúci funkcionalitu vyžadovanú panelom pre vyhľadávanie, tj. Presúvanie, animácie a ďalšiu funkcionalitu spojenú s jeho elementami (napr. textbox pre zadávanie výhľadávaného výrazu)
* Zoomcontrol.js – súbor obsahujúci funkcionalitu potrebnú pre tlačítka umožňujúce zoom a presúvanie mapy

Pri inicializácii stránky sa na jednotlivé elementy naviažu udalosti. Toto sa realizuje v metóde $(document).ready(), ktorej argument je funkcia. Na tomto mieste sa realizujú aj niektoré inicializačné činnosti na strane klienta. Prebehne inicializácia jednotlivých prvkov.

Každý prvok má k dispozícii metódu initialize, ktorá sa volá na začiatku pri vyvolaní udalosti $(document).ready(). Na tomto mieste sú inicializované potrebné viazania udalostí pre konkrétny element. Za touto časťou sú uvedené definície funkcií, ktoré obsluhujú niektoré jednoduché úlohy: skrývanie panelov, ich zobrazovanie, animáciu zmeny farby pozadia a podobne. Ide o konkrétne úlohy spojené so špecifickou činnosťou.

Po inicializácii sa pri nastaní určitých udalostí uskutočňujú asynchrónne volania. Na tento druh komunikácie je využitá primárne vstavaná podpora v Microsoft ASP.NET, ale v niektorých špecifických prípadoch je komunikácia explicitne uskutočnená prostredníctvom JQuery. V takýchto prípadoch (napr volanie suggestions služby) je rutina definujúca volanie definovaná priamo na príslušnom mieste kódu.

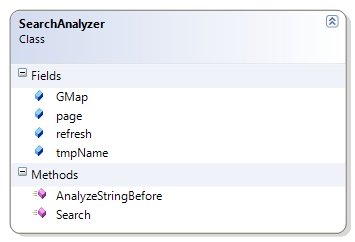
Vzhľadom na princíp fungovania ASP.NET AJAX, sú všetky elementy obsiahnuté v ControlPaneloch na servri pri asynchrónnom volaní aktualizované. Spôsobí to aj problem so zánikom previazanie elementov s udalosťami definovanými pri inicializácii. Toto previazanie je možné obnoviť volaním inicializačnej metódy príslušného prvku.

## Technická dokumentácia servera

Počas implementácie v druhom semestri bolo potrebné pristúpiť k revízii návrhu serverovej časti aplikácie. Vyžiadalo si to isté zmeny aj v častiach, ktoré už boli realizované v prototype. Keďže opis serverovej časti aplikácie v kapitole 4 vychádzal v niektorých prípadoch len z návrhu a bol často len názorný preto uvádzame v revízii podrobnejší popis jednotlivých súčastí a tried, ktorý zároveň zahŕňa vykonané zmeny.

### Triedy logickej vrstvy

### Trieda SearchAnalyzer

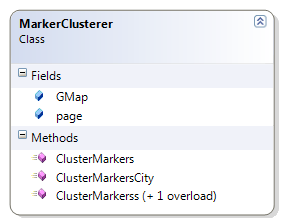


Trieda *SearchAnalyzer* zabezpečuje prostredníctvom svojich funkcií analýzu hľadaného výrazu.

Po aktivovaní vyhľadávania používateľom je notifikovaná funkcia *Search.* V jej tele je najskôr volaná funkcia *AnalyzeStringBefore,* ktorej vstupným parametrom je hľadaný reťazec. Pri analýze sa v jej tele využíva reverzné geokódovanie, ktoré poskytuje Google. My využívame konkrétne funkciu ddd, ktorú poskytuje trieda ddd. Pre každý hľadaný reťazec vráti pole geo-objektov ako výsledky vyhľadávania. Podľa výsledkov reverzného geokódovania sa následne hľadaný reťazec porovnáva s dátami v našej databáze. Vo funkcii sa na základe týchto porovnaní naplní výsledkové pole, podľa toho, či bol rozoznaný konkrétny objekt, koncept alebo oblasť. Toto výsledkové pole je ďalej vyhodnocované vo funkcii *Search.* Podľa jeho obsahu sa vykoná príslušná vetva v zdrojovom kóde. Rozlišujú sa nasledovné možnosti:

* Konkrétny objekt
* Oblasť, miesto
* Koncept
* Koncept v oblasti

### Trieda MarkerClusterer



Trieda *MarkerClusterer* zabezpečuje vo svojich funkciách posunutie mapy a vykreslenie príslušného počtu objektov/klastrov, podľa aktuálnej polohy, priblíženia a dopytu.

Funkcia *ClusterMarkers* je volaná v prípade ak je vyhľadaný a rozpoznaný nejaký koncept a je potrebné na aktuálne zobrazenej oblasti v mape vykresliť objekty daného konceptu pre danú oblasť. Vo funkcii sa najskôr vykoná dopyt na databázu. V databáze sa hľadajú všetky objekty daného konceptu nachádzajúce sa v aktuálnej oblasti. V prípade že objektov, ktoré majú byť vykreslené je viac ako 80 je volaná funkcia *getClusters* triedy *Program,* ktorá vypočíta súradnice klasterov.

Obdobne funguje funkcia *ClusterMarkersCity,* ktorá sa vykonáva v prípade, že okrem konceptu bola rozoznaná aj oblasť resp. konkrétne miesto.

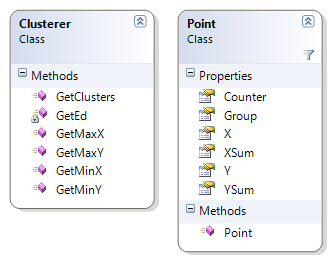
Tú istú činnosť vykonáva aj funkcia ClusterMarkerss a jej preťažená funkcia, sú však volané v prípade rozpoznávania konkrétnej vygenerovanej URL s parametrami v prípade jej zadania priamo do prehliadača.

### Trieda MapPainter

Trieda *MapPainter* a jej funkcie zabezpečujú vykreslenie konkrétnych objektov na mape. Vykreslenie spočíva v generovaní textového reťazca inštrukcií jazyku *JavaScript*, ktoré sú nakoniec odoslané na vykonanie na klientskú stranu. Pre každý geo-objekt ktorý ma byť vykreslený na mape, je vytvorený objekt triedy *GoogleMarker*, ktorému sú následne nastavené všetky potrebné vlastnosti ako pozícia, vzhľad, informácie o ňom a iné. Takto pripravený objekt je nakoniec priradený objektu mapy.

Všetky funkcie triedy *MapPainter* fungujú na uvedenom princípe, líši sa len ich účel a spôsob využitia. Funkcia AddMarker a jej preťažené funkcie zabezpečujú pridanie jednoduchého markeru. Funkcie AddMarkerRet a AddMarkerRetAndZoom vykonávajú to isté s tým rozdielom, že hneď neodosielajú vygenerované inštrukcie v jazyku JavaScript, ale odovzdávajú ich ako návratovú hodnotu. Využíva sa to v prípade keď je potrebné pripojiť ešte ďalšie inštrukcie zabezpečujúce iné procesy s mapou.

### Klastrovanie geo-objektov



Klastrovanie geo-objektov zabezpečuje metóda *GetClusters* triedy *Clusterer*, ktorej argumentami sú geo-objekty určené pre klastrovanie, body určujúce ľavý dolný a pravý horný roh oblasti, v ktorej klastrovanie prebieha, počet iterácií a počet klastrov.

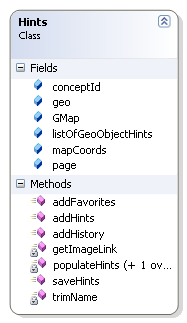
Prvým krokom je náhodné rozmiestnenenie centroidov klastrov v rámci oblasti zadanej ľavým dolným rohom a pravým horným rohom. Následne je v každej iterácií pre každý bod určený klaster s najbližsím centroidom. Po určení najbližšieho centroidu pre každý bod, je pre každý z klastrov určená nová pozícia jeho centroidu nasledovne:



Vzdialenosti bodov sú určované prostredníctvom Euklidovskej vzdialenosti.

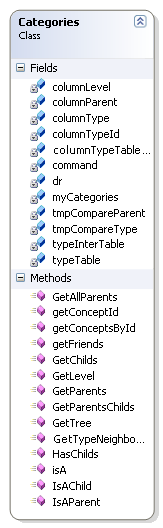
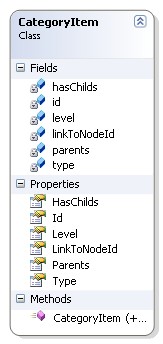
### Trieda Hints

Slúži na poskytnutie odporúčaní, obľúbených a histórie pre klienta. Poskytnutie je možné vyvolať pohybom po mape, prekliknutím tabulátorov označujúcich typ odporúčania a vyhľadávaním. Po prihlásení používateľa cez Facebook sa zobrazia nové objekty pridané priateľmi používateľa a iné odporúčania. Metóda *addFavorites* získa z databázy obľúbené položky používateľa a tie poskytne na výstupe. Metóda *addHints* zabezpečí získanie a spracovanie odporúčaní. Metóda *AddHistory* zase poskytne na výstupe históriu používateľa. Metóda *populateHints* slúži na naplnenie zoznamu obľúbených/histórie/odporúčaní získaného cez vyššie popísané metódy tak, že každú položku zoznamu vnorí do *div* elementu spolu s potrebnými dátami položky. Metóda *saveHints* slúži na uchovanie preferencií odporúčaní – lokalita, posledné vyhľadávanie a ktoré z odporúčaní sa má zobraziť. Metóda *trimName* sa používa na skrátenie názvu objektu v prípade, že je príliš dlhý.



### Triedy databázovej vrstvy

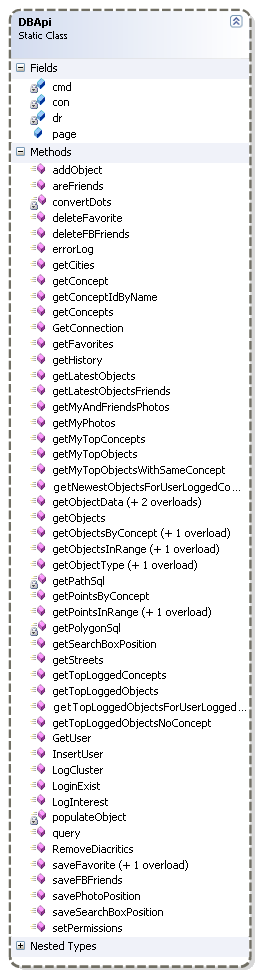
Dátová vrstva systému umožňuje prístup k databázovému systému PostgreSQL na vyššej úrovni. Databázové rozhranie je rozčlenené na triedu *Categories* atriedu *DBApi*. Na komunikáciu s databázou sa používa knižnica *Npgsql*.



Trieda *Categories* zabezpečuje prístup ku konceptuálnej mape pomocou metód zabezpečujúcich navigáciu v strome konceptov pomocou získania rodičov, potomkov resp. súrodencov daného konceptu. Väčšina výsledkov je prezentovaná ako inštancie triedy *CategoryItem*.

Metódou *GetAllParents* získame všetkých rodičov konceptu, *getFriends* vráti všetky koncepty so vzťahom k vstupnému konceptu, *GetChilds* všetkých potomkov, *GetParents* všetkých rodičov, *GetParentsChilds* aj potomkov aj rodičov, *GetTree* vráti priamych rodičov a všetkých potomkov konceptu, *GetTypeNeighbours* všetkých susedov konceptu – na rovnakej úrovni so spoločným rodičom. *HasChilds* zistí, či má koncept nejakého potomka, *IsA* zisťuje či existuje vzťah medzi dvom konceptami, *IsAChild* a *IsAParent* takisto pre vzťahy potomka a rodiča.

Trieda *DBApi* zabezpečuje prístup ku všetkým dátam uložených v databázovom systéme s výnimkou metód v triede *Categories*. Deje sa to pomocou metód zabezpečujúcich naplnenie údajov a ich získanie podľa zadaných podmienok. Trieda *DBApi* sa delí na viac častí:



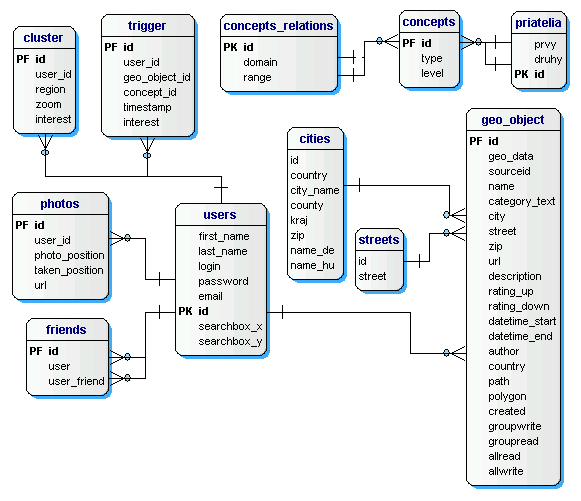
1. Všeobecná – nachádzajú sa tu metódy používané ostatnými časťami triedy. Trieda si drži jedno spojenie s databázou, aby sa predišlo vytváraniu veľkého počtu spojení. Na jeho získanie slúži metóda *GetConnection*. Na projekciu dát z databázy slúži všeobecná metóda *Query*, ktorá zabezpečí výber a získanie dát. Na evidenciu chybových hlásení slúži metóda *errorLog*.
2. Object Management – tu sú združené metódy na prácu súvisiacu s geo objektami. Najpoužívanejšia metóda je *populateObject*, ktorá slúži na naplnenie objektu typu *GeoObject* dátami z databázy. Na výber objektov z databázy slúžia metódy s rôznymi možnosťami vstupov *getObjectData*. Objekt je možné zaznamenať v databáze pomocou metódy *addObject*.
3. Autocomplete – nachádzajú sa tu metódy používané na službu automatického dopĺňania vyhľadávaného výrazu. Rôznymi metódami sa poskytujú uložené názvy objektov, konceptov, miest a ulíc z databázy. Ďalej sa poskytujú objekty alebo ich lokácie na základe nastavenej lokácie na mape v klientovi.
4. Authorization – časť používaná triedou *UserManagement* na získanie používateľských dát. Možné je zistiť, či existuje používateľ (*LoginExist*), získanie jeho údajov (*GetUser*) a vloženie nového používateľa (*InsertUser*).
5. Log Interests – časť, v ktorej sa nachádzajú metódy slúžiace na ukladanie profilu používateľa – aký typ objektov prehliada, v akých konceptoch sa orientuje najčastejšie (*LogInterest*) a po akej lokalite sa pohybuje najviac (*LogCluster*).
6. Hints – združuje metódy používané na výber a uchovávanie dát pre odporúčania, obľúbené a históriu. Mnohými metódami získavame potrebné dáta, ktoré sa následne v logickej časti spracujú a tým sa vytvoria odporúčania. Na ukladanie obľúbených slúži metóda *saveFavorite*, ich zmazanie sa deje pomocou *deleteFavorite*. Dáta pre históriu ako aj pre odporúčania sú uchovávané v časti 5 – okrem projekcie sa s nimi sa nijak ďalej nemanipuluje.
7. User profile – obsahuje metódy pre prácu s používateľským profilom a metódy na uchovávanie dát o fotkách používateľa. Metódou *setPermissions* sa nastavia na daný *GeoObject* povolenia zápisu a čítania, pre ostatných používateľov. Pri používateľovi prihlásenom cez *Facebook* sa dočasne uložia jeho priatelia cez metódu *saveFBFriends*. Po vypršaní *Session* používateľa sa tieto dáta zmažú pomocou metódy *deleteFBFriends*. Na získanie informácii o priateľoch používateľa sa používa metóda *areFriends*. Metódou *savePhotoPosition* sa uloží informácia o označenej fotke používateľa, ktoré je neskôr možné získať. Pri získaní je možné filtrovať medzi výberom len fotiek používateľa (*getMyPhotos*) alebo fotiek používateľa aj jeho priateľov (*getMyAndFriendsPhotos*).

### Dátový model

V tabuľke 2 sa nachádza zoznam používaných databázových tabuliek spolu s krátkym popisom, ktorý vyjadruje aké dáta daná tabuľka uchováva. Na obrázku 32 je znázornený používaný dátový model.

|  |  |
| --- | --- |
| **Tabuľka** | **Popis** |
| users | Používatelia registrovaný cez našu aplikáciu |
| friends | Dočasne uložené identifikátory priateľov používateľov prihlásených cez Facebook |
| geo\_object | Dáta o geografických objektoch |
| streets | Zoznam ulíc pre tabuľku geo\_object. |
| cities | Zoznam miest pre tabuľku geo\_object. |
| photos | Pozície fotiek na mape definované používateľom. |
| cluster | Zoznam oblastí, úrovne priblíženia a úrovne záujmu pre používateľa. |
| Trigger | Zoznam najprezeranejších objektov a konceptov spolu s úrovňou záujmu používateľa. |
| Concepts | Názvy konceptov a ich úroveň. |
| Concepts\_relations | Vzťahy konceptov pre stromovú štruktúru. |
| priatelia | Vzťahy konceptov pre odporúčanie podobných konceptov. |

Tabuľka 2 – Použité tabuľky a ich popis.



Obrázok 32 – Dátový model údajov.

# Overenie aplikácie

Overenie aplikácie prebiehalo pomocou testovacích scenárov a akceptačných testov. Najdôležitejšie z nich sú uvedené v tejto kapitole.

**Testovanie prebehlo 10. - 12. 5. 2010**

**Testovací scenár č.1:** Manipulácia s konceptuálnou mapou

**Testoval**: Dušan Torda

**Predpoklady:** Používateľ má spustenú aplikáciu a je prihlásený

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Očakávaný výsledok** |
| 1. | Používateľ kliknutie v hlavnom menu na pole konceptuálna mapa. | Zobrazí sa panel s grafom konceptuálnej mapy. | ok |
| 2. | Používateľ klikne na vybraný koncept (uzol grafu). | Na hlavnej mape sa zobrazia objekty zakliknutého konceptu. Konceptuálna mapa sa prekreslí tak, že zakliknutý koncept (uzol) sa nacentruje na stred grafu. A jeho farba sa zmení na modrú. | ok |
| 3. | Kliknutie na tlačidlo „Pridaj do obľúbených“, | Aktuálny uzol sa pridá do obľúbených položiek | ok |
| 4. | Kliknutie na tlačidlo „Strom“. | Graf sa zmizne a zobrazí sa hierarchický strom tak, že je ho aktuálny uzol je rovnaký ako bol v grafe | ok |

**Testovací scenár č.2:** Manipulácia s hierarchickým stromom

**Predpoklady:** Používateľ má spustenú aplikáciu a je prihlásený

**Testoval**: Dušan Torda

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Kliknutie na tlačidlo „Strom“ v paneli s grafom | Zobrazí sa panel s hierarchickým stromom | Ok |
| 2. | Používateľ klikne na ikonku pri vybranom koncepte (uzol grafu). | Ak na ikonku + tak sa rozbalia deti, potomkovia daného konceptu, ak na - tak sa skryjú, ak na samotný text uzlu tak sa na hlavnej mape vyhľadajú objekty zakliknutého konceptu. | Ok |
| 3. | Kliknutie na tlačidlo „Pridaj do obľúbených“, | Aktuálny uzol sa pridá do obľúbených položiek | ok |
| 4. | Kliknutie na tlačidlo „Graf“. | Strom sa zmizne a zobrazí sa graf tak, že je ho aktuálny uzol je rovnaký ako bol v strome | ok |

**Testovací scenár č.3:** Navigácia po mape

**Predpoklady:** Používateľ má spustenú aplikáciu

**Testoval**: Peter Abelovský

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Mapa je spustená. | Navigačný panel sa nachádza v pravom hornom rohu aplikácie. | ok |
| 2. | Používateľ klikne na šípku n navigačnom paneli. | Mapa sa posunie v smere definovanom šípkou. | ok |
| 3. | Používateľ klikne na tlačidlo + alebo – v navigačnom paneli. Dvojklik na myši pravým tlačidlom má rovnaký význam ako tlačidlo - v navigačnom paneli a dvojklik ľavým má rovnaký význam ako +. | Mapa sa priblíži ak je kliknuté na +, respektíve oddiali ak je kliknuté na -. | ok |
| 4. | Kliknutie a držanie pravého tlačidla na myši nad mapou a následné hýbanie myšou. | Mapa sa posúva v smere pohybu myši. | ok + po kliku sa na mape zobrazí šípka |

**Testovací scenár č.4:** Vyhľadávanie objektov na mape cez horný vyhľadávací panel

**Predpoklady:** Používateľ má spustenú aplikáciu

**Testoval**: Peter Abelovský

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Používateľ píše reťazec do poľa pred tlačidlom Hľadaj. | Zobrazuje sa ponuka s možnými variantmi písaného reťazca . | ok |
| 2. | Používateľ klikne na možnú variantu písaného výrazu | Písaný výraz sa doplní do podoby vybratej varianty. | po prejdení zoznamu suggestions pomocou dolnej šípky zmiznú krúžky a ak sa dostanem pomocou dolnej šípky úplne dole prestane zoznam na šípky reagovať  nefunguje rušenie bubliniek pomocou x |
| 3. | Reťazec, ktorý má byť nájdený je zapísaný v poli a klikne sa na tlačidlo Hľadaj alebo stlačí klávesu enter. | Na mape sa zobrazí hľadaný objekt respektíve objekty (samotné alebo vo forme clusterov ak ich je veľa) ak zadanému výrazu zodpovedá viacero objektov. Vo vyhľadávacom paneli sa zobrazí ikonka, ktorá definuje daný hľadaný výraz, teda o aký koncept, oblasť alebo objekt ide.  Ak je zadaný výraz koncept, tak sa konceptuálna mapa a strom prekreslí tak, že stredným uzlom grafu je daný koncept. | zadaný vyraz je koncept a pri zobrazenom strome sa tento nenastaví na vybraný koncept  reťazec ktorý ma byt nájdený je zapísaný ale po stlačení tlačidla sa vyhľadávanie nespusti, musí byt vybraný zo zoznamu |

**Testovací scenár č.5:** Manipulácia s klastermi

**Predpoklady:** Používateľ má spustenú aplikáciu, je vybratý daný koncept

**Testoval**: Peter Basár

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Vyberie sa koncept. | Na mape sú zobrazené klastre pre daný koncept. | Ok |
| 2. | Mapa sa priblíži / oddiali | Načítajú sa klastre v danom priblížení | Ok |
| 3. | Mapa sa posunie | Zobrazia sa klastre pre novú polohu mapy. | Ok |
| 4. | Používateľ klikne na klaster | Klastre sa prekreslia alebo sa zobrazia už samotné objekty ak ich počet už nie je v danej oblasti veľký | ok |

**Testovací scenár č.6:** Úprava používateľského prostredia

**Predpoklady:** Používateľ má spustenú aplikáciu

**Testoval**: Peter Basár

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Chytením ikonky [+] vo vyhľadávacom paneli používateľ metódou DragAndDrop posunie panel po obrazovke | Panel sa posunie na nové miesto. | ok |
| 2. | Používateľ klikne na šípku v pravom hornom rohu hlavného panela | Hlavný panel sa skryje | ok |
| 3. | Používateľ klikne na šípku v pravom hornom rohu skrytého hlavného panela | Hlavný panel sa vysunie | ok |

**Testovací scenár č.7:** Prihlásenie sa používateľa

**Predpoklady:** Používateľ má spustenú aplikáciu, používateľ je zaregistrovaný

**Testoval**: Marián Hraško

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Používateľ na pole Prihlásenie vo vyhľadávacom paneli | Rozbalí sa ponuka s prihlásením. | Ok |
| 2. | Používateľ zadá do vyznačených miest svoje meno a heslo a klikne na tlačidlo Prihlásiť. | Ak je používateľské meno a heslo správne, vypíše sa používateľovi v navigačnom panely “ste prihlásený ako Používateľské meno dotyčného“.  Ak sú prihlasovacie údaje nesprávne, vypíše sa správa „Nesprávne meno alebo heslo“. | ok |

**Testovací scenár č.8:** Prihlásenie sa používateľa cez facebook

**Predpoklady:** Používateľ má spustenú aplikáciu, používateľ je zaregistrovaný na facebooku

**Testoval**: Marián Hraško

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Používateľ na pole Prihlásenie cez facebook vo vyhľadávacom paneli | Rozbalí sa ponuka s prihlásením. | ok |
| 2. | Používateľ zadá do vyznačených miest svoje meno a heslo a klikne na tlačidlo Prihlásiť. | Ak je používateľské meno a heslo správne, vypíše sa používateľovi vo vyhľadávacom panely “ste prihlásený ako Používateľské meno dotyčného“.  Ak sú prihlasovacie údaje nesprávne, vypíše sa správa „Nesprávne meno alebo heslo“. | ok |

**Testovací scenár č.9:** Registrácia používateľa

**Predpoklady:** Používateľ má spustenú aplikáciu

**Testoval**: Marián Hraško

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Používateľ klikne vo vyhľadávacom paneli na pole Registrácia | Zobrazí sa registračný formulár. | Ok |
| 2. | Používateľ klikne na políčko Registrácia | Zobrazí sa registračný formulár. | Ok |
| 3. | Používateľ vyplní formulár a klikne na tlačidlo Registruj. | Ak sú údaje správne, vypíše sa používateľovi “ďakujeme za registráciu“ a používateľ je zaregistrovaný, formulár sa zavrie a systém odošle používateľovi email s jeho prihlasovacími údajmi.  Ak sú údaje nesprávne, vypíše sa správa „Nesprávne vyplnené údaje“ a označia sa červenou farbou políčka, ktoré treba opraviť. | Ok |
| 4. | Používateľ klikne na tlačidlo Zrušiť v registračnom formulári. | Zatvorí sa registračný formulár. | Ok |

**Testovací scenár č.10:** Panel moje objekty

**Predpoklady:** Používateľ má spustenú aplikáciu a je prihlásený

**Testoval**: Dušan Torda

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Používateľ kliknutie v hlavnom menu na pole Moje objekty | Zobrazí sa panel so zoznamov, históriou prezeraných objektov | ok |
| 2. | Používateľ klikne na položku v zozname | Položka sa vyhľadá na mape | Nejde prihlásenie tak je zoznam prázdny |
| 3. | Používateľ kliknutie na záložku Obľúbené | Zobrazí sa zoznam s obľúbenými položkami používateľa | Nejde prihlásenie tak je zoznam prázdny |
| 4. | Používateľ klikne na položku v zozname | Položka sa vyhľadá na mape | Nejde prihlásenie tak je zoznam prázdny |
| 5. | Používateľ kliknutie na záložku Odporúčané | Zobrazí sa zoznam s položkami, ktoré systém používateľovi odporúča | Nejde prihlásenie tak je zoznam prázdny |
| 6. | Používateľ klikne na položku v zozname | Položka sa vyhľadá na mape | Nejde prihlásenie tak je zoznam prázdny |

**Testovací scenár č.11:** Práca s objektami

**Predpoklady:** Používateľ má spustenú aplikáciu a je prihlásený

**Testoval**: Kamil Bartal

**Priebeh testu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Krok** | **Akcia** | **Očakávaný výsledok** | **Skutočný výsledok** |
| 1. | Používateľ kliknutie v hlavnom menu na pole Práca s objektami. | Zobrazí sa panel s pre manažment objektov | Ok |
| 2. | Klikne na miesto na mape kde chce pridať objekt | Na mape sa zobrazí ikonka | Ok |
| 3. | Používateľ vyplní údaje o objekte a Klikne na tlačidlo Pridaj | Objekt sa uloží do databázy | Ok |
| 4. | Používateľ vyhľadá pridaný objekt cez vyhľadávací panel | Daný objekt sa vyhľadá na mape | Ok |
| 5. | Používateľ príde myšou nad vyhľadaný objekt | Zobrazí sa okno s informáciami o objekte, ktoré používateľ zadal pri pridávaní objektu | ok |

# Používateľská príručka

## Spustenie aplikácie

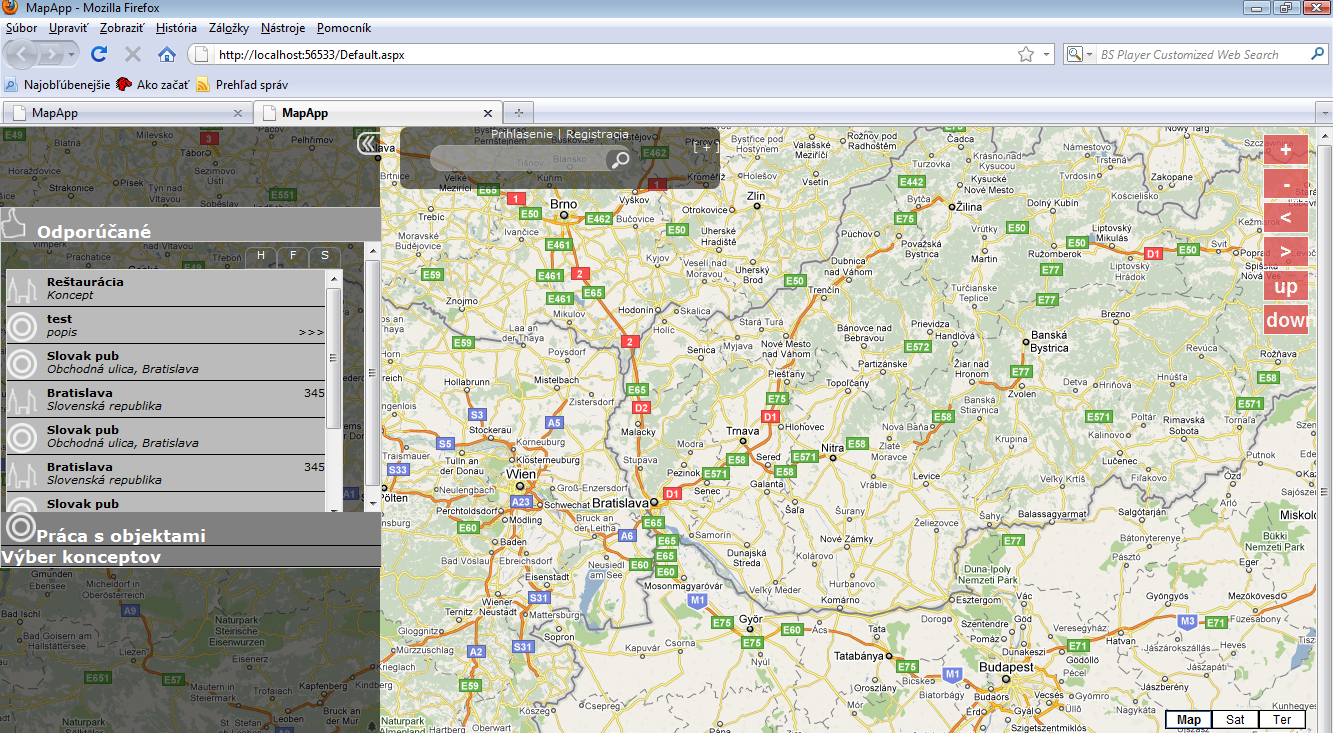
Prototypovú aplikáciu možno spustiť v internetovom prehliadači na adrese:

<http://147.175.159.183/Default.aspx>

Na spustenie treba mať nainštalovanú podporu pre Adobe Flash Player, pre zobrazenie konceptuálnej mapy.

## Ovládacie prvky aplikácie

Na nasledujúcom obrázku je zmenšený celkový pohľad na aplikáciu. Skladá sa z nasledovných hlavných častí: plochy na zobrazovanie mapy , ovládacieho panelu (vľavo), vyhľadávacieho panelu (hore) a tlačidiel na ovládanie mapy(vpravo).



Obrázok 33 - Náhľad na aplikáciu.

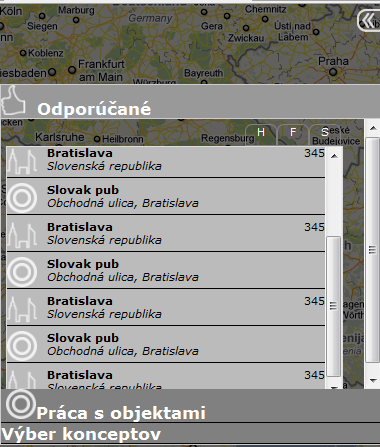
### Pole s mapou

Mapu používateľ ovláda prostredníctvom kurzorového zariadenia v pravom hornom rohu mapy. Spôsob ovládania je rovnaký ako v ostatných aplikáciách využívajúcich mapy od spoločnosti Google. Mapa umožňuje taktiež posun prostredníctvom „chytenia a ťahania“, rôzne priblíženie možno dosiahnuť s použitím dvojklikov pravého respektíve ľavého tlačidla na myši alebo scrollovacím tlačidlom.

### Ovládací panel

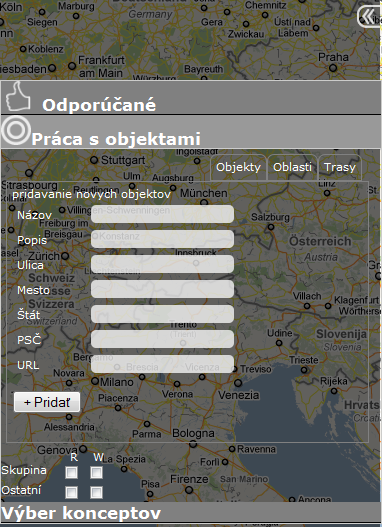
Ovládací panel tvoria tri oddelenia, v paneli uložených do roletiek. Ovládací panel sa dá skryť tlačidlo v pravom hornom rohu panela. Následne sa dá zobraziť tým istým tlačidlom.

**Odporúčané** – Obsahuje položky (objekty a kategórie), ktoré sú používateľovi odporúčané alebo sú medzi jeho obľúbenými položkami alebo v histórii. Je to zoznam položiek, ktoré sa skladajú z mena, popisu a obrázku kategórie, do ktorej patria. Zoznam sa dá scrollovať. V hornej časti sú roletky kde si používateľ vyberá typ H, F, S. S sú odporúčané položky, ktoré sú generované automaticky na základe používateľovej lokácie, priateľov, hľadaného konceptu . F sú používateľove obľúbené položky, ktoré si sám pridal. H predstavuje históriu položiek prehľadávaných používateľom.



Obrázok 34 - Panel Odporúčané

**Práca s objektmi** – obsahuje rozhranie pre prácu s objektmi. Pre pridanie objektu používateľ klikne na miesto na mape, kde chce pridať objekt, čím určí súradnice objektu. Následne vyplní názov objektu a popis objektu. V spodnej časti si vyberá nastavenie práv pre používanie objektu, či ho môžu prezerať alebo editovať iba priatelia alebo všetci. Po vyplnení informácií klikne na tlačidlo “+ Pridaj”. Na nasledujúcom obrázku je zobrazené prostredie pre pridávanie objektov. V hornej časti sa nachádzajú roletky kde si používateľ môže vybrať, či chce pracovať s objekt, trasou alebo oblasťami.



Obrázok 35 - Zobrazenie prostredia pre pridávanie objektov.

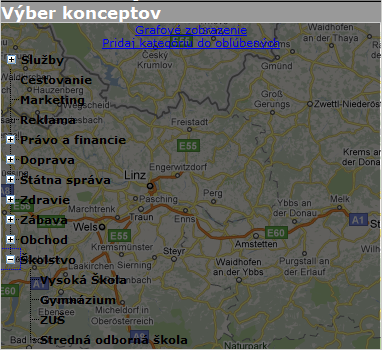
**Výber konceptov** – Vo vrchnej časti sú dve tlačidlá „Stromové zobrazenie” respektíve „Grafové zobrazenie”, podľa toho či je v panely zobrazený Konceptuálny graf alebo hierarchický strom. Toto tlačidlo slúži na prepínanie medzi nimi, a tlačidlo „Pridaj kategóriu medzi obľúbené“, ktoré pridá aktuálne označenú kategóriu medzi používateľove obľúbené položky, či už je v strome alebo v grafe.

**Konceptuálny graf** slúži na preklikávanie sa medzi konceptmi a ich podkategóriami, ktoré sa následne zobrazujú na mape. Rozloženie prvkov je také, že aktuálny koncept je v strede grafu označený modrou farbou a okolo neho sú jeho príbuzné uzly (príbuzné kategórie) označené zelenou farbou a nadkategória (rodič) označená červenou farbou, v hornej časti grafu. Objekty danej kategórie, ktorá predstavuje aktuálny uzol sú vyhľadané na mape v aktuálnej lokácii. Na nasledujúcom obrázku je zobrazený príklad zobrazenia grafu.



Obrázok 36 – Zobrazenie konceptuálnej mapy

**Hierarchický strom** obsahuje celú štruktúru kategórií. Kliknutím na uzol v hierarchickom uzle sa zobrazia jeho deti a opätovným kliknutím sa tieto deti stratia. Objekty danej kategórie, ktorá predstavuje aktuálny uzol v strome sú vyhľadané na mape v aktuálnej lokácii. Na nasledujúcom obrázku je zobrazený príklad hierarchického stromu.



Obrázok 37 – Zobrazenie hierarchického stromu

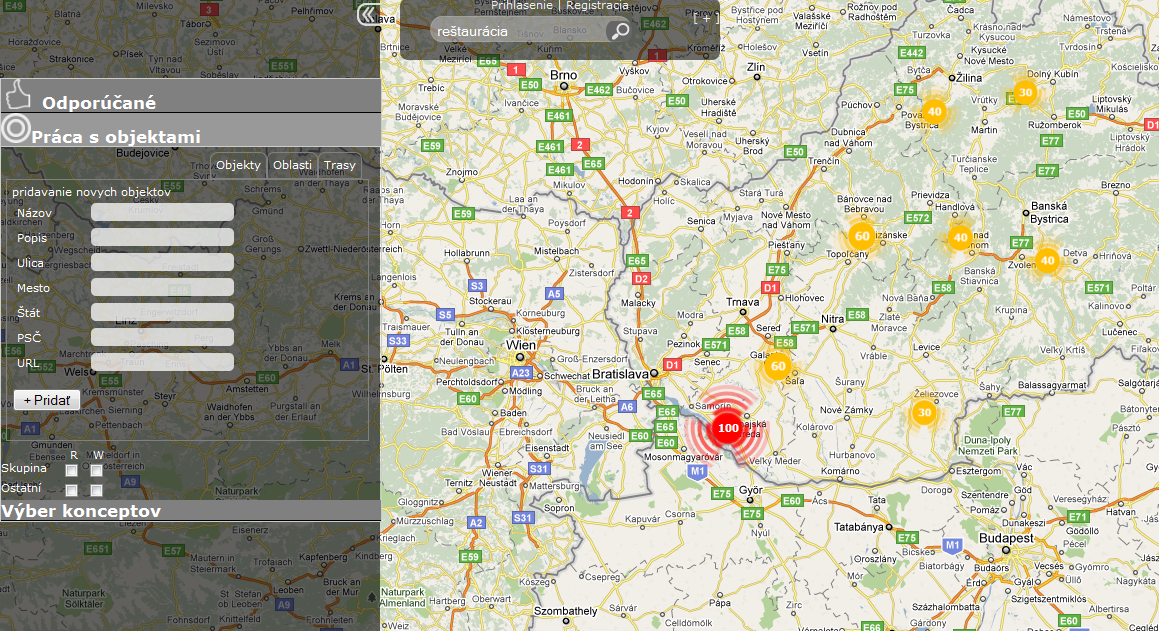
### Vyhľadávací panel

Vyhľadávací panel sa dá ľubovoľne posúvať po okne chytením a ťahaním myšou.

Hore vo vyhľadávacom paneli sú odkazy na prihlásenie a registráciu. Po kliknutí na odkaz **Prihlásenie**  vyskočí používateľovi okno, ktoré umožňuje prihlásenie sa. Pre prihlásenie sa do systému požívateľ vyplní používateľské meno a heslo a potvrdí prihlásenie. Nový používateľ sa do systému registruje kliknutí na pole **Registrácia** a následným vyplnením prihlasovacieho formulára.

V paneli sa nachádza textové pole pre vyhľadávanie. Do textového poľa sa zapíše výraz, ktorý chce používateľ vyhľadať. Popri písaní daného výrazu je používateľovi ponúkaný zoznam možných doplnení zadávaného výrazu. Po dopísaní výrazu alebo vybratí z ponuky možných doplnení používateľ klikne na obrázok s lupou a následne sa na mape zobrazia objekty vyhovujúce zadanému výrazu. Ak ich je veľa v danej lokácii, tak sa zobrazia vo forme klastrov. Po priblížení mapy na príslušnú úroveň priblíženia sa z klastrov vytvoria samotné objekty. Príklad je zobrazený na nasledujúcom obrázku.

Pri písaní do textového poľa systém dáva používateľovi nápovedu, čo asi môže hľadať. Teda rozpoznáva o aký druh objektu ide, či a o aký koncept, mesto, lokáciu ide. Táto sa zobrazí formou tlačidla pod textové pole a používateľ ho môže zrušiť krížikom alebo pridať do obľúbených alebo naň klikne a vyhľadajú sa dané položky.



Obrázok 38 – Zobrazenie výsledkov vyhľadávania.

# Finálny produkt – systémová príručka

Táto kapitola slúži ako príručka na nainštalovanie aplikácie. Sú v nej opísané postupy, nastavenia a prostredie potrebné k inštalácii aplikácie. Inštalácia predpokladá prístup na prostredie *labss2*.

Na pridelenom serveri v prostredí *labss2* je nainštalovaný operačný systém *Windows Server 2003*. Server nie je dostupný z internetu, preto je pre prístup naň potrebné mať založené konto na serveri *labss2.fiit.stuba.sk* (ďalej len *labss2*). Aplikácia na strane servera je implementovaná v technológii *.NET*, preto je potrebné overiť dostupnosť technológie na serveri. Aplikácia používa databázový server *PostgreSQL*.

## Získanie zdrojových kódov aplikácie

Projekt využíva *SVN* dostupné na *https://147.175.159.183/svn/tp/trunk/solution*. Na získanie súborov odporúčame použiť klienta *Tortoise SVN*.

## Inštalácia aplikačného servera

Na úspešné nainštalovanie aplikačného servera je potrebné vykonať nasledujúce kroky:

1. Publish solution vo Visual Studio.
2. Upload publikovaného adresára do wwwroot adresára na serveri.

## Inštalácia databázy

Databázu je potrebné nainštalovať na rovnaký server. Inštalačné súbory k databázovému serveru *PostgreSQL* je možné získať na adrese *http://www.enterprisedb.com/products/pgdownload.do#windows.* Po úspešnom nainštalovaní je potrebné nasadiť schému aplikácie.

# Príloha A : Opis dátového disku

Priložené CD obsahuje nasledovné položky:

* zdrojové kódy (server, klient)
* nasaditeľná verzia – publish z Visual Studio
* knižnice
* dokumentácia (zimný, letný semester)

Adresárová štruktúra je nasledovná:

doc/ - kompletná dokumentácia

doc/summer - dokumentácia k letnému semestru

doc/winter - dokumentácia k zimnému semestru

lib/ - obsahuje potrebné knižnice

src/ - obsahuje všetky zdrojové kódy