

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

Evidenčné číslo: FEI-5382-52598

**VYUŽITIE GRAFOVEJ DATABÁZY V PRAXI
BAKALÁRSKA PRÁCA**

2017

Juraj Kubričan

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-5382-52598

VYUŽITIE GRAFOVEJ DATABÁZY V PRAXI
BAKALÁRSKA PRÁCA

Študijný program: Aplikovaná informatika
Číslo študijného odboru: 2511
Názov študijného odboru: 9.2.9 Aplikovaná informatika
Školiace pracovisko: Ústav informatiky a matematiky
Vedúci záverečnej práce: Ing. Maroš Čavojský

Bratislava 2017

Juraj Kubričan



ZADANIE BAKALÁRSKEJ PRÁCE

Študent: **Juraj Kubričan**
ID študenta: **52598**
Študijný program: **aplikovaná informatika**
Študijný odbor: **9.2.9. aplikovaná informatika**
Vedúci práce: **Ing. Maroš Čavojský**
Miesto vypracovania: **Ústav informatiky a matematiky**

Názov práce: **Využitie grafovej databázy v praxi**

Jazyk, v ktorom sa práca vypracuje: **slovenský jazyk**

Špecifikácia zadania:

V dnešnej dobe sa okrem tradičných zaužívaných relačných databáz, využívajú aj menej známe grafové databázy, v ktorých sú dáta uložené odlišným spôsobom ako v relačných databázach. Cieľom práce je oboznámiť sa s jednotlivými predstaviteľmi grafových databáz, vybrať jedného a navrhnúť a implementovať využitie vybranej grafovej databázy na reálnom príklade.

Úlohy:

1. Naštudujte si literatúru ohľadom jednotlivých predstaviteľov grafových databáz
2. Vyberte jedného predstaviteľa grafových databáz
3. Navrhnite reálny príklad pre implementáciu grafovej databázy
4. Implementujte reálny príklad pre implementáciu grafovej databázy
5. Zhodnoťte a uveďte výhody použitia grafovej databázy oproti iným typom databáz (relačné, dokumentové,...) v implementovanom reálnom príklade

Zoznam odbornej literatúry:


1. Ian Robinson, Jim Webber, and Emil Eifrem: Graph Databases, O'Reilly Media, Inc. USA 2015, p.224, ISBN: 978-1-491-93200-1


Riešenie zadania práce od: **19. 09. 2016**

Dátum odovzdania práce: **19. 05. 2017**


Juraj Kubričan
študent

SLOVENSKÁ TECHNICKÁ UNIVERZITA
V BRATISLAVE
Fakulta elektrotechniky a informatiky
Ústav informatiky a matematiky
Ilkovičova 3, 812 19 Bratislava


prof. RNDr. Otokar Grošek, PhD.
vedúci pracoviska


prof. Dr. Ing. Miloš Oravec
garant študijného programu

SÚHRN

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Študijný program:	Aplikovaná informatika
Autor:	Juraj Kubričan
Bakalárska práca:	Využitie grafovej databázy v praxi
Vedúci záverečnej práce:	Ing. Maroš Čavojský
Miesto a rok predloženia práce:	Bratislava 2017

Práca sa zaoberá vytvorením webovej aplikácie ktorá bude využívať grafovú databázu. V prvej časti sa nachádza prehľad technológií, ktoré sme použili na implementáciu projektu, ďalej sa tu nachádza priblíženie najpopulárnejších zastupiteľov grafových databáz, ich výhod a nevýhod. V ďalšej časti sa nachádza špecifikácia našej aplikácie cestovného plánovača.

Kľúčové slová: Využitie grafovej databázy v praxi

ABSTRACT

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

Study Programme:	Applied Informatics
Author:	Juraj Kubričan
Bachelor Thesis:	Graph database use in a real world application
Supervisor:	Ing. Maroš Čavojský
Place and year of submission:	Bratislava 2017

The bachelor thesis is about creating of a plugin for web browser, that modifies information used to identification of user during accessing a server. There is an overview of methods that increase anonymity during browsing websites, in the first part. The thesis also contains a list of the most used extensions nowadays, that function is a change of some identification components of browser or special ways of anonymization. In the next part of the thesis is an overview of the characteristics of web browser. By combination of these characteristics we can with high level of success identify a user, who have visited the web site. The last part of thesis contains project, implementation and testing of extension created for the web browser Mozilla Firefox. There is also description of source code of extension, the link between the characteristics of web browser, detected limitations and way how to solve them. The resulting extension increases anonymity of user by modification of some characteristic components of web browser or by blocking sending components, that can not be in extension changed. In comparison with most used modules nowadays, this module can modify HTTP headers including characteristics detected by JavaScript commands.

Keywords: Graph database use in a real world application

Obsah

Úvod	1
1 Analýza problému	2
1.1 Cieľ práce	2
1.2 Relačné	2
1.3 Grafové	2
2 Špecifikácia	3
3 Vývoj	5
3.1 Návrh	5
3.1.1 Usecase diagramy	5
3.1.2 Class diagram	5
3.1.3 Role hráčov	5
3.2 Implementácia	5
3.2.1 Výber grafovej databázy	5
4 Implementácia	10
4.1 Použíte technológie	10
4.1.1 Laravel framework	10
4.1.2 NeoEloquent OGM	10
4.1.3 Mapbox.js	10
4.1.4 Rome2Rio Api	10
4.2 inštalácia a nastavnie Laravel Framework-u	10
4.3 Inšatlácia a Konfigurácia Databázy neo4j	11
4.4 OGM	11
4.5 Autentifikácia	12
4.6 Smerovač	12
4.7 Vzhľad	12
Záver	15
Zoznam použitej literatúry	16
Prílohy	I

Zoznam obrázkov a tabuliek

Obrázok 1	Usecase - Registrácia, autentikácia a nastavenia	6
Obrázok 2	Usecase - dashboard	7
Obrázok 3	Usecase - tsp	7
Obrázok 4	Class diagram	8

Zoznam skratiek a značiek

TSP - The Travelling Salesman Problem

UML - Unified Modeling Language verzie 2.5

GDBMS RDBMS GPLv3 ORM - Object-Relational Mapping OGM - Object-Graph Mapping URL SSH - Secure Shell WebScket

Zoznam algoritmov

1	Ukážka algoritmu	11
2	Ukážka algoritmu	13
3	Ukážka algoritmu	14

Úvod

Pri návrhu aplikácie treba myslieť na štruktúru dát a podľa toho vybrať správnu databázu/DBMS. Správny výber DBMS vie zabezpečiť rádovo nižšie prístupové časy a tým aj väčšiu scalability. Pri aplikáciách kde sú dáta štruktúrované do uzlov a prepojení, je vhodné zvážiť použitie grafovej databázy. My sme navrhli a naimplementovali aplikáciu cestovného plánovača, ktorý potrebuje uchovávať dáta o destináciách a cestách medzi nimi. Preto je pre túto aplikáciu vhodné využiť grafový DBMS

1 Analýza problému

1.1 Cieľ práce

Cieľom práce je navrhnúť a na implementovať aplikáciu, ktorá využíva grafovú databázu, ďalej zhodnotiť výhody tohto riešenia oproti iným typom databáz.

1.2 Relačné

Relačná databáza je databáza, v ktorej sú údaje uložené podľa relačného databázového modelu podľa E. F. Codd z roku 1970. Podľa Relačného modelu sú dáta uložené v tabuľkách. Jeden riadok tabuľky je jeden záznam. Stĺpec tabuľky reprezentuje jeden atribút objektu. Väzby (vzťahy) medzi tabuľkami sú riešené pomocou unikátnych identifikátorov tzv. kľúčov. Jheda tabuľka obsahuje kľúč, čo je atribút ktorý unikátne identifikuje každý záznam a druhá tabuľka obsahuje tzv cudzí kľúč, atribút ktorý odkazuje na záznam v prvej tabuľke. Výhodou tohto spôsobu ukladania dát je jednoduchosť a prehľadnosť. Nevýhoda tohto prístupu sa však ukazuje v škálovateľnosti. Pri vyhľadávaní každé toto prepojenie pridáva výpočtovú komplexitu, keďže v každej ďalšej prepojennej tabuľke treba vyhľadať záznam s požadovaným kľúčom ($O(\log(n))$). Všetky používané relačné databázové systémy riešia tento problém škálovateľnosti použitím indexov a rôznymi inými optimalizáciami, no pokiaľ sú naše dáta štrukturované s veľa prepojeniami systém sa spomalí.

1.3 Grafové

V grafovej databáze sú údaje štrukturované vo vrchoch(node) a hranách(edge). Prepojenia medzi vrcholmi sú realizované priamo pomocou hrán. Grafové databázové systémy podporujú vlastné sémantické prostriedky ktoré umožňujú priamu manipuláciu so štruktúrou dát. Jedna z hlavných výhod Táto štruktúra umožňuje priamo

2 Špecifikácia

1. Funkcionálne požiadavky

- (a) Aplikácia bude umožňovať registráciu a prihlásenie používateľa
- (b) Pri registrácii sa budú vyžadovať prihlasovacie údaje: e-mail, heslo. Okrem toho sa bude vyžadovať zadanie mena a domáceho miesta. Toto domáce miesto bude možné vyhľadať v online databáze.
- (c) Po prihlásení používateľa sa mu zobrazí obrazovka s mapou, zoznamom obľúbených destinácií, ktoré chce navštíviť a zoznam odporúčaných destinácií
- (d) Na mape bude vyobrazené používateľove domáce miesto a všetky destinácie, ktoré má v zozname obľúbených destinácií. Po kliknutí na destináciu sa používateľovi otvorí príslušný riadok v zozname obľúbených.
- (e) V zozname obľúbených budú všetky miesta, ktoré si používateľ pridal. Zoznam bude vo forme tabuľky, riadok bude obsahovať meno destinácie a lokalitu, v ktorej sa destinácia nachádza. V riadku tiež bude tlačidlo na vymazanie destinácie z obľúbených a tlačidlo na zobrazenie detailov.
- (f) V detailoch obľúbeného miesta bude zoznam ostatných ľudí, ktorí dané miesto majú v obľúbených a výpis možných trás z domáceho miesta používateľa do destinácie.
- (g) V zozname odporúčaných destinácií budú destinácie, ktoré majú v obľúbených používatelia, ktorí majú v obľúbených rovnaké miesta ako miesta, ktoré má v obľúbených prihlásený používateľ. Vynechané budú miesta, ktoré už prihlásený používateľ má obľúbených. Každá položka z odporúčaných sa bude dať jednoducho pridať do obľúbených prihláseného používateľa.
- (h) V aplikácii bude obrazovka s nastaveniami, na ktorej si bude človek môcť zmeniť heslo, domáce miesto.
- (i) V aplikácii bude obrazovka, kde si bude používateľ vybrať niekoľko zo svojich obľúbených miest a nechať si vyrátať najkratšiu trasu z domáceho miesta cez všetky zvolené miesta a potom späť. (TSP)

2. Nefunkcionálne požiadavky

- (a) Systém bude zrealizovaný na webovej platforme.
- (b) Aplikácia bude využívať natívnu grafickú databázu.

- (c) Aplikácia bude byť kompatibilná s webovými prehliadačmi Google Chrome Mozilla Firefox, Microsoft Edge, Microsoft Internet Explorer.
- (d) Užívateľské rozhranie systém musí byť plne použiteľné aj na mobilných telefónoch s OS Android a IOS.
- (e) Aplikácia bude implementovaný s použitím jayka PHP a PHP frameworku.
- (f) Systém bude nasadený na virtuálnom serveri s operačným systémom Ubuntu 16.04.2 LTS poskytnutom Ústavom informatiky a matematiky FEI STU.

3 Vývoj

3.1 Návrh

V časti návrhu projektu popíšeme prípady použitia,

3.1.1 Usecase diagramy

3.1.2 Class diagram

3.1.3 Role hráčov

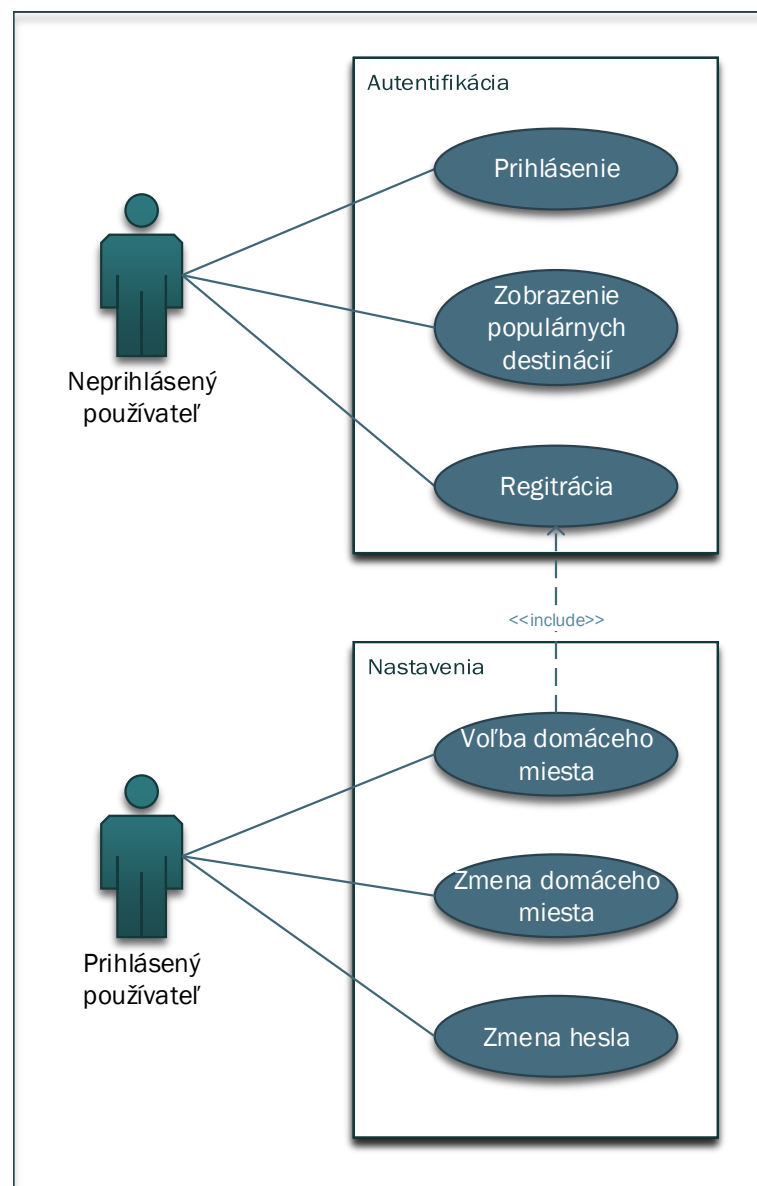
1. Neprihlásený používateľ bude mať prístup na úvodnú stránku. Na úvodnej stránke sa bude môcť buď zaregistrovať, prihlásiť ak už má vytvorený účet, alebo si bude môcť pozrieť mapu a zoznam s najpopulárnejšími destináciami používateľov aplikácie.
2. Prihlásený používateľ má prístup do štyroch subsystémov
 - (a) Dashboard je domáca obrazovka používateľa bude môcť na nej vyhľadávať, pridávať a odstraňovať miesta medzi svoje obľúbené. Ďalej si bude môcť pozrieť detaily destinácie ako rôzne trasy ktoré vedú z jeho domáceho miesta do destinácie a zoznam ostatných používateľov, ktorí majú rovnaké miesta v obľúbených. Kliknutím na používateľa bude môcť prejsť na obrazovku používateľa.
 - (b) Na obrazovke nastavení si bude môcť používateľ nastaviť heslo a zmeniť svoje domáce miesto.
 - (c) Na obrazovke TSP si bude používateľ môcť naplánovať trasu medzi niektorými zo svojich obľúbených miest. Bude si môcť pridávať miesta do trasy a odoberať ich. Ďalej si bude môcť prezrieť detaily aktuálnej trasy s cenami a mapu s vyobrazenou trasou.
 - (d) Na obrazovke používateľa si bude prihlásený používateľ prezrieť obľúbené destinácie konkrétneho používateľa na mape a v zozname.

3.2 Implementácia

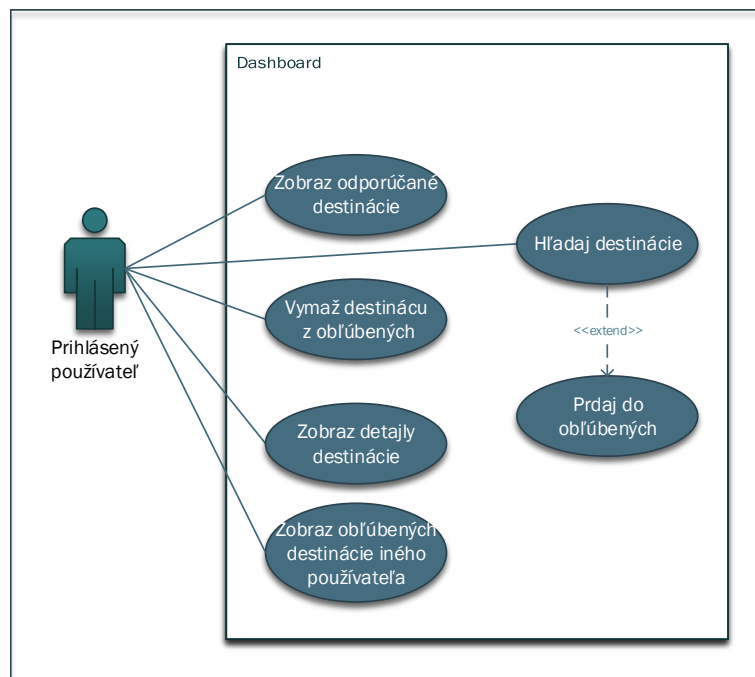
3.2.1 Výber grafovej databázy

Vybrali sme si troch najpopulárnejších predstaviteľov grafových databáz podľa [1] rebríčka na DB-Engines.com. V nasledujúcej v skratke časti priblížime históriu každého GDBMS ich výhody, nevýhody pre naše použitie a proces výberu použitej databázy.

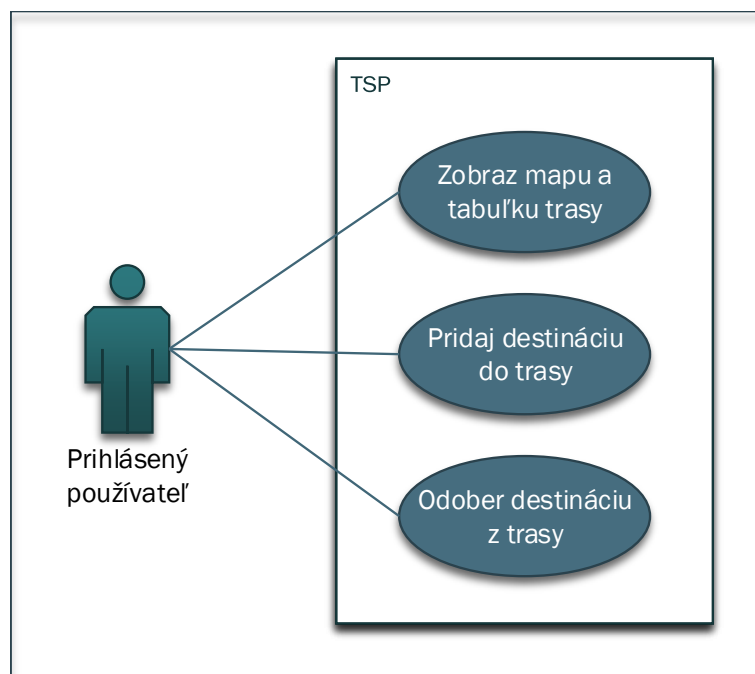
1. NEO4J Prvá verzia Neo4J bola vydaná v roku 2007 od vtedy sa stala dlhodobo najpoužívanejšou grafovou databázou. Je vyvíjaná Neo Technology, Inc. Neo4j je



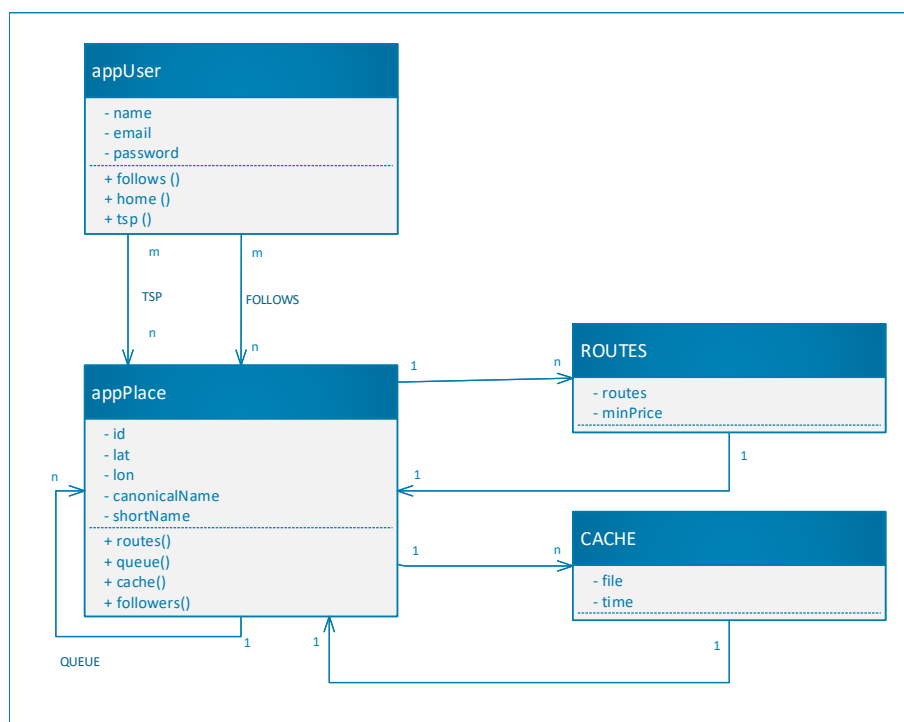
Obrázok 1: Usecase - Registrácia, autentifikácia a nastavenia



Obrázok 2: Usecase - dashboard



Obrázok 3: Usecase - tsp



Obrázok 4: Class diagram

ponúkaná v dvoch variantách: Neo4j Community je open source (GPLv3) grafová databáza obsahujúca všetky základné funkcie (Ďalej budeme spomínať len túto verziu). A Neo4j Enterprise edícia, ktorá má rozšírené funkcie ako Shardovanie cache pamäte, rozšírené monitorovanie a zálohovanie za behu.

Medzi hlavné výhody neo4j patrí to, že je to dlhodobo najrozšírenejšia grafová databáza, má dobrú dokumentáciu, podporuje mapovanie na objekty. Nevýhodou je, že podporuje iba grafový model ukladania údajov.

2. OrientDB OrientDB je vyvíjané od roku 2010 firmou OrientDB LTD. Databáza OrientDB rýchlo nabrala na popularite a v roku 2015 sa dostala na druhé miesto rebríčky db-engines. OrientDB sa rovnako ako Neo4j distribuuje v dvoch edíciach: Community - open source (Apache Licence 2.0) so základnými funkciami a Enterprise edíciou s podpodou migrácie a synchronizácie na Neo4J a pridanými analytickými nástrojmi.

Výhodou OrientDB je podpora okrem grafového modelu ukladania dát aj dokumentový a key/value model ukladania údajov.

Nevýhodou je horšia podpora pre nami vybraný framework Laravel

3. Titan Titan bol od 2012 vyvíjaný skupinou thinkaurelius, no v roku 2017 bol odkúpený firmou Datstax a projekt Titan bol zastavený. Projekt je ďalej udržiavaný ako open source verzia pod menom JanusGraph. Titan je projekt určený na veľké distribuované enterprise riešenia, je nasadzovaný na cloudové platformy ako napr. Apache Hadoop, Apache Spark, ... podporuje rôzne distribuované úložné priestory ako napr. Apache HBase, Oracle BerkeleyDB, ... Ďalej podporuje rôzne vyhľadávacie enginy ako napr. Elasticsearch, Solr, ...
Toto riešenie je pre naše použitie nevhodné, lebo na správne fungovanie vyžaduje distribuovaný systém.

4 Implementácia

4.1 Použité technológie

4.1.1 Laravel framework

Laravel Framework je komplexný voľne šíriteľný framework. Tento framework je od roku 2015 najpopulárnejší PHP framework. Medzi jeho hlavné výhody patrí použitie relatívne novej verzie PHP 5.4, ktorá obsahuje technológie, ktoré v minulosti v PHP chýbali ako menové priestory a anonymné funkcie. Dalej obsahuje veľmi silný nástroj Eloquent ORM pre objektovo relačné mapovanie databázy, Blade šablónovací nástroj na rýchlu tvorbu dynamického obsahu.

Komplexita Frameworku Laravel je však aj jeho hlavnou nevýhodou, nieje vhodný na menšie projekty. V rýchlosti patrí medzi priemer PHP frameworkov.

4.1.2 NeoEloquent OGM

NeoEloquent OGM je voľne šíriteľná knižnica, ktorá umožňuje využívať grafovú databázu neo4j spolu s existujúcim dátovým modelom vo frameworku Laravel. Štruktúra NeoEloquent je modelovaná podľa existujúceho Eloquent Modelu a preto je jeho integrácia do ekosystému Laravel

4.1.3 Mapbox.js

Mapbox.js je komerčná knižnica na vytváranie projektov s interaktívnymi mapami. Je založená na voľne šíriteľnej knižnici Leaflet, rozširuje túto knižnicu o funkcie ako automatické zoskupovanie bodov do skupín a poskytuje bohatú a prehľadnú dokumentáciu. My používame verziu zdarma, ktorá je obmedzená na 50000 zobrazení mape na mesiac. Mapbox.js podporuje štandardný formát dát GeoJSON, tento formát umožňuje ukladať dáta o polohe, type bodu, rôznych atribútoch upresňujúcich vizuál zobrazovaného bodu. Tento formát ďalej umožňuje ukladať geometrické útvary a krivky.

4.1.4 Rome2Rio Api

Rome2Rio je portál ktorý zbiera údaje o cenách dopravy po celom svete a umožňuje vyhľadať cenu cesty medzi dvomi ľubovoľnými destináciami. Rome2Rio taktiež poskytuje niekoľko otvorených a platených API.

4.2 inštalácia a nastavnie Laravel Framework-u

Na inštaláciu frameworku Laravel sme použili nástroj pre správu PHP balíkov *Composer*. Pomocou tohto nástroja sme nainštalovali balík *laravel/installer* [2], následne sme použitím príkazu *laravel new projekt* vytvorili priečinok so základnou inštaláciou frameworku.

4.3 Inštalácia a Konfigurácia Databázy neo4j

Aby sme mohli nainštalovať databázu Neo4j muíme si najprv do systému pridať repozitár Neotechnology, potom je nám k dispozícii na inštaláciu balík neo4j. Po inštalácii je nám keď dostupné administráčne rozhranie databázy na adrese: *localhost:7474*. Pri prvom prihlásení sme vyzvaní na zmenu hesla.

Keďže základná inštalácia frameworku Larave lneobsahuje ovládač pre databázu neo4j museli sme použiť ovládač integrovaný v balíku NeoEloquent, to sa registráciou poskytovateľa služby. Po zaregistrovaní služby NeoEloquentServiceProvider sa automaticky zaregistruje ovládač pre databázu a pridajú sa nové možnosti pre konfiguráciu databázy. Následne stačí vykonať štandardnú konfiguráciu mena hostiteľa, port a prístupových údajov.

Na získanie zviazaného prístupu k administráčnemu rozhraniu databázy bez otvorenia portu 7474 verejnosti využívame SSH tunel. Administráčne rozhranie používa okrem portu 7474 ešte port 7687 lebo na komunikáciu s databázou využíva technológiu Web-Socket.

Algoritmus 1 Ukážka algoritmu

```
1      $app->register('Vinelab\NeoEloquent\NeoEloquentServiceProvider');
2      ...
3      'default' => env('DB_CONNECTION', 'neo4j'),
4
5      'connections' => [
6          'neo4j' => [
7              'driver' => 'neo4j',
8              'host'   => env('DB_HOST', 'neo4j'),
9              'port'   => env('DB_PORT', 'neo4j'),
10             'username' => env('DB_USERNAME', 'neo4j'),
11             'password' => env('DB_PASSWORD', 'neo4j'),
12         ],
13     ],
```

4.4 OGM

Keďže framework Laravel natívne obsahuje len ovládače pre relačné databázové ovládače a nástroj na objektovo relačné mapovanie Eloquent. Použili sme volne šíriteľný balík NeoEloquent, ktorý obsahuje ovládač pre databázu Neo4J a zároveň rozširuje dá-

tový model o prvky grafovej databázy. NeoEloquent umožňuje manipuláciu s vrcholmi aj hranami v Neo4J. Manipulácia s vrcholmi je rovnaká ako s entitami v relačnej databáze, NeoEloquent umožňuje vytvárať perzistentné objekty, upravovať ich a vyhľadávať v nich. Pri práci s hranami je mierne odlišná. Najprv treba zadať ktorý objekt môže mať aké vzťahy, tieto vzťahy treba unikátne identifikovať ich typom a kardinalitou. Tento vzťah vraciame ako návratovú hodnotu funkcie daného objektu. Vrátený objekt sa správa podobne ako objekt, dá sa vytvárať upravovať a v prípade vyššej kardinality aj vyhľadávať.

V nasledujúcom príklade vidíme implementáciu dvoch rôznych tried. Trieda používateľa dedí od triedy NeoEloquent a teda sa stáva naviazanou na vrchol v našej databáze. Názov tejto entity v databáze je spojením menovného priestoru v ktorom bol vytvorený a názvu triedy, takže v našom prípade `AppUser`. Trieda obsahuje verejné funkcie, ktoré vracajú objekty hrán. Objekt hrán dostávame volaním zdedených funkcií `hasMany`, `hasOne` a `belongsToMany`. Ako prvý argument funkcie berú názov triedy s ktorou vzťah chceme vrátiť, ako druhý argument berie typ hrany, pomocou tohto typu je identifikovaný typ hrany v databáze.

4.5 Autentifikácia

Jednou zo silných stránok Frameworku Laravel je práve autentifikácia. Pre vytvorenie základnej funkcionality registrácie, prihlasovania a obnovenia zabudnutého hesla stačí použiť Artisan - konzolu frameworku príkaz (`php artisan make:auth`), ktorá vytvorí URL cesty, obrazovky, triedu používateľa a treidy obluhujúce túto funkcionality. My sme potrebovali použiť vlastnú triedu používateľa, ktorá dedí od nášho balíka NeoEloquent, na implementáciu autentifikácie stačilo naimplementovať rozhranie `Authenticatable`, pridať do treidy používateľa pole skrytých a verejných atribútov, a použiť charakteristiku `'AuthenticatableTrait'` a autentifikácia fungovala rovnako ako s relačnou databázou vďaka tomu, že NeoEloquent pokrýva všetky funkcie natívneho Eloquent ORM.

4.6 Smerovač

Framework Laravel poskytuje silný nástroj pre tvorenie ciest.

4.7 Vzhľad

Dôležitou požiadavkou kladenou na rozšírenie bolo príjemné používateľské rozhranie. Z tohto dôvodu malo rozšírenie obsahovať modifikovaných vlastností a tlačidlo pre prístup k nastaveniam rozšírenia v jednoduchšej a praktickej forme. Predpokladaný vzhľad je zobrazený na obrázku č. ??.

Algoritmus 2 Ukážka algoritmu

```
1 namespace App;
2
3 class User extends \NeoEloquent implements Authenticatable {
4
5     // Jeden používateľ môže mať v obľúbených viac miest
6     public function follows() {
7         return $this->hasMany('App\Place', 'FOLLOWS');
8     }
9
10    // Jeden používateľ má jedno miesto ako domáce
11    public function home() {
12        return $this->hasOne('App\Place', 'HOME');
13    }
14    ...
15 }
16 ...
17 class Place extends \NeoEloquent {
18     // Inverzný vzťah – jedno miesto má v obľúbených viac používateľov
19     public function followers(){
20         return $this->belongsToMany('App\User', 'FOLLOWS');
21     }
22     ...
23 }
```

Algoritmus 3 Ukážka algoritmu

```
1
2 namespace App;
3
4 use Illuminate\Contracts\Auth\Authenticatable;
5 use Illuminate\Auth\Authenticatable as AuthenticableTrait;
6
7 class User extends \NeoEloquent implements Authenticatable {
8
9     use AuthenticableTrait;
10
11     // pole verejných atribútov
12     protected $fillable = [
13         'name', 'email', 'password', 'tspCache'
14     ];
15
16     // pole skrytých atribútov
17     protected $hidden = [
18         'password', 'remember_token',
19     ];
20     ...
21 }
```

Záver

Cieľom práce bola analýza anonymizačných modulov, identifikačných prvkov prehliadača a vytvorenie anonymizačného modulu pre internetový prehliadač.

Analýzou najpoužívanejších modulov a vlastností prehliadača, ktoré slúžia na identifikáciu používateľa, sme zistili aktuálny stav a funkcionality rozšírení, ktorými je možné anonymizovať prístup na internet. Väčšina týchto rozšírení modifikuje len časť vlastností prehliadača, ktoré sú odosielané na server, alebo úplne blokuje ich odosielanie. Nami vytvorené rozšírenie dokáže modifikovať väčšinu identifikačných prvkov rozšírenia, pričom dodržiava súvislosti medzi vlastnosťami (používateľský agent odosielaný v hlavičke dopytu je totožný s používateľským agentom zisťovaním pomocou JavaScript príkazu, súvislosť medzi šírkou a dĺžkou rozšírenia obrazovky). Dokáže blokovat údaje, ktoré sú posielané v otvorenej podobe na server a obsahujú informácie o identifikačných údajoch prehliadača, ktoré sa nedajú na úrovni rozšírení modifikovať.

Testovanie rozšírenia nám overilo funkčnosť a správnosť implementácie. Rozšírenie dokáže buď vždy, alebo v časových intervaloch modifikovať väčšinu charakteristických prvkov prehliadača odosielaných na server, a tým zvyšuje anonymitu používateľa.

Zoznam použitej literatúry

- [1] AT, S. I. Db-engines ranking - popularity ranking of graph dbms, 2017.
- [2] OTWELL, T. Laravel - the php framework for web artisans, 2017.

Prílohy