UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

DIPLOMOVÁ PRÁCE

2017 Bc. Jan Svatoš

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Prostorová analýza realitního trhu

Jan Svatoš

Diplomová práce

2017

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2015/2016**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Jan Svatoš**
Osobní číslo:	**I14284**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Prostorová analýza realitního trhu**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Prostorové analýzy veřejně dostupných dat ekonomických subjektů v České republice často vychází z dat
Českého statistického úřadu. Práce bude zaměřena na analýzu dat průměrných cen nemovitostí, které zajistí vlastní aplikace pomocí parsování webových stránek realitních společností. Cílem práce je vytvořit sadu časových řad zobrazujících vývoj cen nemovitostí ve zvolených regionech a sídlech ČR. Očekávanými výsledky práce jsou nejen statistické vyhodnocení dostupných dat, ale zejména grafické znázornění vývoje. Jedním z kroků je i vlastní návrh operace geokódování, kdy jsou k prvkům přiřazovány prostorové souřadnice porovnáváním s databází RÚIAN.
V části práce se autor soustředí na popis oblastí České republiky z pohledu analýzy odhadnutých cen nemovitostí pomocí faktorových modelů a vhodných vysvětlujících proměnných (počet obyvatel, průměrná mzda, vzdálenost ke krajskému městu). V případové studii bude provedeno vícerozměrné shlukování zkoumaných oblastí s cílem vytvořit prostorové rozdělení České republiky z pohledu cen nemovitostí.

Rozsah grafických prací: **10**

Rozsah pracovní zprávy: **40**

Forma zpracování diplomové práce: **tištěná**

Seznam odborné literatury:

CIPRA, T. Finanční ekonometrie. Praha: Ekopress, 2008. 538 s. ISBN 978-80-86929-43-9.
FIALA, P. Úvod do ekonometrie. Praha: ČVUT v Praze, 2008. 173 s. ISBN 978-80-01-04004-1.

Vedoucí diplomové práce: **Mgr. Jaroslav Marek, Ph.D.**
Katedra matematiky a fyziky

Datum zadání diplomové práce: **31. října 2015**
Termín odevzdání diplomové práce: **13. května 2016**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2015

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 17. 05. 2017

podpis autora
Jan Svatoš

PODĚKOVÁNÍ

Chtěl bych velice poděkovat svému vedoucímu práce Mgr. Jaroslavu Markovi, PhD. za konzultace, které pro mě byly velkým přínosem. Dále bych chtěl své rodině, přátelům a všem, kteří mi poskytovali psychickou podporu nejen při tvorbě této diplomové práce, ale i při studiu.

ANOTACE

Diplomová práce se věnuje odhadu střední hodnoty ceny bytů v jednotlivých lokalitách ČR, využívá tzv. faktorové modely a metodu nejmenších čtverců. K výpočtu se využívá lineární regresní model, kde je právě metoda nejmenších čtverců využívána. Dále tato práce obsahuje popis operace geokódování a zobrazení časových řad vývoje střední hodnoty ceny bytů v okrese. Naleznout lze popis lineárního regresního modelu, časových řad, shlukové analýzy, které aplikace využívá. Nechybí popis zvolené technologie ASP .NET MVC včetně použitých knihoven a nástrojů použitých v aplikaci a jejich konkrétního využití v aplikaci.

KLÍČOVÁ SLOVA

geokódování, odhad ceny, časová řada, shluková analýza, Google Charts, Google Distance Matrix

TITLE

Spatial analysis of real estate market

ANNOTATION

This thesis is dedicated to the estimation of mean price value using factor models and the least squares method. The calculation is performed using a linear regression model, wherein the estimated value is calculated using the least squares method. The thesis also includes a description of the geocoding operation and a display of time series of mean price value according to each district. Description of linear regression, time series and cluster analysis used by the application can also be found within the thesis. Furthermore, the programming technique of ASP .NET MVC, including the C# programming libraries and tools used in the application, are also described.

KEYWORDS

geocoding, least square method of uknown parametres, time series, Google Charts, Google Distance Matrix

Obsah

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

ČR	Česká republika
XML	eXtensible Markup Language
HTML	HyperText Markup Language
JSON	Java Script Object Notation
MSSQL	International Organization for Standardisation
LINQ	Language Integrated Query
MNČ	Metoda nejmenších čtverců
MVC	Model View Controller
URL	Uniform Resource Locator
WMS	Web Map Service
WMTS	Web Map Tile Service

0 ÚVOD

Cílem této diplomové práce bylo vytvoření aplikace, která uživatelům umožní získat informace o tom, jaká je aktuální průměrná cena za $1m^2$ u inzerátů v jednotlivých lokalitách (kraje, okresy, obce). Tyto údaje poté aplikace zakresluje v souladu s územním členěním České republiky do mapy, aby uživatel měl vizuální porovnání se sousedními regiony (např. porovnání mezi kraji). Dále pak aplikace uživateli zobrazuje historický vývoj ceny ve sledovaném období. Dalším cílem bylo zobrazit podobnost vývoje cen mezi okresy České republiky. Zpracovávaná data byla načítána z realitního serveru Sreality.cz.

Vyvinutá aplikace pracuje s vlastní navrženou databází, ve které jsou ukládány potřebné informace načítané ze serveru Sreality.cz. Data jsou do databáze načítána pomocí dotazů na API rozhraní tohoto serveru a následně vracena v podobě JSON objektů. Aplikace ukládá i další potřebné informace pro výpočet odhadu ceny, buď dopočítané samotnou aplikací a použitými nástroji od společnosti Google, anebo načtené z různých serverů například z webových stránek českého statistického úřadu. Aplikace tedy vytváří pro uživatele statistický přehled o vývoji cen a jejich odhadech v jednotlivých obcích.

1 ZÁKLADNÍ NÁSTROJE PRO ANALÝZU REALITNÍHO TRHU

1.1 Lineární regresní model

K nejdůležitějším ekonometrickým nástrojům patří regresní analýza. Tato analýza slouží ke kvantitativnímu popisu vztahu mezi ekonomickými faktory a finančními veličinami (lze je označit jako proměnné). Využití lineární regresní funkce vysvětluje hodnoty jedné vysvětlované proměnné pomocí vysvětlujících proměnných. Tato vysvětlovaná proměnná se zpravidla označuje jako ***y*** a vysvětlující proměnné se označují jako $x_0, x_1, \ldots \ldots, x_k$. Uvažujme lineární regresní model ve tvaru:

$$y_t = \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + \ldots + \beta_k x_{tk} + \varepsilon_t, \quad t = 1, \ldots n, \qquad (1)$$

kde složkami ***t*** se označuje časový index. Proměnná označená jako y_t je hodnota vysvětlované proměnné y pozorované v čase t. Proměnné označované jako $x_{t1}, \ldots, x_{tk}$ představují hodnoty vysvětlujících proměnných $x_1, \ldots, x_k$ pozorovaných v čase t (proměnnou x_{t0} lze označit jako speciální proměnou, jejíž hodnota je ve všech pozorováních rovna jedné).

Parametry $\beta_0, \beta_1, \ldots, \beta_k$ se označují jako neznámé regresní parametry lineárního regresního modelu, kde parametr β_0 je absolutní člen, parametr ε_t je reziduální složka modelu (náhodná složka). V reziduální složce ε_t je obsažen souhrn vlivů, které nejsou uvedeny v modelu a také tato složka zahrnuje chyby měření a nekorektní volbu regresního vztahu anebo jevy náhodného charakteru. Proto je potřeba vyjádření lineárního regresního modelu vhodně modifikovat. Pro náhodnou složku platí, že $E(\varepsilon_t) = 0, t = 1, \ldots, n$, pak lze očekávaná hodnota y_t lze zapsat tímto vztahem:

$$E(y_t) = \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + \ldots + \beta_k x_{tk} + \varepsilon_t, \quad t = 1, \ldots n. \qquad (2)$$

Tento modifikovaný vztah se nazývá regresní funkce. Koeficienty $\beta_2, \beta_3, \ldots, \beta_k$, se nazývají regresní koeficienty a měří změnu hodnoty $E(y_t)$, která odpovídá jednotkové změně libovolné jedné vysvětlující proměnné. Ostatní vysvětlující proměnné jsou neměnné.

Protože koeficienty regresní rovnice a parametry rozdělení náhodné složky v základním souboru není znám, musí postačit pouze s odhady získaných z výběrových dat. K dispozici je zpravidla jeden konečný výběr *n* pozorování, kde každé z nich obsahuje konkrétní hodnotu vysvětlované proměnné **Y** a množinu *k* pozorovaných hodnot

vysvětlujících proměnných $X_1, X_2, \ldots, X_k$. Aplikováním jednoho z adekvátního postupu lze odhadnout z výběru n pozorování deterministickou regresní rovnici pomocí výběrové regresní funkce:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_k X_k, \tag{3}$$

kde $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k$ představují bodové odhady neznámých parametrů $\beta_1, \beta_2, \ldots, \beta_k$. Proměnná $\hat{Y}$ je vyrovnaná nebo modelem predikovaná hodnota Y a její teoretické hodnoty pro jednotlivá pozorování lze psát tímto vztahem:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k, \quad t = 1, \ldots n. \tag{4}$$

Rozdíl mezi skutečnými hodnotami Y_i a vyrovnanými hodnotami $\hat{Y}_i$ v jednom výběru lze zapsat jako:

$$Y_i - \hat{Y} = \varepsilon_i, \quad t = 1, \ldots n, \tag{5}$$

a nazývá se tzv. reziduum. Měřitelná rezidua ε_i se mohou chápat jako odhady neznámých náhodných složek u_i. Vzhledem ke vztahu platí, že

$$Y_i = \hat{Y} + \varepsilon_i = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_k X_k + \varepsilon_i, \quad t = 1, \ldots n. \tag{6}$$

Zápis tohoto vztahu lze napsat přehledněji a to do maticového zápisu lineární regresního modelu

$$\boldsymbol{Y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}, \tag{7}$$

nebo

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_{21} & \ldots & X_{k1} \\ 1 & X_{22} & \ldots & X_{k2} \\ \vdots & \vdots & \ldots & \vdots \\ 1 & X_{2n} & \ldots & X_{kn} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}, \tag{8}$$

kde

- $\boldsymbol{Y}$ je sloupcový vektor n pozorování stochastických hodnot vysvětlované proměnné,
- $\boldsymbol{X}$ – matice $n \times k$ pozorovaných fixních hodnot vysvětlujících proměnných,
- $\boldsymbol{\varepsilon}$ – sloupcový vektor n hodnot nepozorovatelné náhodné složky,
- $\boldsymbol{\beta}$ – sloupcový vektor k neznámých parametrů.

Rozdílem počtu pozorování n a počtu odhadovaných parametrů k je počet stupňů volnosti, přičemž musí platit podmínka $n > k$.

Zdroj: (2)

1.1.1 Regresní analýza

Regresní analýza patří v ekonometrii mezi často užívané nástroje. Využívá se zejména v případech, kdy je k dispozici model pro popis vysvětlované proměnné pomocí vhodných vysvětlujících proměnných. Při výpočtu odhadů neznámých parametrů známého modelu se obvykle používá metoda nejmenších čtverců (MNČ). K dispozici jsou testy umožňující ověřit, zda je model vhodný a v datech signifikantně přítomný.

Zdroj: (1)

1.1.2 Metoda nejmenších čtverců

Pokud je regresní model lineární, tj. neznámé parametry se v modelu vyskytují ve tvaru $\beta \cdot x_i$, můžeme model zapsat ve tvaru

$$\boldsymbol{Y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}. \tag{9}$$

V takovém případě je metoda nejmenších čtverců založena na minimalizaci součtu čtverců:

$$S_e = \sum_{t=1}^{n}\left(\boldsymbol{Y}_t - (\beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + \ldots + \beta_k x_{tk})\right)^2 = \sum_{t=1}^{n}(\boldsymbol{Y}_t - \boldsymbol{X}_t\beta)^2 = (\boldsymbol{Y} - \boldsymbol{X\beta})'(\boldsymbol{Y} - \boldsymbol{X\beta}). \tag{10}$$

Funkcionál (10) minimalizuje součet druhých mocnin vertikálních vzdáleností hodnot vysvětlované proměnné $\boldsymbol{Y}_t$ od regresní nadroviny. Tuto nadrovinu představuje v nejjednodušších případech regresní přímka nebo regresní rovina. Cílem je snaha co nejlepší proložení nadroviny množinou pozorovaných bodů. Řešení má v maticovém zápisu tvar

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y}. \tag{11}$$

Tento odhad $\widehat{\boldsymbol{\beta}}$ parametrů $\boldsymbol{\beta}$ je tzv. bodový odhad získaný metodou nejmenších čtverců, který označujeme jako MNČ-odhad. Můžeme v této souvislosti zavést další dva pojmy:

a) výběrová regresní funkce s MNČ – hodnotami $\hat{y} = \boldsymbol{X\widehat{\beta}}$, respektive

$$\hat{y}_t = \hat{\beta}_0 + \hat{\beta}_1 x_{t1} + \hat{\beta}_2 x_{t2} + \ldots + \hat{\beta}_k x_{tk} \text{ pro } t = 1, \ldots, n, \tag{12}$$

jsou tzv. vyrovnané hodnoty $\boldsymbol{y}_t$;

b) MNČ – rezidua

$$\hat{\varepsilon} = y - \hat{y} = y - \boldsymbol{X}\widehat{\boldsymbol{\beta}}. \tag{13}$$

Zdroj: (2)

1.1.3 Index determinace

Při aplikaci MNČ platí tento vztah

$$S_y^2 = S_{\hat{y}}^2 + S_e^2, \tag{14}$$

kde

$$S_y^2 = \sum_{i=1}^{n} \left(\hat{Y}_i - \bar{Y}\right)^2 \tag{15}$$

je vztah pro celkový součet čtverců,

$$S_y^2 = \sum_{i=1}^{n} \left(\hat{Y}_i - \bar{Y}\right)^2, \tag{16}$$

označme za součet čtverců modelu a výrazem

$$S_e^2 = \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2, \tag{17}$$

vyznačujeme reziduální součet čtverců. Ve výpočtu součtu čtverců bychom měli průměr z pozorovaných hodnot nahradit průměrem odhadnutých hodnot, dále při aplikaci metody nejmenších čtverců odvodíme, že tyto průměry jsou si rovny. Lze je tedy zapsat tímto vztahem

$$\bar{Y} = \bar{\hat{Y}}. \tag{18}$$

Lze tedy tímto vypozorovat jakousi kvalitu regresního modelu. To představuje, že čím bude model „lepší“, bude mít tedy vetší součet čtverců modelu a také bude menší reziduální součet čtverců. Za špatný model můžeme považovat takový model, který bude mít velkou hodnotu součtu reziduálních čtverců oproti hodnotě součtu čtverců modelu. Následně lze tedy celou rovnost vydělit celkovým součtem čtverců a poté dostaneme tento tvar

$$I = \frac{S_{\hat{y}}^2}{S_y^2} + \frac{S_e^2}{S_y^2}. \tag{19}$$

V tomto tvaru vidíme dva zlomky, které nabývají kladných hodnot a jejich součet je roven jedné. Tudíž hodnoty obou zlomků musí být v intervalu od nuly do jedné. Jestliže model bude dobře vystihovat závislost vysvětlované proměnné na pravé straně rovnice, hodnota

prvního zlomku poroste a bude se blížit k hodnotě jedna, hodnota druhého zlomku bude klesat a bude se blížit hodnotě nula. Jakmile bude model popisovat závislost vysvětlované proměnné špatně, budou hodnoty prvního a druhého zlomku přesně naopak. Pokud vezmeme první zlomek tohoto vztahu a označíme ho jako „kritérium kvality“ regresního modelu, pak jej můžeme vyjádřit tímto vztahem

$$I^2 = \frac{S_{\hat{y}}^2}{S_y^2}, \tag{20}$$

a můžeme ho nazvat jako **index determinace**.

Tento index determinace představuje kvalitu regresního modelu, čili označuje procentuální hodnotu rozptylu vysvětlované proměnné, která je vysvětlena modelem a také procentuální hodnotu proměnných, které jsou nevysvětleny. Z předchozího popisu tohoto indexu determinace může být zřejmá, že jeho hodnota se pohybuje v intervalu od nuly do jedné, kde hodnoty blízké hodnotě nula značí špatnou kvalitu regresního modelu a hodnoty blízké jedničce značí dobrou kvalitu regresního modelu.

Nevýhodou indexu determinace je závislost na počtu vysvětlujících proměnných. Roste-li jejich počet, roste také i jeho hodnota. Proto pro výpočet se spíše používá tzv. modifikovaného indexu determinace, který je zbaven nadbytečného počtu vysvětlujících proměnných a má tento tvar

$$I_M^2 = I^2 - \frac{(I - I^2)(k - I)}{n - k}, \tag{21}$$

kde k představuje počet odhadovaných parametrů modelu. Hodnota tohoto modifikovaného indexu determinace je nepatrné menší než hodnota indexu, který nebyl modifikován.

Zdroj: (3)

1.2 Geoinformatické nástroje

Geografické nástroje mají uplatnění v různých profesních sférách, nejen přímo souvisejících se samotnou geografií. Dají se pomocí nich vytvářet různá studie, analýzy a modely týkajících se určitého území a vytvořit jejich vizualizaci. Hlavní myšlenkou geoinformatických nástrojů je možnost propojení prezentačního média například

s databází či jinými zdroji dat, obsahujících polohopopisné či popisné charakteristiky objektů, případně popsat vztahy mezi nimi. K těmto nástrojů se řadí tzv. geografické informační systémy (GIS), které výše pospaných charakteristik využívají.

Nejprve je potřeba si charakterizovat co to takový geografický informační nástroj je. Geografický informační systém umožňuje sběr prostorových dat (geodat), nástroje pro jejich analýzu a nástroje pro vizuální prezentaci modelů nad vybraným územím. Načtená vstupní data tento systém ukládá do tzv. geodatabáze a tato data jsou pro aplikaci zpřístupněna pomocí dotazů nad touto databází, anebo prováděním složitých analýz se vzájemnou spoluprací s nějakým softwarem GIS. Po zpracování dat vrací nějaké výstupy a ty mohou být v například v podobě klasických, nebo digitálních map, v zobrazení trojrozměrného modelu určitého území, anebo v podobě dynamické mapy. Data, která se mají zobrazit, bývají často spravována pomocí mapových serverů pro publikaci mapových služeb (například WMS, WMTS) a dá se pomocí nich vyvíjet moderní webové aplikace.

Jak již bylo zmíněno, tyto systémy mají uplatnění v mnoha sférách například v organizacích státní správy a samosprávy, ale i v soukromém sektoru. Lze je například využít k evidenci majetku, parcel, nemovitostí, k územnímu rozhodování a regionálnímu rozvoji (pro tvorbu územních plánů nebo koncepcí strategického rozvoje), k řízení energetických a vodohospodářských soustav, ke kartografii a mnoha dalších.

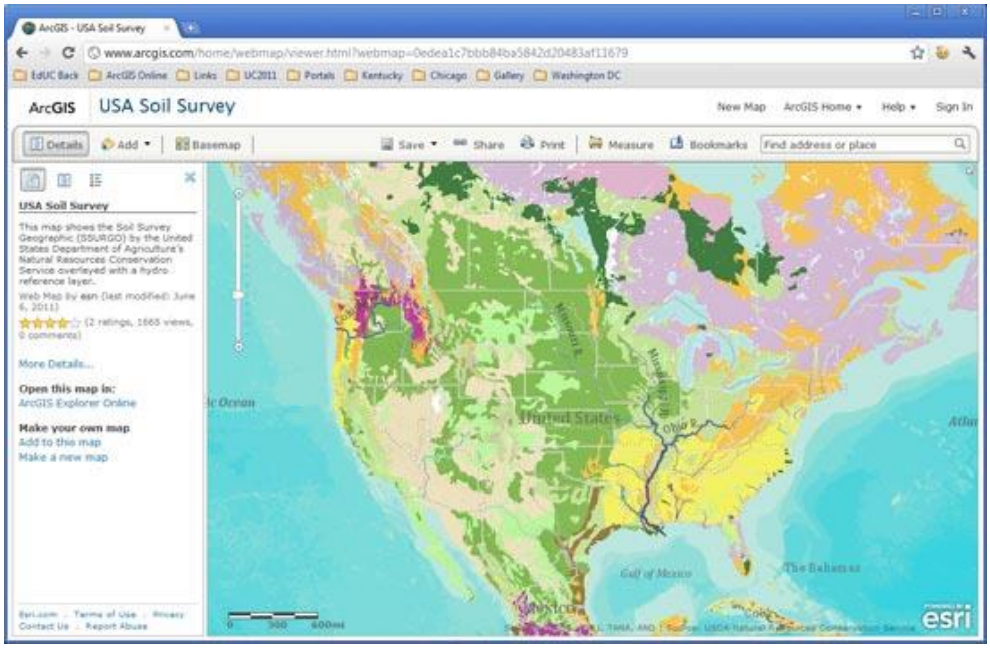
Zdroj: (5)

1.2.1 ArcGIS

Pojmem ArcGIS je skupina názvů SW produktů od společnosti ESRI a slouží k tvorbě geografických informačních systémů. Tento software je postaven na architektuře modulů, to znamená, že díky tomu lze vytvořit řešení vyhovující potřebám od uživatelů, pracovních skupin, až po komplexní podnikové informační systémy. Skupina softwarů ArcGIS v sobě zahrnuje několik základních produktů ArcGIS for Desktop, ArcGIS for Server, ArcGIS Mobile a další vývojové nástroje.

ArcGis for Desktop je aplikace vyvinutá pro načítání, zpracování, vyhledávání a prezentaci geografických informací. Dodávány jsou ve třech licencích a to Basic, Standart a Advaced. Basic v sobě zahrnuje SW ArcView, Standart zahrnuje ArcEditor a Advanced verze v sobě zahrnuje ArcInfo. Ke všem těmto aplikacím lze využít nástroje ke stažení zdarma a to ArcExplorer a ArcGlobe. **ArcGIS for Server** je software vyvinutý

pro tvorbu podnikových informačních systémů a internetových řešení. Tento software podporuje SOA (servisně orientovaná architektura) pro informační systém, platformy J2EE, Microsoft .NET, kompatibilitu s webovými službami, systémy na bázi řízení dat a podpory mnoha dalších aplikací. **ArcGIS Mobile** je software vyvinutý zejména pro mobilní platformy a umožňuje tak využívat geografické informace v terénu. Další vývojové nástroje balíků SW ArcGIS jsou komponenty a programové rozhraní pro vývoj aplikací desktopových, internetových a mapových serverů, případně k vytváření webových služeb.



Obrázek 1.1 - Ukázka aplikace ArcGIS online, zdroj: www.esri.com

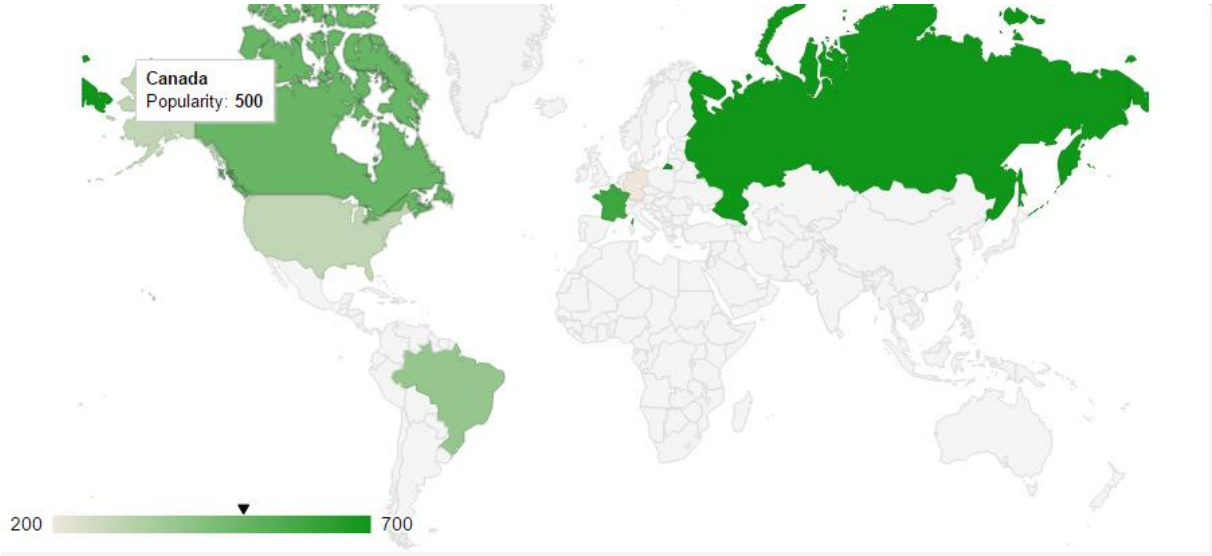
Zdroj: (6)

1.2.2 Google geochart

Jedním z odlehčených geoinformatických nástrojů je Google geochart. Jedná se o graf ze sady nástrojů Google charts, umožňující vykreslit mapu státu, kontinentu nebo regionu s oblastmi, které jsou identifikovány třemi způsoby a to:

a) **region** vybarvující celý region, jako jsou např. státy a provincie (kraje),
b) **markers** používající kruhového vybarvení regionů, které mají různý průměr dle zadané hodnoty,
c) **text** zobrazující značky nad regiony, které je identifikují (např. Rusko nebo Asie)

Pomocí nastavení **region chart** se vyplňují celé regiony (typicky země) barvami odpovídající hodnotám, které jim uživatel přiřadí. Příklad jak takový graf vypadá je vidět na obrázku Obrázek 1.2.



Obrázek 1.2 - Příklad vykreslení Region geochart, zdroj: vlastní

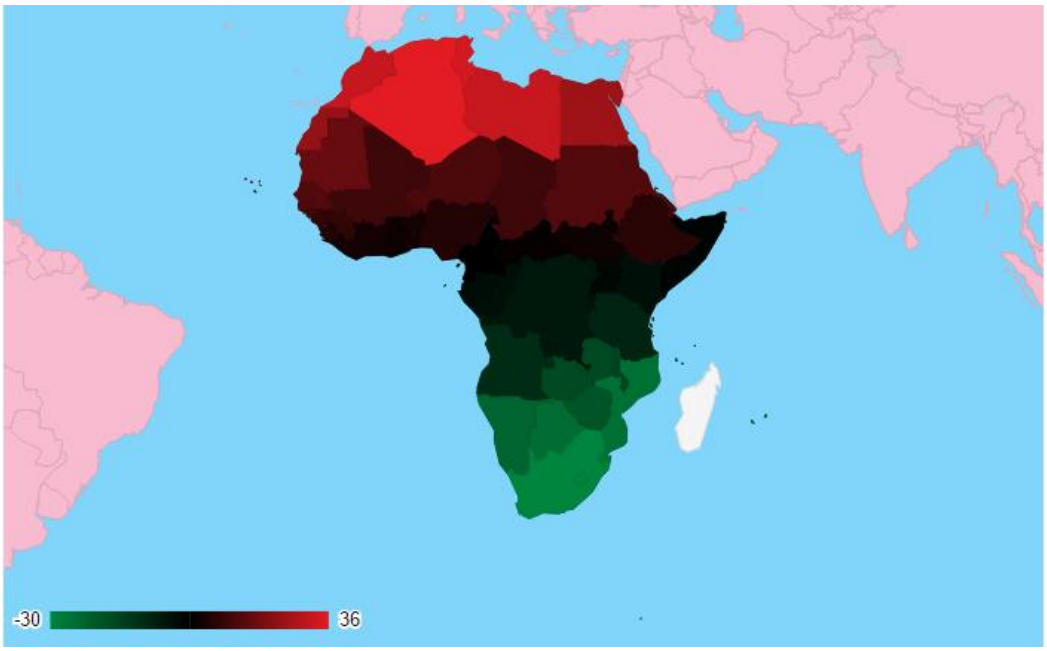
Pro vybarvení jednotlivých regionů, je zde několik nastavení pro uživatelské nastavení a přizpůsobení:

a) **colorAxis** – nastavuje spektrum barev pro obarvení jednotlivých regionů z tabulky dat,
b) **backgroundColor** – nastavuje barvu pozadí grafu,
c) **datalessRegionColor** – přiřazuje barvu regionu, k němuž nejsou přiřazena žádná data,
d) **defaultColor** – přiřazuje barvu regionu s tabulky dat, jehož hodnota je NULL nebo není specifikována.

ColorAxis je důležité nastavení, specifikující rozsah barev pro datové hodnoty. Z pravidla se nastavuje jako pole, kde první prvek pole představuje barvu nejmenší hodnoty z tabulky dat, a kde poslední prvek pole představuje barvu největší hodnoty z tabulky dat. Lze tedy nastavit více než dvě barvy a poté při vykreslení vznikne vzájemná interpolace mezi nastavenými barvami.

Na příkladu nastavení grafu jsou použity všechna vyjmenovaná nastavení. **ColorAxis** je zde využit pro zobrazení kontinentu Afrika s panafrickými barvami, využívající zeměpisných souřadnic zemí. Takže jsou nastaveny barvy červená pro země ležící na severu, černá pro země ležící uprostřed a zelená pro země ležící na jihu afrického

kontinentu. Nastavení **backgroundColor** určuje barvu pozadí kontinentu a tato barva je světle modrá, dále **datalessRegionColor** obarvuje země neležící na africkém kontinentu na světle růžovou a **defaultColor** pro nespecifikovanou hodnotu u ostrovu Madagaskar. Výsledek tohoto vykreslení je vidět na obrázku



Obrázek 1.3 - Vybarvený RegionChart dle vlastního nastavení barev, zdroj: (20)

Tento graf od společnosti Google má ještě mnoho dalších nastavení. Tato nastavení jsou skvěle popsána na stránkách *https://developers.google.com/chart/*, kde je možnost si je podrobněji nastudovat, či vyzkoušet jejich zobrazit pomocí serveru *https://jsfiddle.net/* všechna zvolená nastavení, bez toho aniž by bylo potřeba vytvořit webovou aplikaci. Anebo případně pomocí tohoto serveru provézt nastavení a výsledek pak vložit na vytvořenou stránku webové aplikace.

Zdroj: (7)

2 NÁVRH APLIKACE, METODIKA, ANALYZOVANÁ DATA, ZÍSKANÉ VÝSLEDKY

2.1 Návrh aplikace

2.1.1 Návrh databáze

Protože aplikace má zobrazovat časové řady a počítat odhady cen ve sledovaném období, bylo třeba vytvořit návrh databázové aplikace umožňující uchování potřebných dat. Nejprve bylo potřeba zjistit, jaká data se mají v databázi uchovávat. Jak již bylo zmíněno, data jsou do databáze načítána ze serveru Sreality.cz, tudíž bylo potřeba vytvořit si základní tabulky odpovídající zhruba typům nemovitostí nacházejících se na tomto serveru.

Pro každý typ nemovitosti bylo potřeba vytvořit samostatnou tabulku obsahující potřebné informace pro další statistické zpracování. To znamená, že tyto tabulky slouží pro uchování jednotlivých inzerátů ze serveru Sreality.cz Aby se dal vypočítat odhad pro vybrané obce ČR, bylo potřeba mít uloženy všechny obce České republiky a jejich základní informace, jako je například počet obyvatel a k těmto obcím uchovávat vypočítané průměry a odhady sledovaných hodnot.

Aby aplikace umožňovala uchovávat data dle územního členění České republiky, bylo potřeba uchovávat základní informace o jednotlivých krajích, okresech a obcích.

2.2 Analýza dat

2.2.1 Získání inzerátů ze serveru

V této podkapitole získání dat je popsáno jaká data byla potřeba ukládat a v jaké podobě jsou získána. Jak již bylo zmíněno, inzeráty jsou načítány ze serveru Sreality.cz. V bakalářské práci Monitorování realitního trhu byla data načítána pomocí parsování webových stránek. Parsováním stránek se získávaly informace z HTML tagů a následně se zobrazovaly. Nevýhodou však této varianty je, pokud se změní vzhled webové aplikace, odkud jsou data získána, musí se předělat celé parsování podle změněných HTML tagů.

Proto bylo potřeba najít alternativní variantu získání dat. Po dlouhém hledání na internetovém vyhledávači Google byla na této webové stránce zjištěna URL adresa, kde jsou data vracena ze serveru Sreality.cz. Tato data jsou ze serveru vracena ve formátu

JSON a URL adresa dotazující se na serve využívající API rozhraní serveru Sreality má následující tvar.

Adresa API rozhraní http://www.sreality.cz/api/cs/v1/estates?category_main_cb=2& category_type_cb=1&locality_region_id=12&per_page=20. Bylo tedy potřeba zjistit, co jednotlivé parametry nastavují a jaký výsledek poté vrací API rozhraní.

Z testování různých hodnot parametrů a výstupu dat vracených ze serveru, byly tyto parametry rozklíčovány a následně jsou zde popsány.

Parametr ***category_main_cb=2*** představuje dle nastaveného čísla, o jaký typ nemovitosti se jedná. Lze nastavit hodnotu od 1 do 5 a podle toho, jaké je nastaveno číslo, budou vráceny inzeráty určitého typu nemovitosti, kde číslo 1 představuje byty, 2 představuje domy, číslo 3 představuje pozemky, číslo 4 komerční nemovitosti a číslo 5 představuje ostatní nemovitosti.

Druhý parametr ***category_type_cb=1*** určuje druh nabídky nemovitosti. Nastavuje se mu číselná hodnota od čísla 1 do čísla 3, kde číslo 1 představuje prodej, číslo 2 pronájem a číslo 3 dražbu.

Třetí parametr ***locality_region_id=12*** nastavuje, pro jaký okres se mají data z API vrátit. Opět se nastavuje číselnou hodnotou a číslo představuje jednotlivé okresy ČR. Hodnoty jsou od 1 až po 77, pro jednotlivé okresy, avšak pro části Prahy se zde nastavují jiné číselné hodnoty a to od 5001 po 5010. Tyto číselné hodnoty parametrů neodpovídají žádným předurčeným hodnotám (například ze statistického úřadu, kde každý okres a kraj mají přiřazený svůj kód), ale jedná se zřejmě o interní identifikátory jednotlivých okresů na serveru Sreality.cz

Poslední čtvrtý parametr ***per_page=20*** nastavuje počet vracených inzerátů na jednu stránku. Jak jsou tato data zpracována a ukládána, je popsáno v kapitole *Načítání inzerátů v aplikaci*, kde je podrobněji rozepsáno, jak se postupovalo při načítání a získání potřebných dat pro navrženou databázi.

2.2.2 Získání ostatních dat

V kapitole **Chyba! Nenalezen zdroj odkazů. Chyba! Nenalezen zdroj odkazů.**, byly určeny 3 parametry pro výpočet odhadu ceny. První parametr byl počet obyvatel obce, druhý vzdálenost ke krajskému (okresnímu) městu a třetí nezaměstnanost v okrese. Proto byly potřeba

získat základní informace o obcích České republiky. Nejlepší informace o obcích lze nalézt na stránkách českého statistického úřadu, který ty informace zpracovává a uchovává. Základní informace o všech obcích České republiky se nachází v tzv. malém lexikonu obcí. Dá se z tohoto serveru stáhnout buď v podobě souboru xlsx, anebo v souboru pdf. Obsahuje informace o všech více jak 6000 obcích ČR, jako jsou například kód obce, kód okresu, ve kterém se obec nachází, počet obyvatel a mnoho dalších informací. Ukázka jak takový soubor vypadá je vidět na obrázku Obrázek 2.1.

Tab. 101. Obce České republiky 2016

Kód okresu	Pořadové číslo	Název obce	Kód obce	Počet částí obce	Počet katastrů	Katastr. výměra v ha	Počet obyvatel			Správní obvod obce s rozšířenou působností	Matriční úřad	Pošta	Škola	Zdrav. zař.	Typ obce
							celkem	ve věku 0 - 14 let	ve věku 65 a více let			(ano = 1, ne = 0, *=škola s jedním stupněm)			
	a	b	c	1	2	3	4	5	6	7	8	9	10	11	12
CZ010	1	PRAHA	554782	112	112	49 616	1 267 449	188 832	233 685		PRAHA	1	1	1	hlavní město
CZ0201	1	Benešov	529303	15	2	4 687	16 555	2 603	2 984	Benešov	Benešov	1	1	1	město
CZ0201	2	Bernartice	532568	2	2	1 011	221	26	55	Vlašim	Dolní Kralovice	1	0	0	obec
CZ0201	3	Bílkovice	530743	3	1	578	209	31	43	Vlašim	Vlašim	0	0	0	obec
CZ0201	4	Blažejovice	532380	2	2	463	113	14	26	Vlašim	Dolní Kralovice	0	0	0	obec
CZ0201	5	Borovnice	532096	1	1	310	80	6	27	Vlašim	Čechtice	0	0	0	obec
CZ0201	6	Bukovany	532924	1	1	740	762	132	115	Benešov	Týnec nad Sázavou	0	0	0	obec
CZ0201	7	Bystřice	529451	26	10	6 336	4 334	740	732	Benešov	Bystřice	1	1	1	město
CZ0201	8	Ctiboř	532690	2	1	392	128	22	23	Vlašim	Vlašim	0	0	0	obec
CZ0201	9	Čakov	529478	3	3	529	131	31	19	Benešov	Benešov	0	0	0	obec
CZ0201	*10*	*Čechtice*	*529486*	*14*	*5*	*3 941*	*1 370*	*205*	*271*	*Vlašim*	*Čechtice*	*1*	*1*	*1*	*městys*
CZ0201	11	Čerčany	529516	2	1	645	2 774	476	514	Benešov	Čerčany	1	1	1	obec
CZ0201	12	Červený Újezd	529532	6	2	1 259	318	55	64	Votice	Votice	1	0	0	obec
CZ0201	*13*	*Český Šternberk*	*529541*	*1*	*1*	*547*	*158*	*28*	*35*	*Benešov*	*Divišov*	*0*	*0*	*0*	*městys*

Obrázek 2.1 - Ukázka souboru Malý lexikon obcí, zdroj: ČSÚ

V návrhu databáze je popsáno, jaké sloupce ze souboru xlsx jsou z tohoto souboru načítány, protože všechny informace pro výpočet odhadu ceny, nebo pro zobrazení detailu obce v aplikaci nebyly potřeba.

Pro výpočet odhadu ceny bylo potřeba ještě zjistit nezaměstnanost v jednotlivých okresech. Tato data jsou volně přístupná na *webových stránkách Ministerstva práce a sociálních věcí* a jsou v souboru uloženy míry nezaměstnaností pro jednotlivé okresy České republiky za předchozí měsíc.

2.3 Metodika

Podkapitola metodika v sobě zahrnuje jednotlivé použité metody, které jsou použity v aplikaci. Jsou zde ukázány příklady výpočtu jednotlivých metod. Lze zde najít příklad výpočtu lineárního regresního modelu, sestrojení grafu pro zobrazení vývoje průměrné ceny bytu za 1 m^2. Dále pak je zde vysvětlena metoda, počítající podobnost vývoje průměrné ceny bytu za 1 m^2 a jejich zobrazení.

2.3.1 Příklad výpočtu regresního modelu

Po úvodu do teorie regresního modelu a indexu determinace si v této podkapitole ukážeme příklad, jak lze tento model použít. Data pro tento model najdeme v tabulce

Tabulka 2-1, která pochází z bakalářské práce Monitorování realitního trhu. Tato tabulka obsahuje dvacet obcí pardubického kraje, u kterých byly zjištěny tři faktory: počet obyvatel, vzdálenost ke krajskému městu a nezaměstnanost v okrese. V aplikaci se výpočet odhadu průměrné ceny aplikuje na jednotlivé okresy. Další parametr, který je důležitý pro výpočet, je průměrná cena za m^2 v obci.

Cílem využití tohoto modelu je, že se snažíme získat odhady parametrů, ze kterých lze vypočítat odhad průměrné ceny za m^2 v obci, která má jeden, anebo žádný inzerát. Takže pokud existuje takováto obec, není možnost porovnání průměrné ceny s žádným jiným inzerátem.

	Obec	Počet obyvatel	Vzdálenost od kr. města [h]	Nezaměstnanost v okr. [%]	Průměrná cena za m^2 [Kč]
1	Česká Třebová	15892	1,05	6,95	13760,61
2	Heřmanův Městec	4830	0,32	8,39	14350,00
3	Hlinsko	9972	0,70	8,39	12934,08
4	Holice	6489	0,32	6,13	19295,40
5	Choceň	8840	0,68	6,95	16490,08
6	Chrudim	23182	0,25	8,39	31492,88
7	Chvaletice	3074	0,52	6,13	14530,38
8	Lázně Bohdaneč	3355	0,23	6,13	28533,09
9	Letohrad	6351	1,12	6,95	20804,59
10	Litomyšl	10178	0,83	9,29	21567,70
11	Moravská Třebová	10491	1,23	9,29	12592,10
12	Opatovice nad Labem	2490	0,25	6,13	21694,45
13	Pardubice	89467	0,00	6,13	25907,27
14	Polička	8903	1,06	9,29	13812,00
15	Přelouč	9057	0,36	6,13	19817,08
16	Skuteč	5204	0,62	8,39	13081,58
17	Králíky	4444	1,42	6,95	10957,88
18	Svitavy	17027	1,05	9,29	14423,00
19	Ústí nad Orlicí	14472	1,00	6,95	15485,56
20	Vysoké Mýto	12429	1,46	6,95	18926,94

Tabulka 2-1 - Data pro sestavení lineárního modelu, zdroj: (4)

K výpočtu odhadu parametrů využijeme metodu nejmenších čtverců (MNČ) a konkrétně vztah

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y}, \qquad (22)$$

u kterého si nejprve musíme určit vysvětlující proměnné X_1, X_2, X_3. Parametrem x_1 označme v našem případě počet obyvatel, parametrem x_2 vzdálenost od krajského města a parametrem x_3 nezaměstnanost v okrese. Máme stanoveny parametry X_1, X_2, X_3 a nyní si můžeme sestavit matici **X,** která bude obsahovat dvacet řádků (počet obcí v kraji) a čtyři sloupce. První sloupec bude obsahovat samé jedničky, druhý sloupec parametry

X_1(počet obyvatel), třetí sloupec parametry X_2(vzdálenost od krajského města) a čtvrtý bude obsahovat parametry X_3 (nezaměstnanost v okrese).

Dostaneme tedy matici **X,** která bude mít následující tvar

$$\boldsymbol{X} = \begin{pmatrix} 1 & 15892 & 1{,}05 & 6{,}95 \\ 1 & 4830 & 0{,}32 & 8{,}39 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 12429 & 1{,}46 & 6{,}95 \end{pmatrix}. \tag{23}$$

Abychom mohli uskutečnit výpočet regresního modelu, je potřeba si ještě určit sloupcový vektor **Y.** Tento vektor bude mít dvacet řádků a bude obsahovat hodnoty představující průměrné ceny za m^2 v jednotlivých obcích. Jeho tvar bude vypadat takto

$$\boldsymbol{Y} = \begin{pmatrix} 13760{,}61 \\ 14350 \\ \vdots \\ 18926{,}94 \end{pmatrix}. \tag{24}$$

Dále se ve vztahu pro výpočet pomocí metody objevovala matice $\boldsymbol{X'}$, tato matice nám představuje transponovanou matici X. Její tvar bude následující

$$\boldsymbol{X'} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 15892 & 4830 & \cdots & 12429 \\ 1{,}05 & 0{,}32 & \cdots & 1{,}46 \\ 6{,}95 & 8{,}39 & \cdots & 6{,}95 \end{pmatrix}. \tag{25}$$

Teď, když známe všechny matice potřebné pro výpočet lineárního modelu, můžeme je dosadit do vzorce

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X'X})^{-1}\boldsymbol{X'Y}\,, \tag{26}$$

a po výpočtu nám vyjde výsledný sloupcový vektor $\widehat{\boldsymbol{\beta}}$ jehož tvar je

$$\widehat{\boldsymbol{\beta}} = \begin{pmatrix} 25315{,}61 \\ 0{,}60 \\ -6354{,}51 \\ -469{,}10 \end{pmatrix}. \tag{27}$$

Tento vektor bude mít pro data z tabulky **Chyba! Nenalezen zdroj odkazů.** dvacet řádků s parametry $\hat{\beta}_0, \hat{\beta}_1,\ \hat{\beta}_2, \hat{\beta}_3$ a tyto parametry využijeme k následném výpočtu odhadu ceny $\widehat{\boldsymbol{Y}}$**.**

Obecný tvar pro výpočet odhadu parametru $\widehat{\boldsymbol{Y}}$ vypadá takto

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 - \hat{\beta}_{2,} X_2 - \hat{\beta}_3 X_3, \quad (28)$$

a po dosazení našich vypočtených parametrů $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_{2,} \hat{\beta}_3$ bude mít následující tvar

$$\hat{Y} = 25315{,}61 + 0{,}60 X_1 - 6354{,}51 X_2 - 469{,}10 X_3. \quad (29)$$

Tento tvar můžeme nyní použít pro odhad průměrné ceny za m^2 v konkrétní obci, když dosadíme za parametry X_1, X_2, X_3. První parametr X_1 je počet obyvatel, parametr X_2 představuje vzdálenost od krajského města a konečně parametr X_3 představuje nezaměstnanost v okrese. Nyní si vezměme konkrétní obec a dosaďme do těchto parametrů. Pro náš příklad vezměme tedy třeba obec Jedousov. Tato obec má 150 obyvatel, její vzdálenost od krajského města je 0,38 h a nezaměstnanost v okrese je 6,13 %. Nyní tyto parametry dosaďme do našeho vztahu s vypočítanými parametry $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_{2,} \hat{\beta}_3$ a po dosazení dostaneme následný tvar

$$Y = 25315{,}61 + 0{,}60 \cdot 150 - 6351{,}51 \cdot 0{,}38 - 469{,}10 \cdot 6{,}13, \quad (30)$$

a po výpočtu vychází odhad průměrné ceny za m^2 v obci Jedousov

$$Y = 20116{,}45. \quad (31)$$

Nyní známe, jak počítat odhad v dané konkrétní obci. V praktické části této diplomové práce jsou zde využity stejné parametry pro výpočet odhadu ceny v obci, avšak tyto výpočty jsou aplikovány na okresy, nikoliv na kraje. Pro místo parametru vzdálenost od krajského města je použita vzdálenost od okresního města. Konkrétní použití je popsáno v kapitole *Výpočet odhadu ceny za 1 m2 v aplikaci*.

Zdroj: (4)

2.3.2 Sestrojení grafu vývoje průměrné ceny v okrese

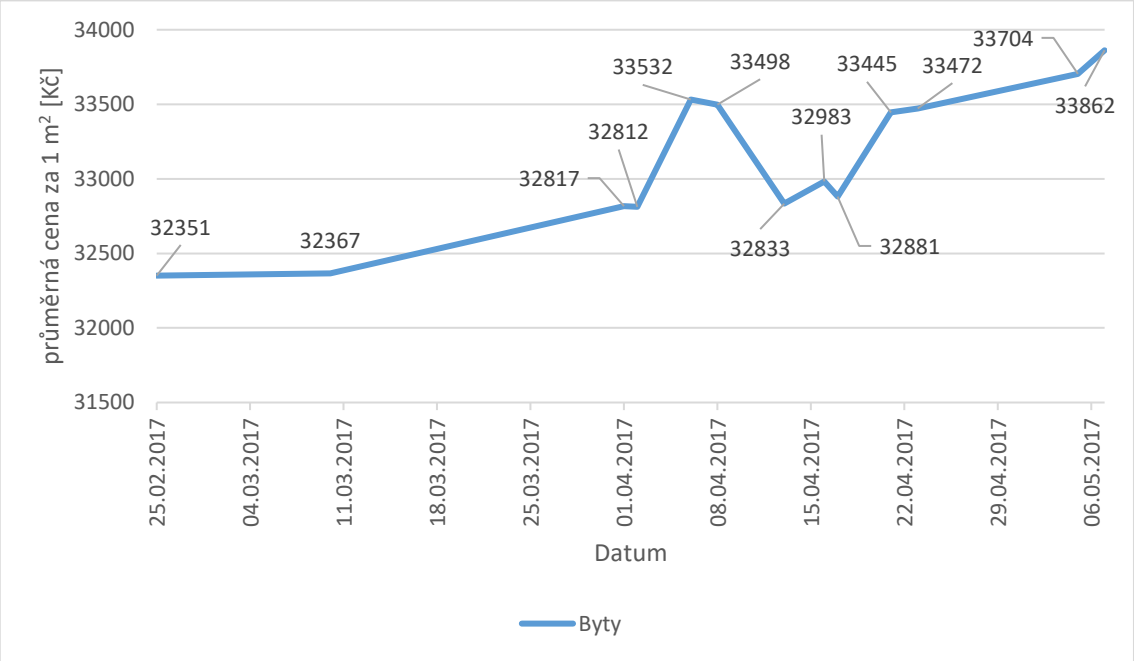
Tato podsekce kapitoly Metodika je věnováno sestavení grafu vývoje ceny za 1 m^2 v daném okrese. Nejprve je potřeba získat ukázková data, na kterých bude příklad ukázán. Data pro ukázku byla získána přímo z aplikace a představují vývoj průměrné ceny za 1 m^2 pro okres Pardubice v období od 25. února 2017 do 7. května 2017. Tato data jsou reálně vypočítána z načtených inzerátů bytů v okrese Pardubice. V tabulce Tabulka 2-2 jsou zobrazena data ve dvou sloupcích.

První sloupec obsahuje data, kdy byla data ze serveru načítána, druhý sloupec představuje vypočítané průměrné ceny za 1 m^2 ke konkrétnímu datu. Důležité je samozřejmě mít data chronologicky utříděna dle data od nejstaršího k nejmladšímu, aby bylo možné křivku zobrazující časovou řadu vytvořit.

okres Pardubice	
Datum	**Prům. cena za 1 m^2 [Kč]**
25. 02. 2017	32351
10. 03. 2017	32367
01. 04. 2017	32817
02. 04. 2017	32812
06. 04. 2017	33532
08. 04. 2017	33498
13. 04. 2017	32833
16. 04. 2017	32983
17. 04. 2017	32881
21. 04. 2017	33445
23. 04. 2017	33472
05. 05. 2017	33704
07. 05. 2017	33862

Tabulka 2-2 - Vývoj ceny v okresu Pardubice, zdroj: vlastní

Data pro sestavení jsou známa, proto nyní lze přejít k samotné konstrukci grafu. Z dat obsažených v tabulce postačí s přehledem spojnicový graf, kde na ose x budou ležet data načtení průměrných cen a na ose y vypočtené hodnoty průměrných cen za 1 m^2 u bytů ke konkrétnímu datu.



Obrázek 2.2 - Vývoj průměrné ceny za 1 m^2 pro okres Pardubice, zdroj: vlastní

Na vytvořeném grafu je vidět vývoj ceny v pardubickém okresu za vybrané období. Kdyby bylo pozorování vývoje této ceny delší, mohl by více tento vývoj analyzovat. K analýze dat by se dalo využít například statistické analýzy časových řad a sledovat tak zda se objevují trendové, sezónní, cyklické, anebo náhodné složky ve vývoji průměrné ceny za 1 m^2 u bytů za dané období.

2.3.3 Výpočet podobnosti vývoje průměrné ceny mezi okresy

V této podkapitole je popsána metodika k získání podobnosti vývoje průměrné ceny za 1 m^2. Nejprve je potřeba získat ukázková data pro výpočet. Jako v přechozí podkapitole data pro ukázku jsou vzata přímo z aplikace. Stejně také představují vývoj průměrné ceny bytů za 1 m^2 v období od 25. února 2017 do 7. května 2017, avšak pro všechny okresy pardubického kraje. V tabulce Tabulka 2-3 je pět sloupečku, kde první představuje datum vypočtených průměrných cen za 1 m^2, druhý až pátý představují průměrné ceny v okresech Pardubice, Chrudim, Svitavy a Ústí nad Orlicí.

	okres Pardubice	okres Chrudim	okres Svitavy	okres Ústí nad Orlicí
Datum	**Průměrná cena za 1 m2**			
25. 02. 2017	32351	26656	18090	22446
10. 03. 2017	32367	26656	18090	22412
01. 04. 2017	32817	27156	19388	20930
02. 04. 2017	32812	27175	19388	20930
06. 04. 2017	33532	27756	19784	21535
08. 04. 2017	33498	27718	19349	21719
13. 04. 2017	32833	27540	20073	21635
16. 04. 2017	32983	27541	19946	22248
17. 04. 2017	32881	27412	19946	22342
21. 04. 2017	33445	27458	20064	21975
23. 04. 2017	33472	27526	20215	22036
05. 05. 2017	33704	28169	20008	21621

Tabulka 2-3 - Vývoj ceny okresů Pardubického kraje, zdroj: vlastní

Ukázková data jsou získána, nyní je potřeba pomocí vhodného statistického nástroje zjistit podobnost vývoje průměrné ceny za 1 m^2 bytů mezi jednotlivými okresy. Je potřeba vhodně porovnat načtená data mezi jednotlivými okresy a určit, jak jsou si podobná, či vzdálená. Jedním z použitelných nástrojů pro výpočet podobnosti okresů je pomocí měr vzdálenosti.

Mezi nejznámějšímu typu měr vzdáleností patří Euklidovská vzdálenost, pomocí které lze porovnat jednotlivé složky vektoru načtených průměrných cen a získat tak hodnotu určující, do jaké míry jsou si okresy podobné či ne. Čím blíže se bude tato vypočtená hodnota blížit k nule, tím si budou okresy podobnější. Pro výpočet hodnoty vzdálenosti mezi okresy lze využít následujícího vztahu:

$$D_E(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{i,k} - x_{j,k})^2} = \|x_i - x_j\|, \tag{32}$$

kde $x_{i,k}$ a $x_{j,k}$ představují jednotlivé složky vektoru průměrných cen za 1 m^2 dvou okresů. Nyní bude ukázán příklad jak vypočítat podobnosti například mezi okresem Chrudim a okresem Pardubice. Proto si je potřeba dosadit do vztahu (32 konkrétní hodnoty z tabulky Tabulka 2-1 obou okresů a to se provede následujícím způsobem:

$$D_E(x_i, x_j) = \sqrt{\begin{matrix}(32351 - 25656)^2 + (32367 - 26656)^2 + \\ + (32367 - 26656)^2 + \cdots + (33704 - 28169)^2\end{matrix}}. \tag{33}$$

Po výpočtu následující rovnice vyjde hodnota, která představuje podobnost těchto dvou okresů a ta je rovna:

$$D_E(x_i, x_j) = \|20369\|. \tag{34}$$

Tento výpočet lze aplikovat mezi postupně mezi kombinacemi dvou krajů do té doby, než budeme znát vzdálenosti mezi jednotlivými okresy. Nicméně tato výsledná data je potřeba nějak graficky znázornit, a pro lze využít tzv. matici vzdáleností mezi okresy. Příklad této matice vzdáleností je zobrazena v tabulce Tabulka 2-4.

	okres Pardubice	okres Chrudim	okres Svitavy	okres Ústí nad Orlicí
okres Pardubice	0	20369	48858	40759
okres Chrudim	20369	0	28413	20489
okres Svitavy	48858	28513	0	8820
okres Ústí nad Orlicí	40759	20489	8820	0

Tabulka 2-4 - Matice Euklidovských vzdáleností mezi okresy Pardubického kraje, zdroj: vlastní

Výsledná matice představuje symetrickou matici, kde na hlavní diagonále jsou samé nuly a ostatní buňky této matice představují, jakou mají mezi sebou okresy Pardubického kraje vzdálenost. To znamená, čím menší vzdálenost, tím jsou si okresy podobnější.

2.4 Výsledky

Tato podkapitola zahrnuje výsledky použitých metodik a také diskuzi ohledně použitých nástrojů a zvolení daných řešení. Jsou zde popsány postupy, jak bylo cílů diplomové práce dosaženo a jaké problémy při realizaci aplikace nastaly. Je zde popsáno, jak bylo dosaženo zobrazení prostorových informací nad územním členění České

republiky, jaký nástroj byl využit pro tvorbu grafu vývoje průměrné ceny. Dále pak, kde byl využit odhad průměrné ceny za 1 m^2 pro byty a také jakým stylem byla řešena podobnost vývoje průměrné ceny a její grafické znázornění.

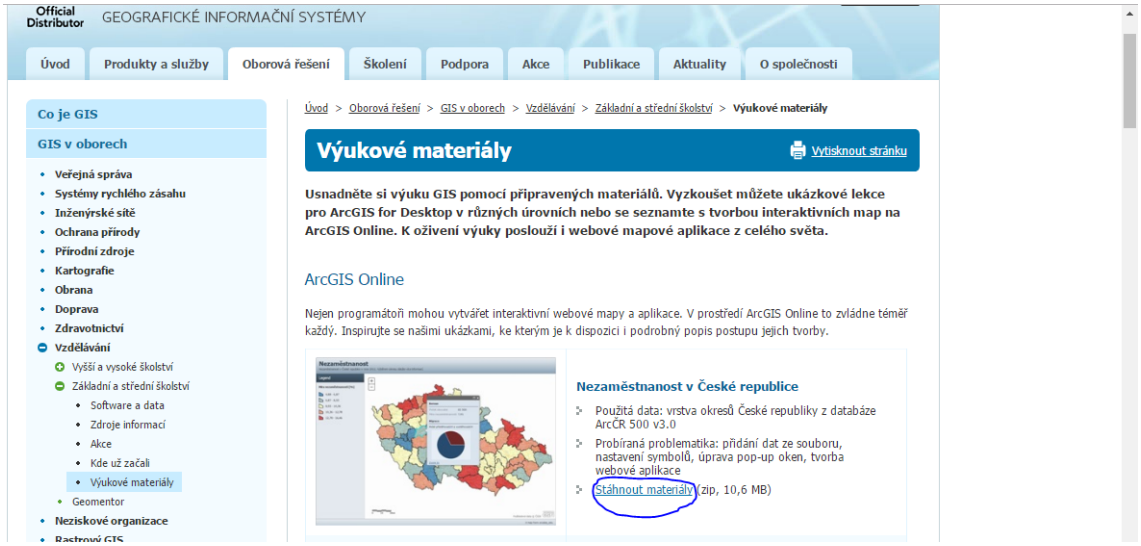
Otázkou může být, proč je práce zaměřena pouze na porovnání bytů. Jedná se o to, že u bytů lze snadněji určit, průměrnou cenu za 1 m^2 z celkové ceny bytu dělené výměrou. U ostatních nemovitostí to nemusí být tak zřejmé například u domů sice nějakou cenu za 1 m^2 lze vypočítat z poměru zastavěné plochy, ale už se v inzerátech neuvádí počet pater a obytná plocha. Dále cena nemusí odpovídat cenám dané lokality, v jaké se dům nachází, protože v inzerátech není určeno, v jakém stavu se nemovitosti nachází. Proto tedy bylo zvoleno pouze porovnání cen u bytů.

Navržená aplikace načítá všechny typy nemovitostí ze serveru Sreality.cz, kromě projektů. Takže vyvinutá aplikace je připravena, v případě nalezení způsobu porovnání ostatních nemovitostí, k případnému využití těchto načtených dat a dalšímu statistickému zpracování.

2.4.1 Zobrazení průměrných cen na mapě

Hlavní myšlenkou je zobrazení prostorových informací na mapě, dle územního členění, to znamená v případě této práce, by měla zobrazovat průměrnou cenu za 1 m^2 pro byty. Proč zrovna pro bylo zvoleno porovnání pro tento typ nemovitosti, je uvedeno v úvodu této podkapitoly. Bylo tedy potřeba najít vhodný nástroj pro zakreslení vypočtených informací a online zobrazení pro uživatele. K takovémuto zakreslení slouží tzv. GIS nástroje. Prvním z uvažovaných nástrojů k zobrazení těchto informací byl nástroj ArcGIS online. Tento nástroj umožňuje vytvářet online aplikace geografických map a případného zobrazení na vlastní webové stránce.

Nejprve bylo tedy potřeba provézt registraci na stránkách. Aby bylo možné oprávnění pro vytváření vlastní mapových podkladů. Ideálně by bylo mít zakresleny prostorové informace zakresleny ve všech okresech České republiky. Jenže jak toho docílit? K tomuto cíli je potřeba znát souřadnice všech polygonů okresů, aby toto vykreslení bylo proveditelné. Po delším pátrání byla nalezena stránka s testovacími daty přímo od společnosti ArcData na adrese *http://old.arcdata.cz/oborova-reseni/gis-v-oborech/vzdelavani/zakladni-a-stredni-skolstvi/vyukove-materialy/*, kde lze stáhnout výukové materiály nezaměstnanosti České republiky.



Obrázek 2.3 - Stažení výukových dat z portálu ArcData, zdroj: http://old.arcdata.cz/

Stažený zip soubor obsahuje ve svém archivu dvě položky, jedna je popis přiložených dat v dokumentu aplikace Microsoft Word a druhá položka je zip archiv se soubory obsahující polygony okresů pro vykreslení v aplikacích GIS. Dále v tomto archivu je obsažen soubor s názvem *okresy_sim.dbf*, který je důležitý pro zobrazovaná data, jak vypadá jeho obsah je vidět na obrázku Obrázek 2.4.

A1 | OBJECTID,N,9,0

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	OBJECTID,N,9,0	NK,C,2	KN	KNOI	KODNUT	LAU1,C,	KNUT	KOD	NAZEVKR,C,40	NAZKR,C,20	NAZOK,C,40	VYMERU,N,	OB91,N,	OB01,N,
2	1	HP	01	0100	CZ0100	CZ0100	1100	3100	HlavnÝ mýsto Praha	HlavnÝ mýsto Praha	HlavnÝ mýsto Praha	49613,0162	1214174	1169106
3	2	ST	02	0201	CZ0201	CZ0201	2101	3201	St°edoŘeskř kraj	St°edoŘeskř	BeneÜov	147471,2508	91619	90625
4	3	ST	02	0202	CZ0202	CZ0202	2102	3202	St°edoŘeskř kraj	St°edoŘeskř	Beroun	66186,3988	75859	75684
5	4	ST	02	0203	CZ0203	CZ0203	2103	3203	St°edoŘeskř kraj	St°edoŘeskř	Kladno	71963,8127	150625	151360
6	5	ST	02	0204	CZ0204	CZ0204	2104	3204	St°edoŘeskř kraj	St°edoŘeskř	KolÝn	74321,1493	91637	89379
7	6	ST	02	0205	CZ0205	CZ0205	2105	3205	St°edoŘeskř kraj	St°edoŘeskř	Kutnß Hora	91715,7944	75250	73628
8	7	ST	02	0206	CZ0206	CZ0206	2106	3206	St°edoŘeskř kraj	St°edoŘeskř	MýlnÝk	70105,9924	94241	94635
9	8	ST	02	0207	CZ0207	CZ0207	2107	3207	St°edoŘeskř kraj	St°edoŘeskř	Mladß Boleslav	102283,3948	110664	113241
10	9	ST	02	0208	CZ0208	CZ0208	2108	3208	St°edoŘeskř kraj	St°edoŘeskř	Nymburk	85018,6447	81247	82804
11	10	ST	02	0209	CZ0209	CZ0209	2109	3209	St°edoŘeskř kraj	St°edoŘeskř	Praha-vřchod	75567,6471	101923	105541
12	11	ST	02	0210	CZ020A	CZ020A	2110	3210	St°edoŘeskř kraj	St°edoŘeskř	Praha-zßpad	58033,4273	74265	82404
13	12	ST	02	0211	CZ020B	CZ020B	2111	3211	St°edoŘeskř kraj	St°edoŘeskř	P°Ýbram	169235,8695	112007	110685
14	13	ST	02	0212	CZ020C	CZ020C	2112	3212	St°edoŘeskř kraj	St°edoŘeskř	RakovnÝk	89625,8319	53545	52487
15	14	JC	03	0301	CZ0311	CZ0311	3101	3301	JihoŘeskÝ kraj	JihoŘeskÝ	LeskÚ Budýjovice	163858,434	173765	178506
16	15	JC	03	0302	CZ0312	CZ0312	3102	3302	JihoŘeskÝ kraj	JihoŘeskÝ	Leskř Krumlov	161463,219	57388	59569
17	16	JC	03	0303	CZ0313	CZ0313	3103	3303	JihoŘeskÝ kraj	JihoŘeskÝ	Jind°ich˙v Hradec	194354,5664	93048	92887
18	17	VY	10	1003	CZ0633	CZ0633	6103	3304	Kraj VysoŘina	VysoŘina	Pelh°imov	128984,7555	74614	72984
19	18	JC	03	0304	CZ0314	CZ0314	3104	3305	JihoŘeskÝ kraj	JihoŘeskÝ	PÝsek	112678,8186	71747	70300
20	19	JC	03	0305	CZ0315	CZ0315	3105	3306	JihoŘeskÝ kraj	JihoŘeskÝ	Prachatice	137504,8632	50985	51369

Obrázek 2.4 - Data pro polygony okresů, zdroj: vlastní

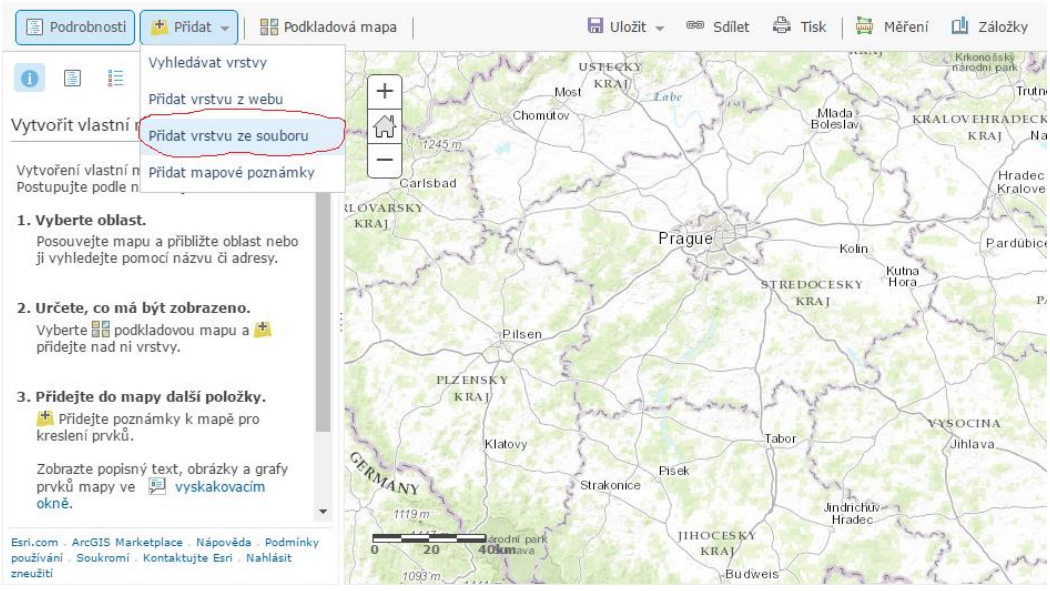
Data lze tedy upravit dle vlastní potřeby, nicméně je potřeba aby, byl zachován sloupeček s názvem *OBJECTID,N,9,0*, protože obsahuje ID, které slouží k identifikaci jednotlivých polygonů okresů. Takže po úpravě mohou vypadat data takto, jak je možné vidět na obrázku .

B29 PlzeÕ

	A	B	C	D	E	F	G	H	I
1	OBJECTID	NAZEV_OKRE,C,19	PRUM_CENA,N,8,0						
2	1	Praha	86843						
3	2	Benešov	29578						
4	3	Beroun	45468						
5	4	Kladno	31663						
6	5	Kolín	27557						
7	6	Kutná Hora	24590						
8	7	Mělník	30000						
9	8	Mladá Boleslav	37658						
10	9	Nymburk	34302						
11	10	Praha	47076						
12	11	Praha	49385						
13	12	Příbram	25915						
14	13	Rakovník	23541						
15	14	Èeské Budějovice	32675						
16	15	Èeský Krumlov	33673						
17	16	Jindřichův Hradec	22519						
18	17	Pelhřimov	18104						
19	18	Písek	23196						
20	19	Prachatice	18611						
21	20	Strakonice	22035						
22	21	Tábor	19705						
23	22	Domažlice	19009						
24	23	Cheb	38122						
25	24	Karlovy Vary	36493						

Obrázek 2.5 - Vlastní data pro polygony okresů, zdroj: vlastní

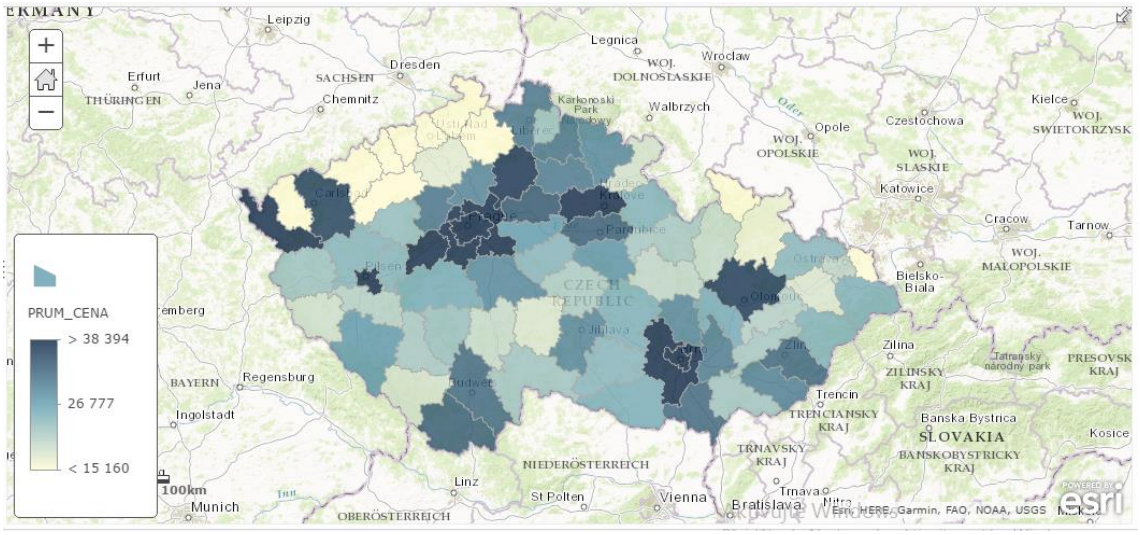
Takto upravený soubor je společně s celým zip souborem okresy_CR.zip připraven na nahrání do webové aplikace ArcGIS online. Zobrazení informací by postačil samostatný soubor *okresy_sim.dbf*, nicméně by nedošlo k vykreslení polygonů okresů. Proto je tedy potřeba nahrát celý zip archiv. Po registraci a přihlášení do aplikace ArcGis online, lze tento zip archiv nahrát, pomocí přidání vrstvy ze souboru a to je možné vidět na obrázku Obrázek 2.6.



Obrázek 2.6 - Přidání vlastní vrstvy do aplikace ArcGIS online, zdroj: vlastní

Po nahrání vlastní vrstvy a nastavení možnosti vykreslení polygonu vznikla mapa s rozdělením České republiky a dle barev od bílé po tmavě modrou jsou zobrazeny informace o průměrné ceně v okresech, kde bílá představuje okres s nejlevnější

průměrnou cenou bytů za 1 m^2 a tmavě modrá představuje nejdražší okres s nejdražší průměrnou cenou bytů za 1 m^2. Výsledná mapa je vidět na obrázku Obrázek 2.7.

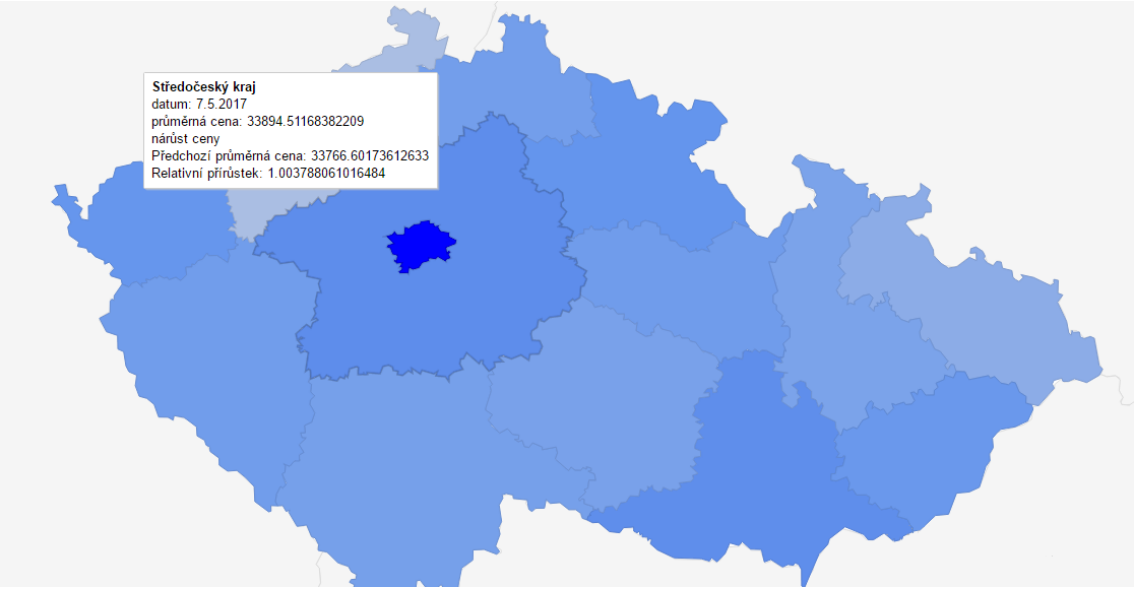


Obrázek 2.7 – Vlastní mapa v aplikaci ArcGIS online, zdroj: vlastní

Výslednou mapu lze sdílet pomocí url adresy na konkrétní stránku webové aplikace, nebo přímo z ní vytvořit mapovou aplikaci. Nicméně při sdílení této mapy na konkrétní webovou stránku aplikace se zobrazí pouze statická data načtená ze souboru. Jak tedy vyřešit zobrazení dynamických dat vypočtených ve vlastní aplikaci? Jedno z řešení je si vytvořit vlastní WMS webovou službu, která by spolupracovala s API rozhraním aplikace ArcGIS online a tato služba je popsána na stránkách https://developers.arcgis.com/javascript/. K tvorbě této služby je potřeba alespoň základní znalost programovacího jazyka JavaScript a nastudování a tvorba vlastní služby by již přesahovala rozsah diplomové práce.

Proto bylo nalezeno řešení v podobě využití geoinformatického nástroje od společnosti Google a to nástroje Google Geochart. Tento nástroj umožnuje zobrazovat data přímo na webové stránce aplikace přesně tam, kde jej potřebujeme využít. Využívá načtených knihoven přímo od společnosti Google a pomocí JavaScript jazyka dynamické zobrazení na stránce. Jedinou nevýhodou tohoto nástroje je ta, že neumí v nastavení rozdělit Českou republiku na okresy. Jak tato služba funguje a její nastavení je popsáno v kapitole *Google geochart* a konkrétní využití v aplikaci zase v popisu metod aplikace.

Výsledná mapa pomocí Google geochart tedy dokáže jednodušeji zobrazit dynamická data, nicméně nevýhodou je, že je lze zobrazit pouze na úroveň krajů. Jak toto rozdělení mapy ČR vypadá, je možné vidět na obrázku Obrázek 2.8.



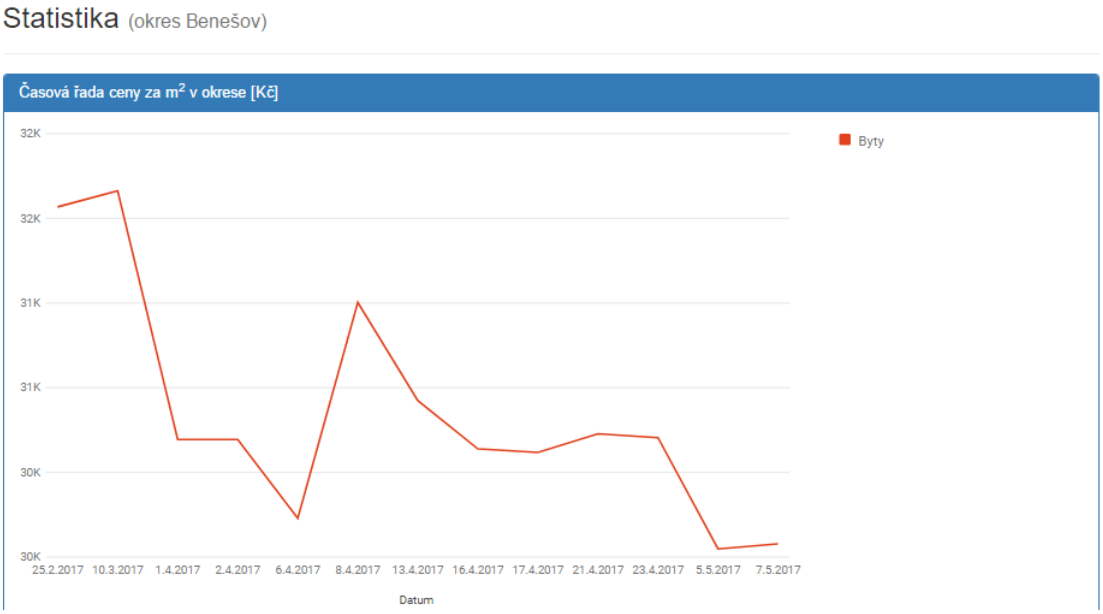
Obrázek 2.8 - Vykreslená mapa pomocí Google geochart, zdroj: vlastní

2.4.2 Sestrojení grafu vývoje průměrných cen

Vyvinutá aplikace by měla uživateli zobrazit graf vývoje průměrné ceny za 1 m^2, a proto bylo potřeba tento graf sestrojit. K dispozici ve vývoji je mnoho nástrojů a knihoven pro toto zobrazení. Například programovací jazyk C# nabízí řadu knihoven a možnosti stáhnutí přímo do aplikace. Mnoho knihoven sice dokáže vykreslit graf, ale výsledné grafy nejsou estetické a interaktivní. Proto bylo potřeba najít knihovnu nebo nástroj, který dokáže být interaktivní a zároveň bude esteticky zapadat do celého konceptu aplikace.

Proto v sestrojení byl, díky dřívějším získaným znalostem a dobře popsaného rozhraní, použit nástroj opět použit nástroj od společnosti Google a to Google Charts. Opět bylo využito externích knihoven od společnosti Google využívajících programovací jazyk JavaScript a díky tomu je možné docílit interaktivních grafů. Lze tedy efektivně využívat předdefinovaného a odladěného nástroje s vlastními daty a následně jej využít na potřebné webové stránce.

Pomocí tohoto grafu se vykresluje v aplikaci křivka vývoje průměrné ceny za 1 m^2 v daném okrese. Je zde využito metodiky z kapitoly *Sestrojení grafu vývoje průměrné ceny v okrese* a tedy na ose x byly zobrazeny data, kdy byly průměrné ceny počítány a na ose y pak vypočtené průměrné ceny za 1 m^2 u bytů. Výsledný graf poté je možné vidět na obrázku Obrázek 2.9, kde je zobrazen vývoj ceny v okrese Benešov za období od 25. února do 7. května 2017.



Obrázek 2.9 - Vývoj průměrné ceny za 1 m² u bytů v okrese Benešov, zdroj: vlastní

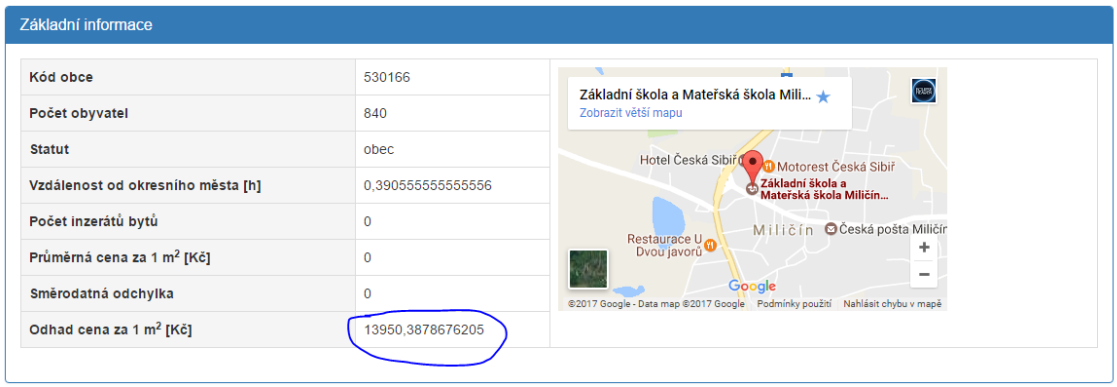
2.4.3 Výpočet odhadu ceny v obci

Aby aplikace umožňovala výpočet odhadu průměrné ceny za 1 m² v konkrétní obci, bylo potřeba využít metodiky *Příklad výpočtu regresního modelu* a bylo třeba získat potřebné faktory pro výpočet. Těmi faktory byly počet obyvatel, vzdálenost k okresnímu městu a nezaměstnanost v okrese. V kapitole *Analýza dat* byla věnována pozornost, jak tato data získat. Je zde využit malý lexikon obcí získaný z Českého statistického úřadu a dále pak získaná nezaměstnanost v okrese ze stránek Ministerstva práce a sociálních věcí. Poslední faktor vzdálenost k okresnímu městu bylo tedy potřeba získat.

K výpočtu vzdálenosti bylo využito dalšího nástroje od společnosti Google a to nástroje Google Distance Matrix. Kde je možné zadat vektor výchozích bodů a vektor cílových bodů pro výpočet vzdálenosti. V této aplikaci byl potřeba zjistit zejména čas dojezdu k okresnímu městu, což tento nástroj také poskytuje. Využití je zejména v dotazech na API rozhraní, kde se do parametrů zadávajíc konkrétní města (případně GPS souřadnice) a toto rozhraní vrací data v podobě JSON objektů, anebo XML. Konkrétní využití nástroje Google Distance Matrix je popsáno v kapitole *Realizace aplikace*.

Jakmile byly zjištěny všechny faktory pro výpočet, mohlo se začít s výpočtem odhadu pro jednotlivé obce ČR. Jelikož je tato práce zaměřena spíše na rozdělení na okresy, byly odhadnuté parametry pro výpočet odhadu ceny za 1 m² počítány

nad jednotlivými okresy České republiky. Tudíž aplikace umožnuje zobrazit odhadnutou cenu za 1 m^2 u bytů v obci, která má jeden nebo žádný inzerát. To znamená, že uživatel si může udělat obrázek, kolik by zhruba mohl stát byt v konkrétní obci. Jak tento odhad je zobrazen v aplikaci je možné vidět na obrázku Obrázek 2.10.



Obrázek 2.10 - Informace o obci v aplikaci, zdroj: vlastní

Nicméně při vývoji aplikace bylo zjištěno, že při výpočtu odhadu se v některých okresech cena nevypočítala a to z důvodu toho, že při využití vztahů z metodiky *Příklad výpočtu regresního modelu* docházelo k lineární kombinaci řádků matice X a po její transpozici nešlo vypočítat inverzní matici, protože determinant této matice byl nula.

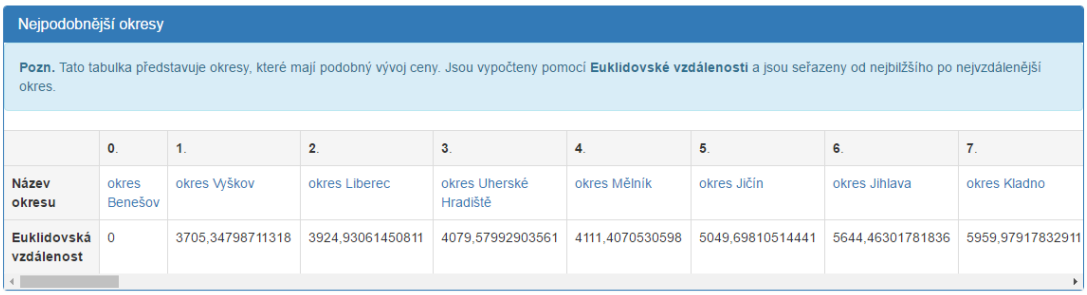
Počet řádků matice určoval obce v okrese, které měly aktuální inzeráty na serveru Sreality.cz a tento problém nastal zejména při malém počtu obcí obsahujících inzeráty na tomto serveru. Proto bylo zvoleno řešení odebrání faktoru nezaměstnanosti v okrese z výpočtu lineárního regresního modelu, aby nedocházelo k lineární kombinaci řádků matice X a aby bylo možné vypočítat odhad pro všechny obce ČR.

2.4.4 Zobrazení podobnosti okresů

V požadavcích pro vývoj aplikace bylo také, že by měla zobrazovat podobnost vývoje ceny za 1 m^2 u bytů. K této realizaci bylo využito metodiky z kapitoly *Výpočet podobnosti vývoje průměrné ceny mezi okresy*, pomocí které se počítala podobnost vývoje ceny mezi všemi okresy České republiky a částmi Prahy.

Programově toto řešení nebylo složité, algoritmus pro tento výpočet je popsán v kapitole *Důležité metody aplikace*, nicméně o něco složitější bylo vymyslet, jak tato data zobrazit. V kapitole *Výpočet podobnosti vývoje průměrné ceny mezi okresy* byla výsledná data zobrazena do matice vzdáleností, kde prvky matice představovaly hodnotu vzdálenosti mezi okresy. Čím tato hodnota byla menší, tím si byly okresy více podobné

a naopak. Proto pro tolik prvků matice by byla jednak zvýšena náročnost výpočtů pro všechny okresy, jednak by výsledná matice nebyla přehledná. Proto byl zvolen způsob zobrazení do řádkového vektoru (a aplikaci zobrazeno v řádkové tabulce), kde jsou vzdálenosti mezi okresy zobrazeny od nejbližšího po nejvzdálenější. Tento vektor se počítá vždy pro zobrazení detailu okresu. Jak tento vektor (tabulka) vypadá je možné vidět na obrázku Obrázek 2.11.



Nejpodobnější okresy

Pozn. Tato tabulka představuje okresy, které mají podobný vývoj ceny. Jsou vypočteny pomocí **Euklidovské vzdálenosti** a jsou seřazeny od nejbližšího po nejvzdálenější okres.

	0.	1.	2.	3.	4.	5.	6.	7.
Název okresu	okres Benešov	okres Vyškov	okres Liberec	okres Uherské Hradiště	okres Mělník	okres Jičín	okres Jihlava	okres Kladno
Euklidovská vzdálenost	0	3705,34798711318	3924,93061450811	4079,57992903561	4111,4070530598	5049,69810514441	5644,46301781836	5959,97917832911

Obrázek 2.11 - Tabulka euklidovských vzdálenosti mezi okresy, zdroj: vlastní

3 REALIZACE APLIKACE

Tato kapitola se zabývá použitými technologiemi a softwary pro tvorbu aplikace. Nalézt zde lze návrh databáze včetně popisu databázových tabulek. Jsou zde popsány využité knihovny a nástroje pro sestrojení grafů, efektivnějšího načítání do databáze a dále pak popis důležitých metod aplikace. Pro porozumění této kapitoly je potřeba znalostí základů programování a zejména základů programovacího jazyka C#.

3.1 Použité technologie

3.1.1 Programovací jazyk C#

Programovací jazyk C# se řadí mezi tzv. objektově orientované programovací jazyky. Byl vyvinut společně s frameworkem .NET a jeho syntaktický zápis vychází z programovacích jazyků C++ a Javy. Jeho první oficiální verze byla vydána v roce 2002 a nesla označení C# 1.0. Zároveň vyšla s touto verzí jazyka také vyšel i .NET Framework 1.0. V roce 2005 přišli vývojáři s další verzí (C# 2.0), kde hlavním přínosem této verze bylo přidání částečných statických tříd, Iterátorů a anonymních metod.

Další verze (C# 3.0) spatřila světlo světa v roce 2007 a vyšla i společně s novou verzí .NET frameworku 3.5. Největší přínos však byl ten, že tato verze s sebou přinesla i technologii LINQ. Tato technologie je podrobněji popsána v kapitole. Dále verze 3.0 přinesla kromě zmíněné technologie LINQ také podporu anonymních typů.

Poslední verze 6.0, vydaná v roce 2015, zatím nepřináší žádné zásadní změny, avšak její předností je vylepšená kontrola syntaxe.

Zdroj: (8)

3.1.2 Technologie LINQ

Technologie LINQ představuje sadu rozšíření zejména pro programovací jazyky C# a Visual basic a je součástí .NET frameworku. Výhodou této technologie, díky její vysoké abstrakci, je především vykonávání dotazů nad různými zdroji dat.

Využívá tzv. klíčová slova, která jsou využívána pro tvorbu dotazovacích výrazů, a pomocí kterých lze zdrojová data řadit, filtrovat, seskupovat, vybírat atd. Zdrojová data využívaná technologií LINQ, jsou zpravidla LINQ to Objects, LINQ to DataSet, LINQ to SQL a LINQ to XML. Výhoda použití LINQ oproti ostatním technologiím (např. Embedded SQL apod.) je ta, že vývojář nemusí přepínat mezi přepínat mezi syntaxemi programovacího jazyka a jazyka SQL.

Zdroj: (9), (10)

3.1.3 ASP .NET a návrhový vzor WebForms a MVC

Popis technologie ASP .NET

Technologie ASP .NET framework sloužící pro vývoj zejména webových aplikací, využívajících HTML, CSS a JavaScript. Umožnuje také vytvářet webové API a v reálném čase tedy využívat například tvz. Web Sockety. Jeho obsahuje sadu knihoven s hotovými řešeními mnoha základních problémů při vývoji webových technologií. To znamená, že je do něj integrována například bezpečnost, autentifikace uživatele, práce s databázemi apod. Tento Framework se kombinuje nejčastěji s programovacím jazykem C# a nebo s Visual Basicem.

ASP .NET lze tedy využít při vývoji od osobních webů, větších projektů až po webové portály. Využívá architekturu klient – server a výstupem této technologie je HTML stránka. Na obrázcích **(doplnit obrázek)** a **(doplnit obrázek)** můžeme zahlédnout porovnání tec

Pro vývoj aplikací využívajících tento Framework lze využít dvou návrhových vzorů WebForms a MVC jejich architekturu si popíšeme v následujících podkapitolách.

WebForms

Hlavní myšlenkou architektury WebForms bylo přenést architekturu WinForms na web, to znamená, že pomocí designeru poskládat formulář ze známých komponent, jako jsou např. tlačítka, popisky, textová pole, kombinovaní seznam, a přiřadit těmto komponentám nějaké události. Výsledkem je tedy to, že na první pohled se aplikace tváří a chová jako desktopová, avšak implementace logiky na pozadí je složitější. WebForms využívají zejména protokol HTTP, nicméně tento protokol má jednu nevýhodu a to bezstavovost. To znamená značnou nevýhodu pro server, jelikož nemá přehled o stavu aplikace, tudíž pouze zaznamená požadavek od uživatele a následně pak vygeneruje HTML stránku.

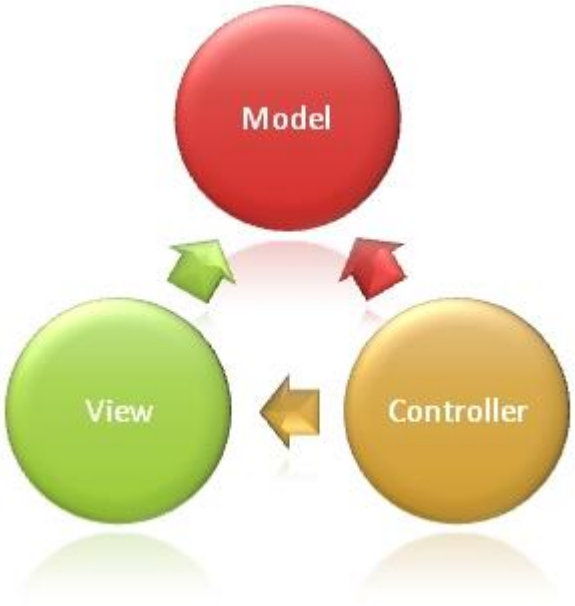
Při vytváření formulářových aplikací je potřeba udržet stav zejména, když odešleme hodnoty vyplněné v textových polích na server a on nám je po validaci vrátí s chybou. V tomto případě by bylo pro uživatele nepříjemné celý formulář vyplňovat znovu. Pro udržení stavu v této architektuře se používá tzv. ViewState, který je charakterizován jako skryté formulářové pole a vněm uchovává stav aplikace a které pak odešle s další dotazem

na server. Server u sebe zaregistruje stav aplikace, následně pak vykoná nějakou akci a vygeneruje nový ViewState.

Jednou z výhod je rychlost při vytváření webových aplikací obsahujících velké množství formulářů. Nicméně nevýhody jsou, jednak že se odesílá poněkud větší HTML stránka, a jednak že tato architektura je celkem komplikovanější. Tato architektura je poněkud starší oproti novějšímu konceptu MVC.

MVC

MVC je architektura dělící se na tři základní logické části, jak již ze zkratky může být zřejmé, a to Model, View a Controller. Nyní si trochu nastíníme logiku celé architektury a postupně si popíšeme jednotlivé části návrhového vzoru MVC.



Obrázek 3.1 - Návrhový vzor MVC

Model

Pod tím to pojmem si může představit jakousi komponentu, která pracuje se sadou datových objektů. To znamená, že model jsou pouze data plus business logika. Tato komponenta komunikuje hlavně s Controllerem.

Controller

Tato komponenta se stará o to, aby se aktualizoval model, provedla se nějaká operace s daty a aby došlo k překreslení všech view. Jedná se tedy o řídící jednotku, která se stará o provázání funkčnosti aplikace.

View

View je v podstatě jakási šablona zobrazující data předaná od modelu a dalších prvků uživatelského rozhraní, které jsou zobrazeny ve výsledné HTML stránce.

Návrhový vzor MVC zpřehledňuje architekturu webových aplikací a dělí ji na dvě hlavní části, a to na logickou část a část s grafickými výstupy.

Zdroj: (11), (12)

3.1.4 Entity Framework

Pod pojmem Entity Framework si můžeme představit tzv. ORM, což značí objektově - relační mapování. To znamená, že vytvořené datové tabulky z databáze se přímo mapují na C# třídy. Toto mapování má značné výhody zejména v kódu, kde se pracuje pouze s objekty a tento Framework si sám generuje SQL dotazy. Výsledkem je objektová aplikace, při které vůbec nepřijdeme do styku s jazykem SQL. Tento Framework využívá dvou přístupů a ty jsou **Code First** a **Database First.**

První zmíněná metoda **Code First** funguje tak, že si nejprve vytvoříme klasickou C# třídu a následně pomocí Entity Framework vygenerujeme tabulku v databázi a její potřebný kontext. Při využití druhé metody **Database first,** jak již z názvu může být patrné, se nejprve založí databáze a pomocí Entity Frameworku z ní vygenerujeme třídy a kontext.

Zdroj: (13)

3.2 Použitý software

3.2.1 Toad Data Modeler

Toad data modeler je visuální modelovací software umožňující návrh relačních databázových systémů. Dále umožňuje rychlé nasazení na více než 20-ti různých platformách. Pomocí toho softwaru je možné vytvářet logické a fyzické datové modely, porovnávací a synchronizační modely, rychlé generování komplexních SQL/DDL dotazů, vytváření a úpravu databázových skriptů a provádění dopředného či reverzního inženýrství pro databáze a datové sklady.

Zdroj: (14)

3.2.2 Microsoft Visual Studio

Microsoft Visual Studio je sada nástrojů a služeb, která slouží k návrhu, vývoji, testování a nasazení softwarových aplikací. Umožňuje nasazení aplikací postavených pro desktopové prostředí Windows, webových aplikací postaveních na verzi HTML 5, SharePoint, nových univerzálních Windows 10 aplikací, mobilních aplikací, multiplatformních aplikací určených pro systémy Android a iOS, a také cloudových prostředí.

Zdroj: (15)

3.2.3 SQL Server Management Studio

SQL Server Management Studio je integrované prostředí pro řízení jakékoliv SQL infrastruktury od SQL serveru po SQL databázi. Tento software poskytuje nástroje pro nasazení, monitorování a aktualizaci komponent datových vrstev, kterými jsou například databáze a datové sklady. Dále umožňuje vytvoření databázových dotazů a skriptů. Pro vývoj aplikace byl použit zejména pro prvotní import dat a pak k následné kontrole správnosti uchovaných dat, které byly získávány z aplikace.

Zdroj: (16)

3.3 Použité knihovny

V této podkapitole jsou popsány použité knihovny při vývoji aplikace. Knihovny, které aplikace využívá a slouží k nadstandartním operacím, které programovací jazyk C# a Entity Framework neobsahuje.

3.3.1 Knihovna Newtonsoft Json.Net

Knihovna Newtonsoft Json.Net je výkonný Framework pro platformu .NET. Její součástí je JSON serializér pro konvertování objektů mezi .NET objekty a objekty formátu JSON. Obsahuje také tzv. LINQ to JSON funkci, pomocí které lze manuálně číst a zapisovat data ve formátu JSON. Její výhodou je, že tato knihovna je rychlejší než zabudované .NET JSON serializéry. V aplikaci byla využita zejména pro deserializaci příchozích dat ze serveru Sreality.cz ve formátu JSON.

Zdroj: (17)

3.3.2 Knihovna Math.Net.Numerics

Knihovna Math.Net.Numerics poskytuje metody a algoritmy pro numerické výpočty. V této knihovně jsou zahrnuty speciální matematické funkce, funkce pro výpočet lineární algebry, pravděpodobnostní modely pro statistické výpočty, generování náhodných čísel, výpočet interpolace, výpočty integrálů, výpočty regrese, optimalizačních problémů a mnoho dalších funkcí. Využití této knihovny v aplikace je zejména při sestavování matic regresního modelu a následného výpočtu odhady ceny v okresu pomocí operací nad maticemi viz. kapitola **Chyba! Nenalezen zdroj odkazů.**.

Zdroj: (18)

3.3.3 Knihovna Z.BulkOperations

Knihovna Z.BulkOperations slouží hlavně k zvýšení výkonu při ukládání velkého množství dat do databáze. Využívá škálovatelné hromadné databázové operace (update, insert, delete a merge) pro zlepšení výkonu ukládání dat do databáze. Výkon jednotlivých operací díky této knihovně je shrnut v tabulce Tabulka 3-1.

Operace	1000 řádků	10 000 řádků	100 000 řádků	1 000 000 řádků
Insert	6 ms	25 ms	200 ms	2 000 ms
Update	50 ms	80 ms	575 ms	6 500 ms
Delete	45 ms	70 ms	625 ms	6 800 ms
Merge	65 ms	160 ms	1 200 ms	12 000 ms

Tabulka 3-1 - Délka operací s využitím knihovny Z.Bulk.Operations, zdroj: (19)

Zdroj: (19)

3.4 Služby od společnosti Google

3.4.1 Google charts

Google chart je efektní, jednoduchý na použití a zdarma nástroj, kterým lze zobrazovat grafy na webových stránkách. Dají se díky němu zobrazovat od spojnicových grafů po hierarchické stromové mapy. Nejčastějším použití tohoto nástroje je s jednoduchým JavaSciptem, umístěným na webové stránce. Uživatel si nám načítá potřebné Google chart knihovny, seznam mapovaných dat k zobrazení, vybere potřebná nastavení k přizpůsobení vlastnímu grafu a vytváří JavaScriptový objekt s vlastním id. Následně pak na webové stránce umístěn tag s tímto zvoleným id zobrazuje výsledný graf.

Jak již bylo zmíněno Google charts využívají JavaScriptové třídy a díky nim umožňují použít mnoho druhů grafů. Ve výchozím vzhledu je již vše nastaveno pro

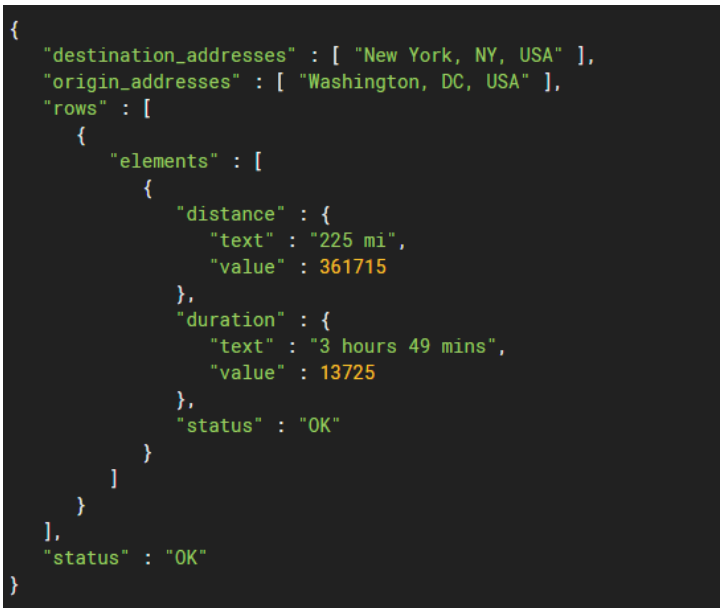
uživatelské použití, avšak uživatel si sám může upravit vzhled tak, aby zapadal do celkového vzhledu jeho aplikace. Tyto grafy jsou vysoce interaktivní a umožňují pomocí tzv. událostí jejich propojení s komplexními ovládacími panely nebo dalšími prvky webové stránky.

Grafy jsou vykresleny pomocí HTML/SVG technologie, která zaručuje kompatibilitu mezi jednotlivými prohlížeči (zahrnují VML pro starší verze prohlížeče Internet Explorer) a zajišťuje přenositelnost mezi platformami jako jsou iPhony, iPady a zařízeními běžícími na systému Android. Uživatel tedy nemusí řešit instalaci pluginů, či dalšího softwaru. To znamená, pokud tyto platformy mají webový prohlížeč, bez problémů se v něm tyto grafy vykreslí.

3.4.2 Google Distance Matrix API

Google Distance Matrix API je služba poskytující vzdálenost a čas pro matici počátečních a koncových destinací. Vrací informace založené na doporučené cestě mezi začátečními a koncovými body, vypočítané pomocí Google Maps API skládající se z řádků obsahujících dobu trvání a vzdálenost pro každou dvojici z matice. Nicméně tato služba nevrací podrobné informace o trase. K podrobným informacím o trase lze získat službou **Google Maps Direction API** a to předáním jedné počáteční a jedné koncové destinace.

Dotaz na službu Google Distance Matrix API je URL adresa využívá protokol http (https), která má podobu *https://maps.googleapis.com/maps/api/distancematrix/outputFormat?parameters.*
Parametr ***outputFormat*** představuje, v jaké podobě budou z této služby vrácena, a může být buď ***json*** (vrací data v JavaScript Object Notation) anebo ***xml*** (vrací data ve formátu XML). U této služby má URL adresa svá omezení a to, že musí být správně kódována a její délka **nesmí překročit 8192 znaků**. Což délka této adresy se u jiných prohlížečů, proxy serverů a serverů může lišit.



Obrázek 3.2 - Výstupní data ve formátu JSON, zdroj: (21)

Důležitá je bezpečnost a protokol https se doporučuje použít zejména tehdy, když aplikace odesílají citlivá uživatelská data, jako jsou např. aktuální poloha uživatele apod.

Tento dotaz kromě výstupního formátu dat, také **obsahuje doporučené parametry**, bez kterých by na serveru nemohl být vykonán.

První parametr je **parametr *origins***. Jedná se o parametr obsahující počáteční destinaci pro výpočet vzdálenosti a doby trvání trasy. Může obsahovat více lokací, které **se oddělují svislou čarou (|)**. Pokud se zadávají GPS souřadnice, jsou tímto znakem od sebe odděleny koordináty pro zeměpisnou šířku a zeměpisnou délku nebo ID místa.

Pokud je zadána adresa, služba řetězec s geokódy převede na koordináty zeměpisné šířky a délky pro výpočet vzdálenosti. Tyto koordináty mohou být odlišné, pokud jsou vraceny pomocí Google Maps Geocoding API. Při zadání přesné adresy vypadá hodnota parametru počáteční destinace např. takto ***origins=Bobcaygeon+ON|24+Sussex+Drive+Ottawa+ON.***

Je-li potřeba zadávat koordináty zeměpisné šířky a zeměpisné délky, neprovádí se jejich převádění a použijí se rovnou pro výpočet vzdálenosti. Nastavení pro tento případ má například tento tvar ***origins=41.43206,-81.38992|-33.86748,151.20699.***

Třetí možný způsob jak nastavit parametr ***origins*** je pomocí place ID. Nicméně před zadáním tohoto ID je potřeba mít nastavený prefix ***place_id.*** Tento prefix může být nastaven pokud dotaz obsahuje API klíč nebo Google Maps APIs Premium Plan client

ID. Tato ID lze načíst z Google Maps Geocoding API a Google Places API (zahrnující automatické doplňování místa). Pro nastavení parametru origins s ***place_id*** je využíván tento tvar ***origins=place_id:ChIJ3S-JXmauEmsRUcIaWtf4MzE.***

Parametr *destination* obsahuje jednu nebo více lokací, které jsou využity jako koncový bod pro výpočet vzdálenosti a doby trvání trasy. Jeho nastavení jsou totožná, jako u parametru ***origins*** a jsou popsány výše.

Posledním doporučeným parametrem je **parametr *key.*** U tohoto parametru se nastavuje tzv. API klíč, který představuje identifikaci aplikace využívající tuto službu. Tento klíč je důležitý pro účely správy kvót. Jelikož tato služba má svá omezení např. je dovoleno 2 500 položek za den (vypočítáno jako součet klientských a serverových dotazů), maximální počet počátečních destinací je 25 a 25 cílových destinací, 100 položek na jeden dotaz a 100 položek za 1 sekundu (vypočítáno jako součet klientských a serverových dotazů). Pro zvýšení limitů je potřeba zaplatit poplatek nebo si zařídit tzv. **Google Maps APIs Premium Plan.**

Další podrobnější informace a nastavení parametrů a Google Distance Matrix API lze na nalézt na stránkách https://developers.google.com/maps/documentation/distance-matrix/.

Zdroj: (20)

3.5 Popis implementace aplikace

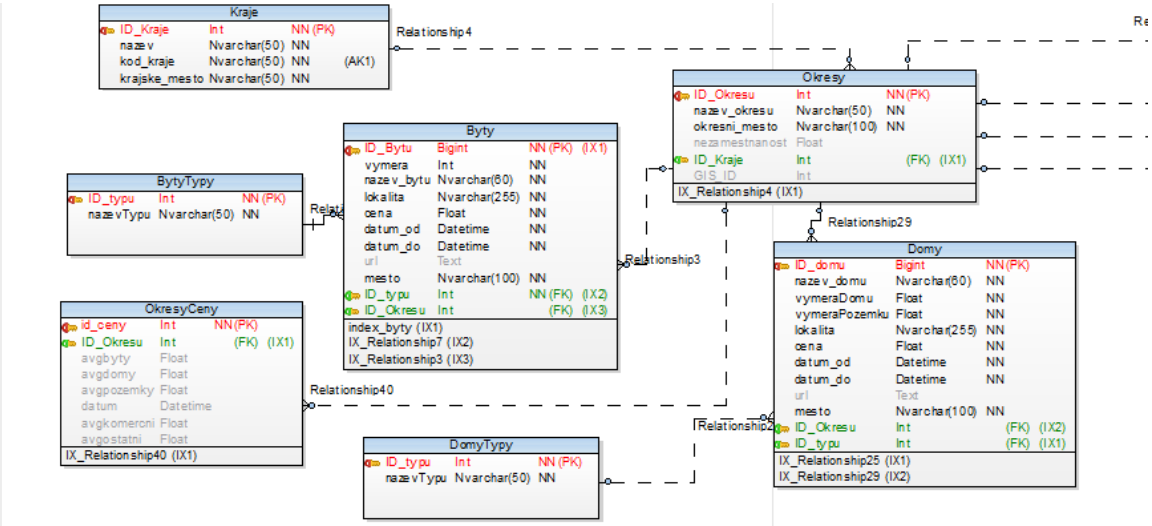
V této kapitole je popsána aplikace včetně popisu důležitých funkčností aplikace, jsou i zde popsány problémy při vývoji aplikace a jejich řešení. Dále pak, jsou zde popsány jednotlivé kroky, jako jsou získání dat ze serveru Sreality.cz, návrh databázových tabulek včetně jejich popisu, konkrétní využití externích knihoven a různých důležitých výpočtů pro výstupní zobrazení či export dat v různých formátech.

3.5.1 Popis databázových tabulek

V této části jsou popsány jednotlivé databázové tabulky, ve které jsou uložena data pro výpočet, uloženy inzeráty všech typů nemovitostí. Je zde popsáno všech 17 tabulek potřebných pro vývoj aplikace. Byl zvolen návrh pro databázi od Microsoftu, čili Microsoft SQL Server 2014. Důvod zvolení databáze od Microsoftu jen ten, že s použitým Entity Frameworkem a programovacím jazykem C# nejlépe spolupracuje. Nicméně na přiloženém CD jsou umístěny v souboru *DB_Scripts*.SQL DDL skripty

pro vytvoření této databáze, jelikož tato databáze byla z důvodu výkonu umístěna na databázový server a nemusí být zajištěna funkčnost této externí databáze do budoucna.

Jsou zde popsány všechny tabulky, jenž aplikace využívá. Nicméně některé slouží jen jako ukládání inzerátů a dalších informací připravených pro případné rozšíření aplikace. Součástí všech tabulek jednotlivých typů nemovitostí jsou atributy *datum_od* a *datum_do*. Tyto atributy slouží k informaci od jakého data, do jakého data byl inzerát aktivní, z důvodu toho, aby se odhady cen počítaly pouze z aktivních inzerátů.



Obrázek 3.3- Návrh databáze část 1, zdroj: vlastní

Tabulka Kraje

Tabulka kraje obsahuje pár základních informací o kraji. Je důležitá z hlediska územního členění inzerátu, anebo výpočtů. Obsahuje čtyři atributy *ID_kraje, nazev, kod_kraje, krajske_mesto.* Všechny tyto atributy jsou NOT NULL, čili při ukládání dat musí být vždy vyplněny. Atribut *ID_kraje* je primárním klíčem této tabulky a je nad ním nastaven index. Podle názvu atributů je zřejmé k jakému typu dat tabulka slouží.

Tabulka Okresy

Tabulka okresy opět ukládá základní informace o okresu. Má šest atributů *ID_Okresu, nazev_okresu, okresni_mesto, nezamestnanost, ID_kraje* a *GIS_ID.* Nejdůležitějšími atributy této tabulky jsou *ID_Okresu,* kde tento atribut představuje primární klíč tabulky. Atribut *okresni_mesto* představuje název okresního města (tento atribut je důležitý zejména při vypočítávání vzdálenosti k tomuto městu v aplikaci). Další atribut *nezamestnanost*, který uchovává míru nezaměstnanosti a ta je důležitá pro výpočet odhadu ceny. Posledním důležitým atributem je *GIS_ID*. Tento atribut je důležitý pro

export výsledných dat GIS aplikace např. ArcGis a přiřazuje jednotlivým okresům jejich ID, které spárováno s polygony jednotlivých okresů.

Tabulka OkresyCeny

Tato tabulka slouží k uchování průměrných cen nemovitostí, které se počítají v aplikaci. Také pomocí dat této tabulky dochází k vykreslení časových řad průměrných cen v okrese. Primárním klíčem je této tabulce atribut *id_ceny,* nad kterým je vybudován index. Obsahuje další atributy jako, jsou *datum* a atributy pro uložení průměrných cen jednotlivých typů nemovitostí.

Tabulka Byty a BytyTypy

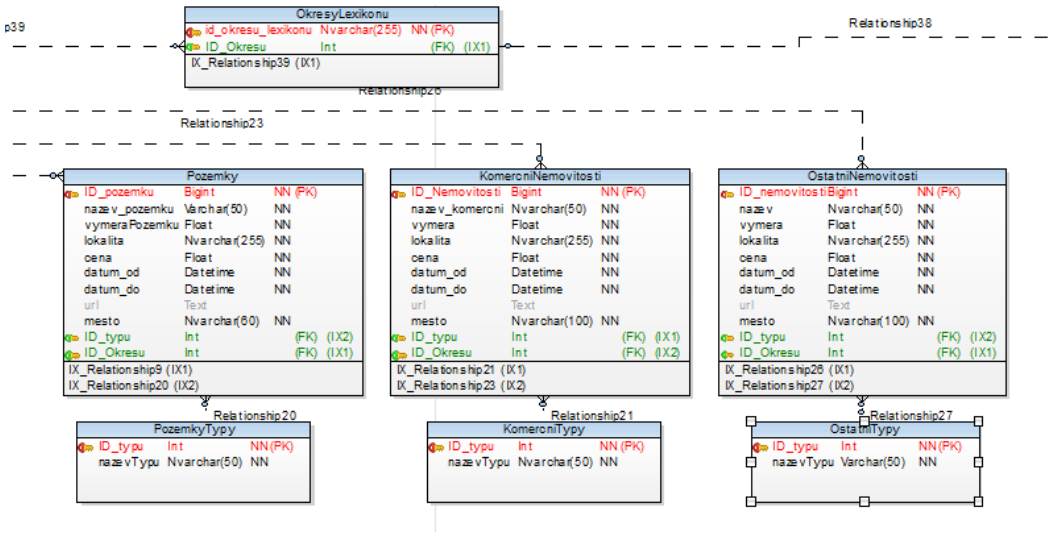
Tabulka byty ukládá rozparsované informace načítaných inzerátů bytů. Atribut *ID_bytu* je primárním klíčem a datový typ tohoto atributu je BigInt, z důvodu toho, že primární klíč se skládá ze součtu id okresu a id inzerátu. Nad tímto atributem je vytvořen index pro ryhlejší vyhledávání záznamů. Dalšími důležitými atributy jsou *vymera, cena, lokalita, datum_do, mesto*. Z těchto atributů jsou načítány informace pro výpočet průměrné ceny za 1 m^2 v dané obci, okresu či kraji.

Tabulka byty typy úzce souvisí s tabulkou byty, slouží jako číselník a tedy uchovává informace o jednotlivých typech bytů. Má pouze dva atributy a to *ID_typu* a *nazevTypu*.

Tabulka Domy a DomyTypy

Tabulka domy uchovává informace z načtených inzerátů domů. Primárním klíčem v této tabulce je atribut *ID_domu* a nad ním je opět vytvořen index pro rychlejší vyhledávání záznamů. Další atributy jsou stejné jako u tabulky bytů, nicméně je přidán atribut *vymeraPozemku*.

Tabulka domy typy představuje číselník jednotlivých typů domů. Na tuto tabulku se odkazuje tabulka domů přes atribut *ID_domu*. Tento atribut je v tabulce domů cizím klíčem. Má opět pouze dva atributy a *ID_typu* a *nazev_typu*.



Obrázek 3.4 - Návrh databáze část 2, zdroj: vlastní

Tabulka Pozemky a PozemkyTypy

Tabulka pozemků ukládá ze serveru potřebné informace z načtených inzerátů pozemků. Primárním klíč v této tabulce je atribut *ID_pozemku* a nad byl ním opět vytvořen index. Atributy jsou více méně podobné, jako jsou u bytů, akorát místo výměry bytu je zde atribut výměry pozemku.

Databázová tabulka pozemky typy slouží opět jako číselník pro uchování jednotlivých typů pozemků. S touto tabulkou má vazbu N:1 tabulka pozemky, kde u této tabulky je atribut *ID_typu* cizím klíčem.

Tabulka KomercniNemovitosti a KomercniTypy

Tabulka komerční nemovitosti slouží k ukládání potřebných informací z inzerátů načtených komerčních nemovitostí. Jako primární klíč byl zvolen atribut *ID_Nemovitosti* a opět je nad tímto atributem vytvořen index z důvodu rychlejšího vyhledávání. Ostatní atributy jsou stejně nazvány jako u tabulky bytů, akorát je atribut *nazev_komercni*.

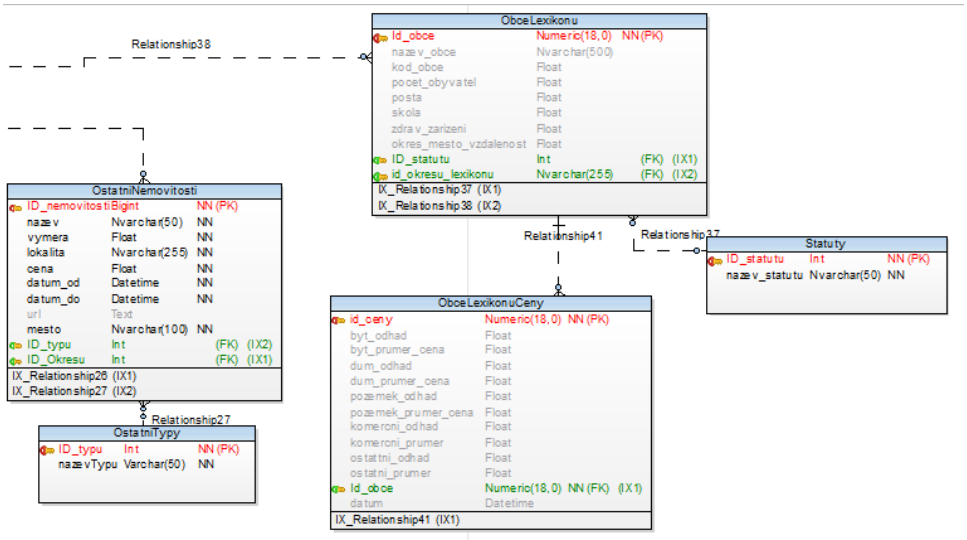
Tabulka OstatniNemovitosti a OstatniTypy

Tabulka ostatní nemovitosti ukládá informace z načtených inzerátů ostatním nemovitostí. Primární klíč je atribut *ID_Nemovitosti* a nad tímto atributem je vytvořen index. Názvy ostatních atributů jsou stejné jako u databázové tabulky byty.

Databázová tabulka ostatní typy slouží opět jako číselník pro tabulku ostatní nemovitosti a je s ním spojena vaznou N:1. Slouží pro ukládání jednotlivých typů ostatních nemovitostí.

Tabulka OkresyLexikonu

Tato tabulka slouží jako tabulka spojující id okresu a *id_okresu_lexikonu*, kde *id_okresu* odpovídá internímu označení okresů na serveru Sreality.cz a kde *id_okresu_lexikonu* odpovídá kódu okresu dle statistického úřadu. Tato tabulka má pak vazbu 1:N s tabulkou obce lexikonu, kde jednotlivým obcím je přiřazen tento kód okresu.



Obrázek 3.5 - Návrh databáze část 3, zdroj: vlastní

Tabulky ObceLexikonu, ObceLexikonuCeny a Statuty

Tyto zbylé tři tabulky ukládají informace z malého lexikonu obcí. Tabulka obce lexikonu uchovává informace načtené ze souboru získaného z webových stránek českého statistického úřadu. Atribut *id_obce* představuje primární klíč databázové tabulky a nad tímto atributem je vytvořen index. Dalšími důležitými atributy této tabulky pro další počítání odhadu ceny jsou *pocet_obyvatel* a *okres_mesto_vzdalenost*. Atributy *kod_obce*, *posta*, *skola*, *zdrav_zarizeni* jsou spíše informativní pro uživatele při zobrazení detailu obce v aplikaci.

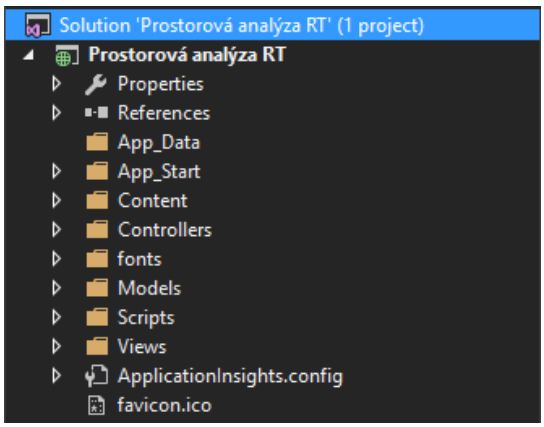
Tabulka statuty slouží jako číselník pro tabulku obce lexikonu. Uchovává v sobě jednotlivé typy obcí a je s tabulkou lexikon obcí vázána vazbou N:1. Obsahuje pouze dva atributy a první atribut je primárním klíčem této tabulky.

Poslední tabulkou databáze je tabulka obce lexikonu ceny. Tato tabulka slouží k ukládání průměrných cen a odhadu cen pro všechny typy nemovitostí a pro konkrétní obec. Primární klíč této tabulky je ID_ceny, nad kterým je vytvořen index.

3.5.2 Adresářová struktura projektu

Tato podkapitola se zabývá vlastní implementací samotné aplikace. Je zde popsáno adresářové dělení projektu ve vývojovém prostředí Microsoft Visual Studio 2015 a také jsou zde ukázány nejdůležitější části aplikace v kódu programovacího jazyka C#. Popsány jsou i problémy, které nastaly při tvorbě aplikace a také, jak tyto problémy byly vyřešeny.

Adresářová struktura vychází ze zvolené architektury ASP .NET MVC. To znamená, že základní členění adresářů je na adresář se všemi kontrolery navržené pro chod aplikace, adresář se všemi modely obsahující potřebné třídy a ADO .NET model (modelové třídy odpovídající databázovým tabulkám), adresář se všemi view a adresář css obsahující kaskádové styly.



Obrázek 3.6 - Adresářová struktura projektu, zdroj: vlastní

Adresář Models

Adresář models v sobě obsahuje třídy programovacího jazyka potřebné pro ukládání dat. Většina těchto tříd je vázána přímo s databází pomocí technologie ADO .NET a zbytek zpravidla slouží pro ukládání dat sloužících pro zobrazení ve view. Při založení nového projektu ASP . NET aplikace jsou v tomto adresáři již vygenerovány tři třídy a to AccountViewModels, ManageViewModels a IdentityModels. Tyto třídy slouží k autorizaci a autentizaci uživatele ve webové aplikaci.

Třídy JsonGoogle a JsonNemovitosti byly vytvořeny pro deserializaci příchozích dat ze serverů a to konkrétně ze serveru Sreality.cz a API rozhraní serveru Google Distance Matrix. Tyto dvě třídy byly vygenerovány pomocí webové stránky *http://json2csharp.com/*. Výhodou této stránky je, že převádí vstupní objekty ve formátu JSON na třídy programovacího jazyka C# a po převedení tyto třídy přesně odpovídají

objektům v JSON včetně jejeich vnořených objektů. Poté stačí jen využít deserializátor a příchozí JSON objekty budou převedeny přímo na třídy. Jak se přímo pracuje s deserializací objektů JSON je popsáno v *kapitole Důležité metody aplikace*

Tato podkapitola popisu samotné aplikace popisuje nejdůležitější části při vývoji. Jsou zde popsány implementace načítání inzerátů ze serveru Sreality.cz, výpočtu podobnosti vývoje ceny v jednotlivých okresech včetně zobrazení, vypočet odhadu ceny za 1 m2 v dané obci, využití nástrojů Google Charts API pro zobrazení dat zakreslených do mapy a zobrazení časové řady.

Načítání inzerátů v aplikaci.

Třída KrajData.cs je vytvořená modelová třída pro uchování informací k zobrazení ve view vrstvě. Obsahuje informace jako průměrnou cenu v kraji, pro jaký datum tato cena byla vypočítána, pak informace o tom, zda cena od posledního vypočtu klesla či vzrostla a také název kraje.

Další třída *KrajeModel.cs* slouží k předání informací pro zobrazení dat ve view. Obsahuje pouze dvě datové struktury zřetězeného lineárního seznamu. První seznam je seznamem krajů, které se zobrazují ve view, druhý seznam slouží pro předání informací k zakreslení do mapy pomocí nástroje Goole Geochart.

Modelová třída *NacteniSreality.cs* slouží podle názvu k načítání inzerátů, deserializaci objektů, parsování potřebných informací ze serveru Sreality.cz a ukládání informací do databáze. Podobnější popis načítání inzerátů lze přečíst v kapitole *Důležité metody aplikace*

Tato podkapitola popisu samotné aplikace popisuje nejdůležitější části při vývoji. Jsou zde popsány implementace načítání inzerátů ze serveru Sreality.cz, výpočtu podobnosti vývoje ceny v jednotlivých okresech včetně zobrazení, vypočet odhadu ceny za 1 m2 v dané obci, využití nástrojů Google Charts API pro zobrazení dat zakreslených do mapy a zobrazení časové řady.

Načítání inzerátů v aplikaci.

Třída Nemovitosti.edmx je modelovou třídou ADO .NET obsahující třídy, které se mapují na jednotlivé tabulky databáze. Tato třída byla vygenerovaná pomocí nástroje vývojového prostředí Visual Studio. Názvy a struktura jednotlivých tříd odpovídá

názvům tabulek vytvořených v databázi a skrz tyto třídy dochází k ukládání či načítání dat.

ObceModel.cs je třída sloužící pro předání informací z kontroleru do view. Obsahuje tři lineární seznamy pro vypsání všech obcí v daném okresu, data pro vykreslení časové řady s využitím nástroje Google Charts a zobrazení seznamu okresů s nejpodobnějším vývojem ceny bytů za 1 m^2.

Pro předání informací do view a následnému zobrazení slouží třída *ObecModel.cs*. Tato třída předává základní informace o konkrétní obci, načtené z lexikonu obcí, jako jsou kód obce, název obce, počet obyvatel, její statut. Dále pak dopočítané informace například vzdálenost od okresního města, směrodatnou odchylku vypočítanou z načtených inzerátů bytů v obci, průměrnou cenu za 1 m^2 u bytů, odhad ceny za 1 m^2 u bytů, název okresu, ve kterém se obec nachází, a lineární seznam všech inzerátů bytů v dané obci.

OkresyDistanceModel.cs je třída sloužící hlavně pro view. Obsahuje id okresu, název okresu, a uloženou Euklidovskou vzdálenost. Tato data jsou pak zobrazována v tabulce, aby uživatel měl přehled, jaký okres má nejpodobnější časovou řadu.

Třída *OkresyModel.cs* slouží k předání informací pro view vrstvu, na níž se zobrazí základní informace o okresu. Uživateli se zobrazí kód okresu, název okresu, nezaměstnanost v okresu a aktuální průměrná cena za 1 m^2 u bytů v okresu.

TimeSeriesData.cs třída obsahuje pouze dva atributy a to datum a hodnotu. Slouží především k předání dat z kontroleru pro view a následného vykreslení časové řady pomocí nástroje Google Charts.

Poslední třída *Vypocty.cs* není čistě modelová třída. Tato třída zajišťuje různé výpočty nad daty předané z databáze. Obsahuje metody pro výpočet průměrné ceny v okrese, výpočet průměrné ceny v obci, výpočtu času dojezdu k okresnímu městu v daném okrese a výpočet odhadu ceny.

Adresář Controllers

Tento adresář obsahuje všechny kontrolery, které se starají o veškerý běh aplikace. Při založení nového projektu ASP .NET MVC ve vývojovém prostředí Miscrosoft Visual Studio se vygenerují tři kontrolery a to *HomeController, AccountController* a *ManageController*. *HomeController* slouží k zobrazení stránky webové aplikace

a zajišťuje její ovládání. *AcountController* a *ManageController* obstarávají veškerá přihlášení uživatelů v aplikaci včetně vypsání chybových hlášek o špatném zadání uživatelského jména a hesla.

Dalším kontroler vytvořený v tomto adresáři je ***StatistikyController***. Tento kontroler se stará o veškeré výpočty dat a jejich předání pro vykreslení na view vrstvě. Obsahuje metody, pomocí nichž dojde k přesměrování z hlavní stránky na ostatní stránky. Metoda *Index* se stará o to co se má stát v případě, že aplikace vyžaduje vykreslení hlavní stránky tohoto kontroleru. V tomto případě se stará o načtení všech krajů z databáze, výpočet průměrných cen za 1 m^2 v jednotlivých krajích a jejich seřazení od nejvyššího data k nejmenšímu. Dále pak, zda tato hodnota od posledního výpočtu vzrostla, poklesla či stagnovala. K předání dat využívá modelovou třídu *KrajeModel.cs*, která byla popsána výše.

Metoda *Details* se vstupním celočíselným parametrem *id* se stará o zpracování detailu konkrétního kraje, dle hodnoty vstupního parametru *id*. Tato metoda předává view načtený seznam okresů ležících v konkrétním kraji z databáze.

ObecDetail metoda s celočíselným parametrem zajišťuje načítání dat s databáze a jejich zpracování pro view, kde se zobrazuje detailní informace konkrétní obce (například kód obce, počet obyvatel, statut obce, vzdálenost od okresního města apod.) načtené z tabulky lexikonu obcí. Dále tato metoda načítá aktuální inzeráty bytů v obci, pokud se v té obci nějaké nachází, vypočítává průměrnou cenu za 1 m^2, směrodatnou odchylku a odhad ceny za 1 m^2 v obci.

3.6 Důležité metody aplikace

Tato podkapitola popisu samotné aplikace popisuje nejdůležitější části při vývoji. Jsou zde popsány implementace načítání inzerátů ze serveru Sreality.cz, výpočtu podobnosti vývoje ceny v jednotlivých okresech včetně zobrazení, vypočet odhadu ceny za 1 m^2 v dané obci, využití nástrojů Google Charts API pro zobrazení dat zakreslených do mapy a zobrazení časové řady.

Načítání inzerátů v aplikaci

V kapitole 2.2.1 bylo popsáno, jak načíst potřebná data ze serveru Sreality.cz a jak tato data zpracovat, když jsou vrácena ve formátu JSON. Tato podkapitola popisuje implementaci samotného načítání, zpracování a ukládání dat do databáze. Nejprve

je popsána metoda *ZiskaniNemovitosti* se vstupním parametrem datového typu string, která zajišťuje získání dat ze serveru.

```
private static HashSet<Estate> ZiskaniNemovitosti(string url)
        {
            using (WebClient webClient = new System.Net.WebClient())
            {
                WebClient n = new WebClient();

                //stažení dat ve formátu json
                var json = n.DownloadString(url);

                //deserializace kořenového objektu
                RootObject root =
JsonConvert.DeserializeObject<RootObject>(json);

                //získání nemovitostí
                Embedded em = root._embedded;

                List<Estate> nemovitosti = em.estates;
                HashSet<Estate> nem = new
HashSet<Estate>(nemovitosti);

                if (nem.Count != 0)
                {
                    return nem;
                }
            }

            return null;
        }
```

Tato metoda využívá třídy WebClient a ta slouží k získání dat pomocí vstupního parametru string v tomto případě vstupní parametr je parametr url, který je předán vstupním parametrem metody. Tento parametr string url odpovídá url adrese API rozhraní serveru Sreality.cz. Jak tato adresa vypadá a jaké má důležité vstupní parametry je popsáno v kapitole *Získání inzerátů ze serveru*. Po zadání tohoto parametru třída WebClient obstará vrácení dat a tato data jsou uložena v proměnné *json*. Tato data jsou v jednom objektu typu string a mají podobu JSON objektů, proto je potřeba tyto objekty deserializovat do tříd, které byly vytvořeny pomocí stránky *http://json2csharp.com/*.

K této deserializaci slouží pomocná třída *JsonConvert* u níž se nastaví v genericitě datový (v tomto případě se jedná o třídu RootObject) a vstupním parametrem bude právě proměnná *json*, kde jsou uložena stažená data. Všechna deserializovaná data jsou uložena v proměnné root a je potřeba získat z atributů této třídy třídu Embeded, která obsahuje lineární seznam všech inzerátů. Proto byla vytvořena proměnná *em* třídy *Embeded*, která získává data z atributů třídy RootObject a ze které se dá získat seznam inzerátů. Z ní

je by vytažen seznam třídy Estate obsahující tyto inzeráty a poté zabalen to datové struktury HasSet a vrácen jako návratový typ této metody.

```
private void ZiskatNemovitostiVOkrese(int typ)
       {
…

while (cisloOkresu < 5011)
            {
                EstateDistrict ed = new EstateDistrict();
                ed.districtNumber = cisloOkresu;
                HashSet<Estate> NemovitostiOkresu = new
HashSet<Estate>();
                do
                {
                    if (cisloOkresu == 47)
                    {
                        break;
                    }
                    string url =
"http://www.sreality.cz/api/cs/v1/estates?category_main_cb=" + typ +
"&category_type_cb=1&locality_district_id=" + cisloOkresu + "&page=" +
cisloStranky + "&per_page=999";// + "&tms=1459772615465";
                    HashSet<Estate> nem = ZiskaniNemovitosti(url);

                    if (nem != null)
                    {
                        NemovitostiOkresu.UnionWith(nem);
                    }
                    else
                    {
                        pokracovat = false;
                    }
                    nem = null;
                    cisloStranky++;

                } while (pokracovat);
ed.estatequeue = new Queue<Estate>(NemovitostiOkresu.ToList());
fronta.Enqueue(ed);
…
}
```

Byla popsána metoda pro deserializaci příchozích objektů a další jejich zpracování ze serveru a proto je potřeba popsat metodu *ZiskatNemovitostiVOkrese*, která přechozí metodu využívá. V kódu nacházející výše je zobrazena část této metody. Tato metoda prochází ve while cyklu všechny okresy dle jejich interního čísla na serveru Sreality.cz. Jejím vstupním parametrem je celočíselná hodnota int představující typ načítané nemovitosti. Tento while cyklus v sobě obsahuje do while cyklus pro procházení jednotlivých stránek inzerátů v okrese. Pokud bude číslo okresu 47, přeskočí se jedna iterace cyklu z důvodu toho, že tento okres neobsahuje žádné inzeráty.

Dále se v tomto do while cyklu nastavuje do proměnné typu string url adresa, kde se nastavuje z příchozího parametru metody id typu načítané nemovitosti okresu

a nastavuje se zde také id okresu, které se iteračně zvyšuje pomocí while cyklu. Tato proměnná je předána metodě *ZiskatNemovitostiVOkrese* a následně tato metoda vrací načtené inzeráty. Tyto inzeráty jsou vkládány datové struktury HashSet *NemovitostiOkresu* a po skončení do while cyklu vkládány do třídy EstateDistrict ed. Tato třída je pak vložena do fronty, kde pomocí této fronty dochází v metodě *UlozitBytyDoDB* k uložení do databáze. Důležitou roli v této metodě hraje třída EstateDistrict, protože tato třída v sobě uchovává, jednotlivé inzeráty a id okresu, podle kterého se pak inzeráty přiřazují k daným okresům.

Ze serveru by dle požadavků šly načíst všechny inzeráty nemovitostí, problémem je to, že jednotlivé inzeráty ve své struktuře neobsahují, k jakému okresu patří, proto byla vytvořena třída EstadeDistrict a ta v sobě tyto informace ukládá. Dále pak byly zvoleny datové struktury jako HashSet a fronta a to z toho důvodu, že při vývoji načítání inzerátů trvalo v řádek hodin. Proto to vypadalo, že problém bude v přístupech k jednotlivým prvkům datových struktur, protože předtím byla zvolena pro všechna načítání inzerátů datová struktura lineární seznam. Nicméně nakonec bylo zjištěno, že problém je jinde a to je popsáno v metodě *UlozitBytyDoDB*.

```
private void UlozitBytyDoDB(Queue<EstateDistrict> NemVOkrese){
…
Estate Byt = okres.estatequeue.Dequeue();
Int64 IDBytu = Convert.ToInt64(okres.districtNumber +
Byt.hash_id.ToString());
//zjištění, zda již daný byt je v DB
Byty NemByt = DBByty.Find(estate => estate.ID_Bytu == IDBytu);
         if (NemByt == null)
         {
… //Nastavení parametrů pro byt

            if (!BytyID.Contains(NemByt.ID_Bytu))
            {
                  KolekkceNovychBytu.Add(NemByt);
                  BytyID.Add(NemByt.ID_Bytu);
            }
         }
            else
            {
                  NemByt.cena = Byt.price;
                  NemByt.datum_do = DateTime.Now;
                  KolekceBytuUpdate.Add(NemByt);
             }
…
      if (KolekceNovychBytu.Count != 0)
                {
                    //db.Byty.AddRange(KolekceBytu);
                    db.BulkInsert(KolekceNovychBytu);
                }
                if (KolekceBytuUpdate.Count != 0)
                {
```

```
                db.BulkUpdate(KolekceBytuUpdate);
            }
            db.BulkSaveChanges();
    }
```

Tato metoda odebírá z fronty třídy EstateDistrict prvky, v tomto případě je prvek uložen v proměnné okres je a z nich poté potřebné informace předává třídě určitého typu nemovitosti. Je zde popsána pouze metoda *UlozitBytyDoDB*, protože ostatní metody *UlozitDomyDoDB, UlozitPozemkyDoDB, UlozitKomerniDoDB, UlozitOstatniDoDB* fungují stejně, jen jsou přizpůsobeny určitému typu nemovitosti.Po odebrání prvku z fronty třídy EstateDistrict odebere z fronty této třídy další prvek, což už je instance třídy *Estate* s názvem *byt*. Z této instance je vytaženo id konkrétního inzerátu. Toto id je ve formátu string, ale obsahuje pouze čísla.

Proto bylo potřeba toto id převézt na číslo. Jelikož toto id po převedení na číslo přesahovalo rozsah datového typu int, bylo zvoleno převedení na datový typ Int64. K tomu to číslu se připočetlo i číslo okresu z toho důvodu, protože se stávalo, že toto id nebylo unikátní a provedlo se pomocí LINQ prohledání tabulky inzerátů bytů v databázi, zda tento inzerát se v této tabulce nachází. Pokud se inzerát v databázi nachází, je mu aktualizována cena, datum a následně je přidán do lineárního seznamu *kolekceBytuUpdate*. Pokud se inzerát v databázi nenchází je vytvořena nová instance třídy Byty, kde jsou jí nastaveny všechny jeho atributy této třídy. Poté je tato nová instance vložena do lineárního seznamu kolekceNovychBytu. Jakmile doběhne odebrání všech prvků ze vstupní fronty této metody, inzeráty se pomocí knihovny *Z.BulkOperations* uloží do databáze pomocí metod *bulkInsert* a *bulkUpdate.*,

Pro právě byla zvolena tato knihovna? Důvod je ten, že umožnuje rychlejší ukládání, upravení záznamů v tabulce databáze. Kdyby se měla pokaždé nová instance objektu ukládat do databáze, vrostlo by ukládání na několik hodin. Protože uložení jednoho záznamu bez této knihovny trvá třeba 25 ms a při 100 000 načtených inzerátech toto zrovna není efektivní ukládání dat. Tabulka doby trvání jednotlivých metod této knihovny je popsána v kapitole *Knihovna Z.BulkOperations.*

Využití Google charts

V této části je popsáno, jak se využil nástroj Google Charts. V aplikaci jsou použity dva typy tohoto nástroje a to Google Chart a Google Geochart. Tyto dva nástroje jsou

implementovány až ve view na konkrétní stránce. Jedná se o zápis JavaScriptu, který je umístěn v html kódu v tazích <script></script>.

```
google.load('visualization', '1', { packages: ['geochart'] });

    function drawVisualization() {
        var data = new google.visualization.DataTable();
        data.addColumn('string', 'Province', 'State');
        data.addColumn('number', 'rank', 'rank');
        data.addColumn({ type: 'string', label: 'branch', id:
'branch', role: 'tooltip' });

        var data1 = @Html.Raw(Json.Encode(Model.mapaKrajuData));

        $.each(data1, function(index, value){
            data.addRows([[value.nazevKraje,
value.prumernaCena,'datum: '+value.datum +'\nprůměrná cena: '
                + value.prumernaCena + '\n'+ value.info + '\nPředchozí
průměrná cena: ' + value.predchoziCena+
                '\nRelativní přírůstek: ' +
value.relativniPrirustek]])
        });
        var geochart = new
google.visualization.GeoChart(document.getElementById('visualization')
);
        var options = {
            chart: {
                title: 'Průměrná cena za m2 v okrese',
                subtitle: 'v Kč'
            }
        };
        options['region'] = 'CZ';
        options['resolution'] = 'provinces';

        options['colorAxis'] = { minValue: 0, colors: ['#DCDCDC',
'#6495ED', '#4169E1', '#0000FF'] }

        geochart.draw(data, options);
    }

  google.setOnLoadCallback(drawVisualization);
```

V kódu napsaném výše je popsáno jak vytvořit mapku se načtenými daty. Tato mapa je vykreslena na stránce *Statistiky* a pod ní je vypsán seznam všech krajů ČR. Je v něm externě volána funkce google load a dle nastavených parametrů se volá balíček geochart. Ve funkci drawVisualization se vše nastavuje. Nejprve bylo potřeba si vytvořit novou instanci DataTable a té poté přidat názvy jednotlivých sloupců a nastvit jim také jakého budou datového typu. První sloupec je typu string a musí u něj být nastaveny parametry *Province* a *State*. U druhého je nastaven datový typ number a jsou u něj nataveny parametry rank, podle kterých dojde k vybarvení určitého kraje, dle hodnoty. Poslední sloupeček slouží zobrazení informací po najetí kurzoru myši na konkrétní kraj.

Z modelové třídy *KrajeModel*, nesoucí data z kontroleru, je vytažen lineární seznam *mapaKrajuData*. Tento seznam je převeden pomocí třídy C# třídy Json na objekty typu JSON a následně pak pomocí foreach cyklu programovacího jazyka JavaScript přidány do řádku DataTable. Aby došlo k přesnému vykreslení k danému kraji, musí do prvního sloupečku být vložen přesný název kraje, o zbytek se už postará samotný nástroj od Google Geochart.

Proměná var geochart zajišťuje, kam se má výsledný graf vykreslit. Volá metodu getElementById s názvem id. Toto id může být jedinečné v jedné HTML stránce a v tomto případě se vykreslí do HTML elementu s tímto id. Proměnná *var options* pomocí JSON objektu nastavuje název a podnázev grafu. Dále pomocí nastavení parametru *options[‚region‘]* se nastavuje CZ, z důvodu vykreslí mapky České republiky a pomocí parametru options[‚resolution‘] s nastavením provinces se nastavuje územní rozdělení na kraje. Poslední parametr, který byl nastaven je parametr options[‚colorAxis‘]. Pomocí tohoto objektu se v objektu JavaScriptu nastavuje rozsah barev pro vykreslení.

Obrázek 3.7 - Mapa průměrných cen podle krajů ČR v aplikaci, zdroj: vlastní

```
google.charts.load('current', { 'packages': ['line'] });
    google.charts.setOnLoadCallback(drawChart);

    function drawChart() {

        var data = new google.visualization.DataTable();
        data.addColumn('string', 'Datum');
        data.addColumn('number', 'Byty');

        var data1 = @Html.Raw(Json.Encode(Model.casovaRada));
        $.each(data1, function(index, value){
            data.addRows([[value.datum, value.hodnota]])
        });

        var options = {
            colors: ['#e2431e', '#d3362d', '#e7711b','#e49307',
'#e49307', '#b9c246'],
            lineWidth: 6,
            width: 900,
            height: 500
        };
        var formatter = new google.visualization.NumberFormat(
        {negativeColor: 'red', negativeParens: true, pattern:
'######.## Kč'});
        formatter.format(data, 1);
```

```
        var chart = new
google.charts.Line(document.getElementById('linechart_material'));

        chart.draw(data, options);
```

Kód uvedený výše představuje vykreslení časové řady na webové stránce. Představuje konkrétní nastavení pro nástroj Google Chart a jak již bylo zmíněno nastavení je psáno v programovacím jazyce JavaScript. Je volána, stejně jako v předchozím kódu, metoda load a pomocí objektu ve vstupním parametru je volán balíček parametrem *line*. Poté bylo potřeba opět vytvořit funkci *drawChart*, ve které se provádí veškerá nastavení. Opět je zde potřeba vytvořit proměnnou data a v ní si držet instanci objektu DataTable. Této tabulce byly nastaveny dva sloupce a to sloupec s datovým typem string *Datum* a sloupec s datovým typem number *Byty*.

Poté bylo potřeba naplnit tabulku řádky, představující data pro vykreslení. Tato data jsou předána z kontroleru pomocí modelové třídy *ObceModel* a z ní byl vytažen lineární seznam casovaRada. Stejně jako v předchozím kódu tato data byla převedena na JSON objekty a poomocí foreach cyklu vložena do datové tabulky. V nastavení options se nastavily barvy pro graf a nastavila se jeho šířka a výška pro vykreslení na stránce. Důležitá je proměnná *var formatter*, která zajišťuje vypsání hodnoty ceny na dvě desetinná čísla. Pomocí metody getElementById s parametrem ‚linechart_material' na stránce vykreslena do divu s tímto id. Vzniklá časová řada se poté vykresluje na stránce konkrétního okresu například pro okres Benešov, jak je možné vidět na obrázku Obrázek 3.8.

Obrázek 3.8 - Časová řada okresu Benešov - obrázek z aplikace, zdroj: vlastní

Výpočet odhadu ceny za 1 m^2 v aplikaci

V této části práce je popsáno, jak byla implementována část metody pro odhad ceny v jednotlivé obci. Je zde popsána hlavně nejhlavnější část metod výpočtu odhadu.

```
  var ObceOkresuCeny = (
            from okres in okresy
            join okresLex in okresyLexikonu on okres.ID_Okresu equals
okresLex.ID_Okresu
            join obec in obce on okresLex.id_okresu_lexikonu equals
obec.id_okresu_lexikonu
            join obecCena in obceCeny on obec.Id_obce equals
obecCena.Id_obce
            where obecCena.byt_prumer_cena > 0 &&
obecCena.byt_prumer_cena != null
```

```
            select new {pocetObyvatel = obec.pocet_obyvatel,
vzdalenost = obec.okres_mesto_vzdalenost,
                nezam = okres.nezamestnanost,
                prumCena = obecCena.byt_prumer_cena,
            okresID = okres.ID_Okresu}).ToList();

            foreach(var okres in okresy)
            {
                var obceCena = (from obec in ObceOkresuCeny
                                where obec.okresID == okres.ID_Okresu
                                select obec).ToList();
                if(obceCena.Count != 0) {

                Matrix<double> X =
Matrix<double>.Build.Dense(obceCena.Count, 3, 1.0);
                Matrix<double> Y =
Matrix<double>.Build.Dense(obceCena.Count, 1, 1.0);

                for (int i = 0; i < obceCena.Count; i++)
                {
                    double[] radekX = new double[3];
                    double[] radekY = new double[1];
                    var obec = obceCena[i];
                    radekX[0] = 1;
                    radekX[1] = (double)obec.pocetObyvatel;
                    radekX[2] = (double)obec.vzdalenost;
                    radekY[0] = (double)obec.prumCena;
                    X.SetRow(i, radekX);
                    Y.SetRow(i, radekY);

                }
…
}
```

Nejprve se provede načtení všech obcí, u kterých jsou inzeráty a je z nich vypočítána průměrná hodnota za 1 m^2. Tyto obce jsou vloženy do proměnné typu *var ObceOkresuCeny*. Poté jsou ve foreach cyklu procházeny postupně jednotlivé okresy a pomocí jejich id jsou do proměnné *var obceCena* ukládány obce procházeného okresu. Pokud tato proměnná obsahuje záznamy, vkládají se do matic X a Y jednotlivé faktory pro výpočet odhadu ceny. Zde je využita knihovna Math.Net.Numerics pro operace nad maticemi. Do matice X jsou vkládány na jeden řádek 1, počet obyvatel obce a vzdálenost. Do matice Y průměrná cena za 1 m^2 inzerátů v obci.

Jsou-li matice naplněny všemi obcemi, které mají inzeráty, provedou se maticové operace přesně podle vztahu, který je definován v kapitole *Regresní analýza* a v programu je to zapsáno tímto způsobem.

```
   Matrix<double> maticeXTrans = X.Transpose();
                Matrix<double> preInvert = maticeXTrans.Multiply(X);
                Matrix<double> invert = preInvert.Inverse();
                Matrix<double> mult = invert.Multiply(maticeXTrans);
                Matrix<double> beta = mult.Multiply(Y);
```

Matice beta obsahuje vypočtené parametry a díky těmto parametrům se dosadí parametry konkrétní obce a vypočte se odhad ceny za 1 m^2. Takže jsou poté zpětné procházeny obce daného okresu a je u nich počítán odhad. Jak je to zobrazeno v aplikaci je možní vidět na obrázku *Obrázek 3.9*.

Původně při výpočtu odhadu ceny bylo použity tři parametry a to počet obyvatel, vzdálenost od okresního města a nezaměstnanost v okres. Právě nezaměstnanost v okrese byla z výpočtu odebrána, protože po dosazení do matice následné její transformaci došlo k lineární kombinaci některých řádků. To mělo za důsledek to, že determinant takovéto matice byl nula a nešlo sestrojit inverzní matici k matici X a tedy dopočítat parametry pro odhad ceny v dané obci.

Obrázek 3.9 - Odhad ceny v obci Mličín, zdroj: vlastní

Výpočet podobnosti okresů v aplikaci

Metoda výpočet podobnosti okresů se stará o výpočet podobnosti ve vývoji ceny mezi okresy. Její výsledky jsou zobrazeny ve view u konkrétního okresu v tabulce, kde tyto vzdálenosti mezi jednotlivými okresy jsou zobrazeny v tabulce a seřazeny

od nejpodobnějšího okresu po nejvzdálenější. K výpočtu je využita Euklidovská vzdálenost a to konkrétně vztah z kapitoly ***Chyba! Nenalezen zdroj odkazů.***.

```
            for (int i = 0; i < okresy.Count; i++)
            {
                Okresy okresY = okresy[i];
                OkresyLexikonu okresL = okresyLexikonu.Find(x =>
x.ID_Okresu == okresY.ID_Okresu);

                if (okresY.ID_Okresu == 47)
                {
                    continue;
                }

                var okresYCeny = (
                from okres in okresy
                join ceny in okresyCeny on okres.ID_Okresu equals
ceny.ID_Okresu
                join okrLex in okresyLexikonu on okres.ID_Okresu
equals okrLex.ID_Okresu
                where okres.ID_Okresu == okresY.ID_Okresu
                orderby ceny.datum ascending
                select new { Cena = ceny.avgbyty }).ToList();

                double vzdalenost = 0;
```

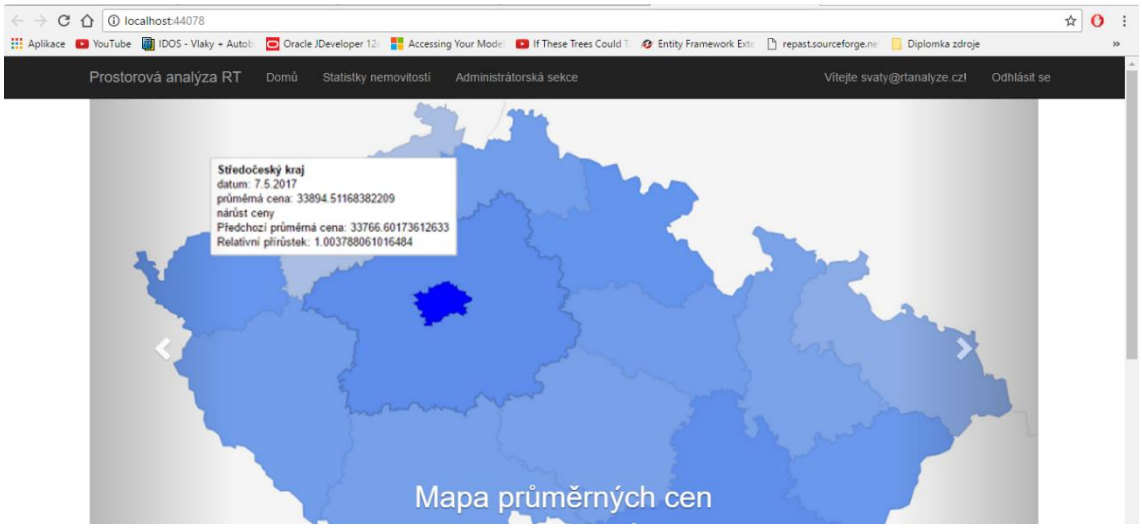
```
                for (int k = 0; k < okresYCeny.Count; k++)
                {
                    double cenaX = (double)okresXCeny[k].Cena;
                    double cenaY = (double)okresYCeny[k].Cena;
                    vzdalenost += Math.Pow((cenaX - cenaY), 2);

                }
                vzdalenost = Math.Sqrt(vzdalenost);
                OkresDistanceModel ov = new
OkresDistanceModel(okresL.id_okresu_lexikonu, okresY.nazev_okresu,
vzdalenost);
                okrVzdal.Add(ov);

            }
```

V kódu výše je popsáno, jak tato metoda byla implementována. Nejprve jsou do proměnné *okresXCeny* načteny veškeré průměrné ceny za 1 m^2 z databázové tabulky *OkresyCeny* uchovávající data pro vykreslení časových řad. Tato data jsou seřazena dle data od nejstaršího po nejnovější. Poté jsou ve for cyklu procházeny veškeré okresy a k těmto okresům jsou načítána stejným způsobem data plus k nim jsou načítána id z tabulky *OkresyLexikonu* (z důvodu přímého odkazu z tabulky na konkrétní okres ve view) a ukládána do proměnné *okresYCeny*.

Ve vnitřním for cyklu jsou tato data počítána dle vztahu z kapitoly ***Chyba! Nenalezen zdroj odkazů.***. Vypočítaná Euklidovská vzdálenost, název okresu, pro který byla vypočítána a id_okresu_lexikonu jsou uložena do nové instance třídy OkresDistanceModel a lineárním seznamu předávána do view, kde jsou vykreslena do tabulky.

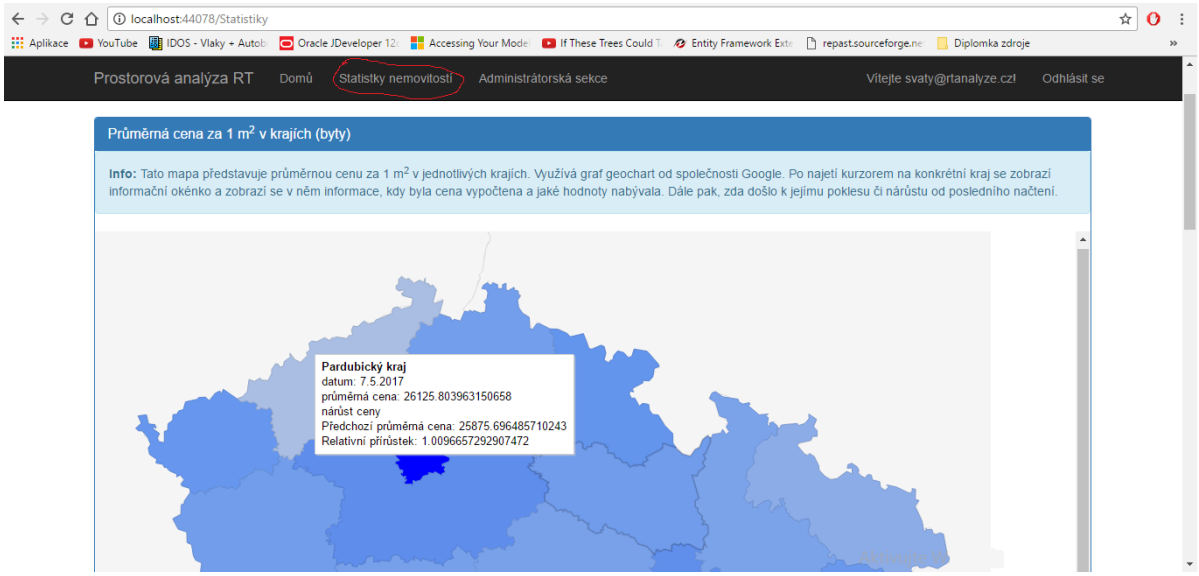
3.7 Grafické rozhraní aplikace

Tato podkapitola popisu aplikace se zabývá ukázkami vyvinuté aplikace a popisu webového grafického rozhraní. Byla vyvinuta pomocí technologie ASP .NET, programovacího jazyka C# a Entity Frameworku. Dále tato aplikace využívá navržené databáze v MSSQL pro ukládání potřebných dat. Aplikace funguje pro dva typy uživatelů a to pro nepřihlášené a administrátory. Nepřihlášení uživatelé mohou pouze prohlížet výsledná data, případně je stahovat do CSV souboru a také stahovat data pro GIS systémy. Administrátoři spravují veškeré číselníky aplikace (okresy, kraje, typy jednotlivých nemovitostí) a také mají oprávnění pro načítání inzerátů ze serveru Sreality.cz.



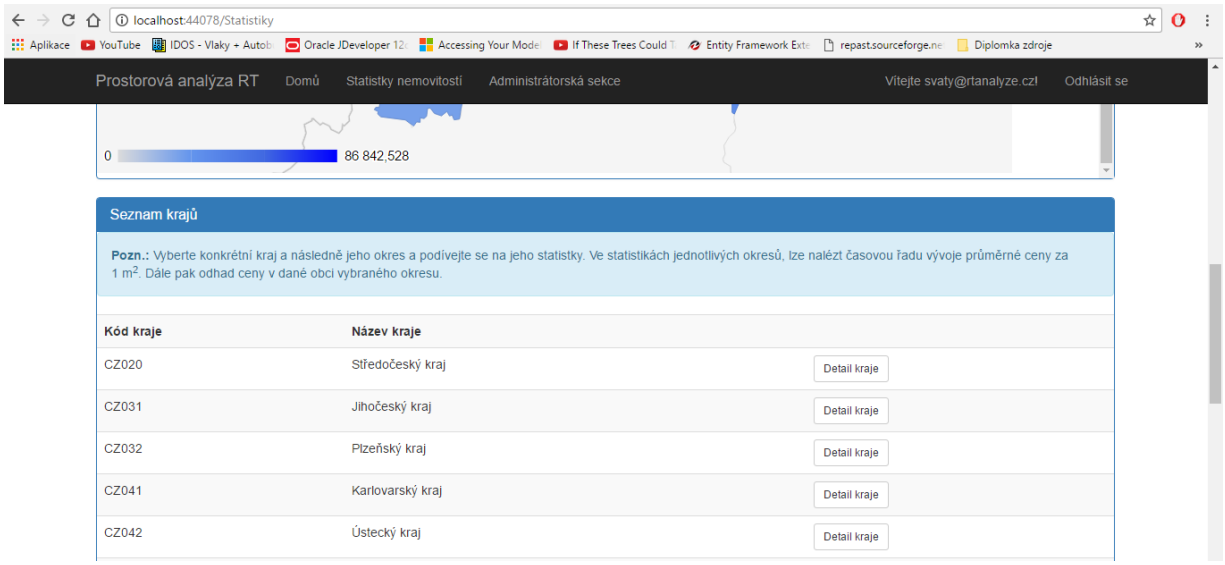
Obrázek 3.10 - Úvodní stránka webové aplikace, zdroj: vlastní

Na obrázku Obrázek 3.10 je vidět úvodní stránka aplikace. Tato stránka zobrazuje úvod a lehký popis aplikace. Na stránce je umístěna komponenta carousell, která zajišťuje střídání obrázků na úvodu. Střídají se obrázky mapy s průměrnými cenami, dále pak obrázek se zobrazeným grafem vývoje ceny za 1 m^2 a obrázek zobrazující informace o obci, obsahující základní informace plus vypočtený odhad ceny za 1 m^2 pro byty v obci. Na tomto obrázku je vidět navigační lištu, kde jsou pouze dvě položky a to *Statistiky nemovitostí* a *Administrátorská sekce* a vpravo pak přihlášeného uživatele. Nejprve bude rozebrána záložka *Statistiky nemovitostí*.



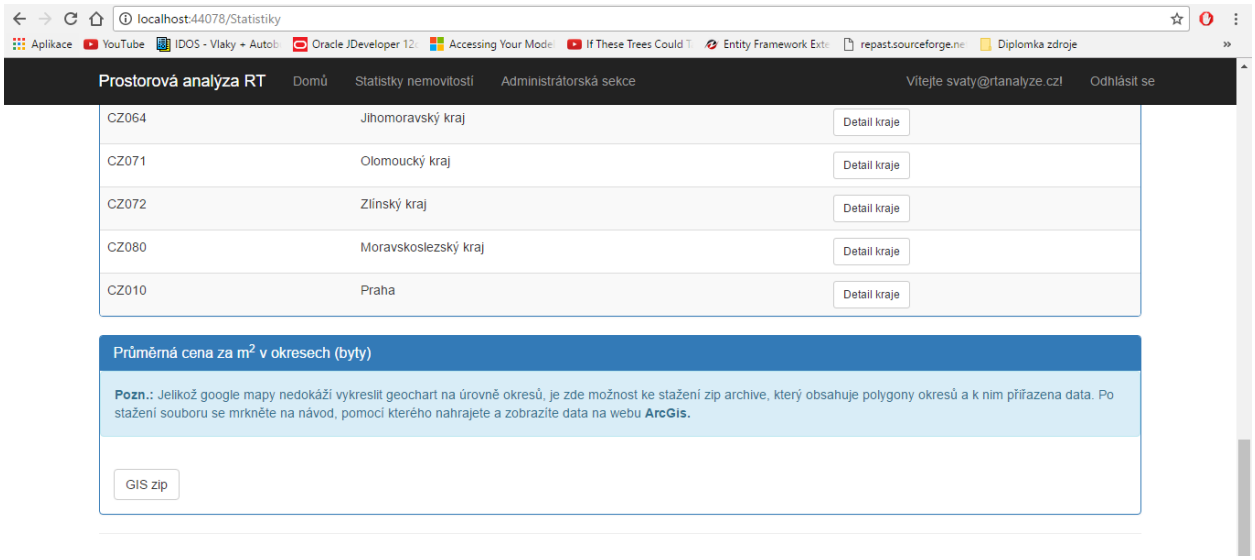
Obrázek 3.11 - Ukázka rozhraní aplikace, stránka Statistiky nemovitostí, zdroj: vlastní

Na obrázku Obrázek 3.11 je vidět první část stránky *Statistiky nemovitostí*. Je zde vidět vykreslená mapa pomocí nástroje Google Geochart, která v sobě nese informace o průměrné ceně za 1 m^2 v kraji, jakého data tato cena byla vypočítána, zda došlo k jejímu poklesu vzrůstu či stagnaci a také jaký byl relativní přírůstek této ceny. Dále na tomto jsou zachyceny informace konkrétně pro Pardubický kraj.



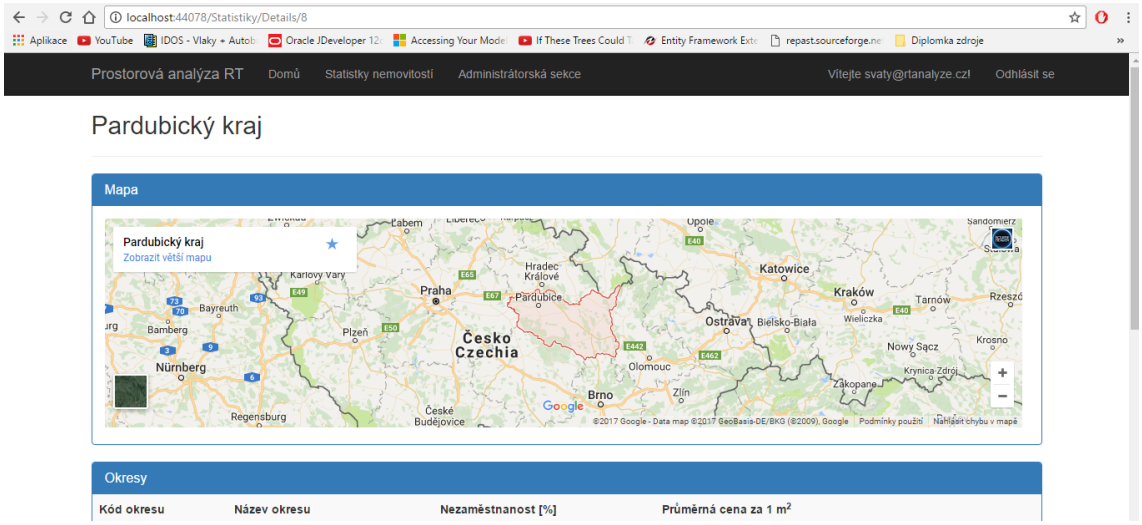
Obrázek 3.12 - Ukázka rozhraní aplikace, stránka Statistiky nemovitostí 2, zdroj: vlastní

Na obrázku Obrázek 3.12 je vidět druhá část stránky, která zobrazuje do tabulky seznam krajů. U každého kraje se dá zobrazit detail, ale tomuto detailu bude věnována pozornost později.

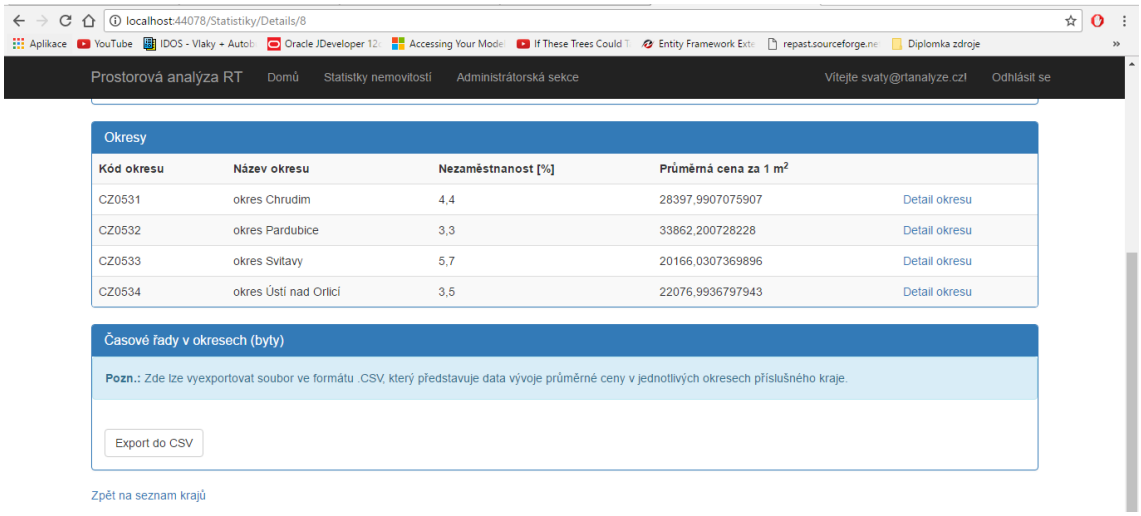


Obrázek 3.13 - Ukázka rozhraní aplikace, stránka Statistiky nemovitostí 3, zdroj: vlastní

Obrázek 3.13 zobrazuje poslední, třetí část, stránky, kde je možné vidět panel *Průměrná cena za m² v okresech (byty)*. V tomto panelu je tlačítko, které slouží ke stažení dat pro GIS systémy a zobrazení vypočtených aktuálních cen za 1 m² v jednotlivých okresech ČR.



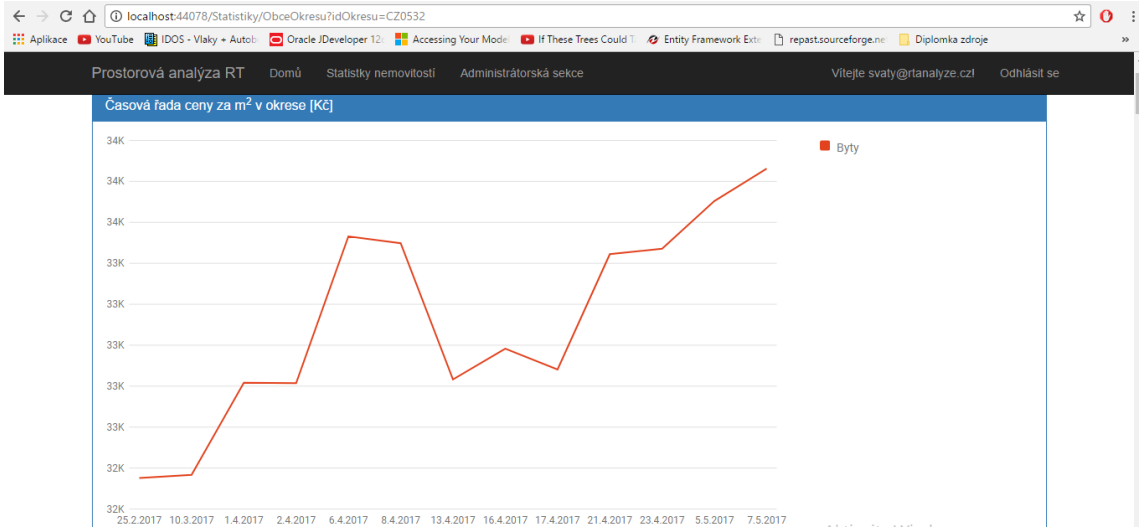
Obrázek 3.14 - Ukázka aplikace, stránka Pardubický kraj, zdroj: vlastní



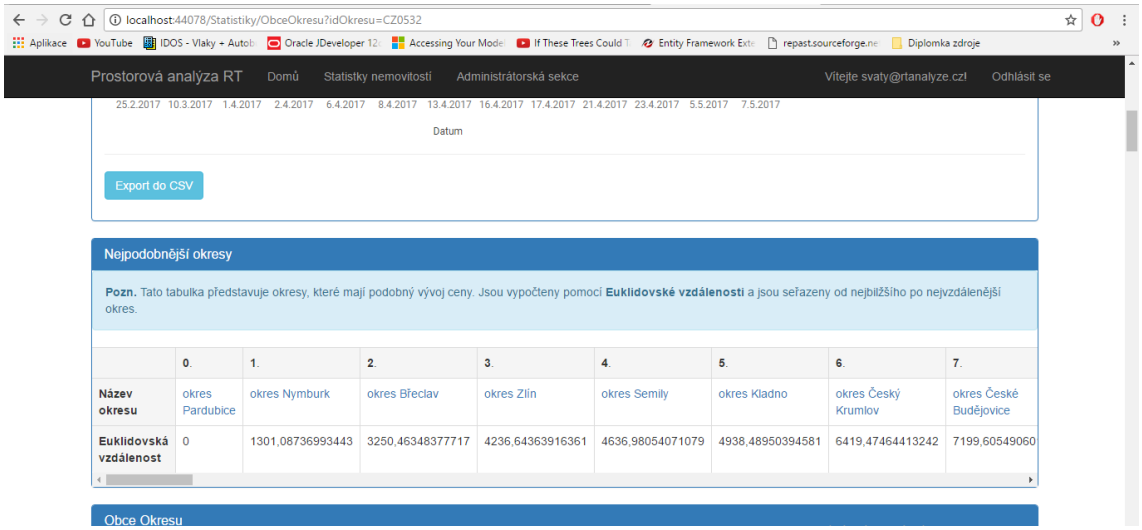
Obrázek 3.15 - Ukázka aplikace, stránka Pardubický kraj, zdroj: vlastní

Na obrázcích Obrázek 3.14 a Obrázek 3.15 je zobrazen detail Pardubického kraje. Tento kraj je zobrazen na mapě pomocí Google nástroje Embeded Map a je umístěn spíše pro efekt, nicméně může alespoň uživateli ukázat, kde tento kraj leží. Tato stránka kromě této mapy obsahuje dva panely. První panel zobrazuje v tabulce okresy konkrétního kraje a u nich informace jako jsou kód okresu (dle souřadnic NUTS3), název okresu a aktuální vypočtenou průměrnou cenu za 1 m² u bytů a odkaz pro zobrazení detailu okresu. Detail okresu bude popsán později.

Na této stránce si uživatel může stáhnout do CSV souboru vývoje cen u všech okresů, v tomto případě u všech okresů pardubického kraje, a může dále s těmito daty pracovat dle vlastního uvážení.



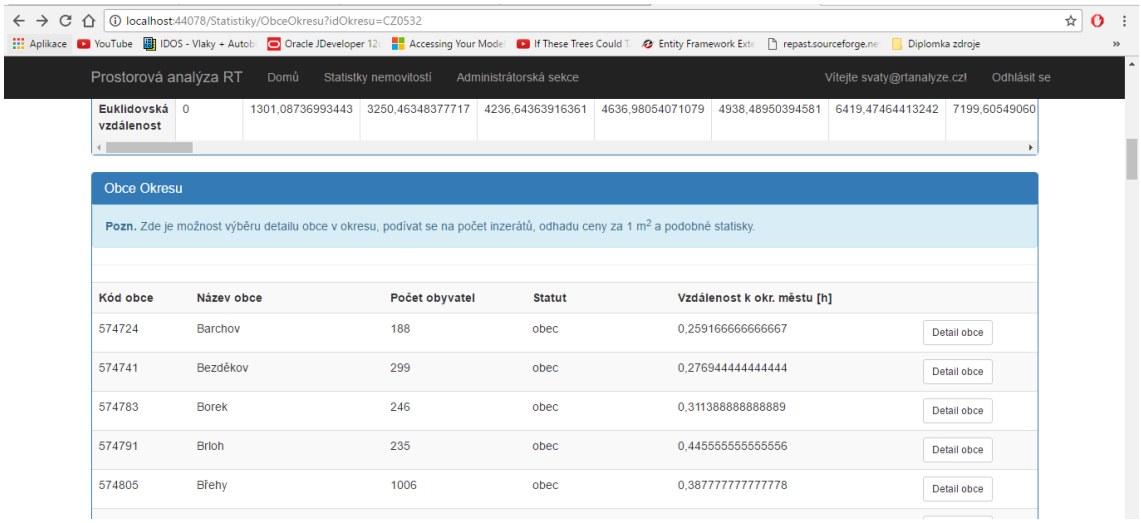
Obrázek 3.16 - Detail okresu Pardubice v aplikaci 1, zdroj: vlastní



Obrázek 3.17 - Detail okresu Pardubice v aplikaci 2, zdroj: vlastní

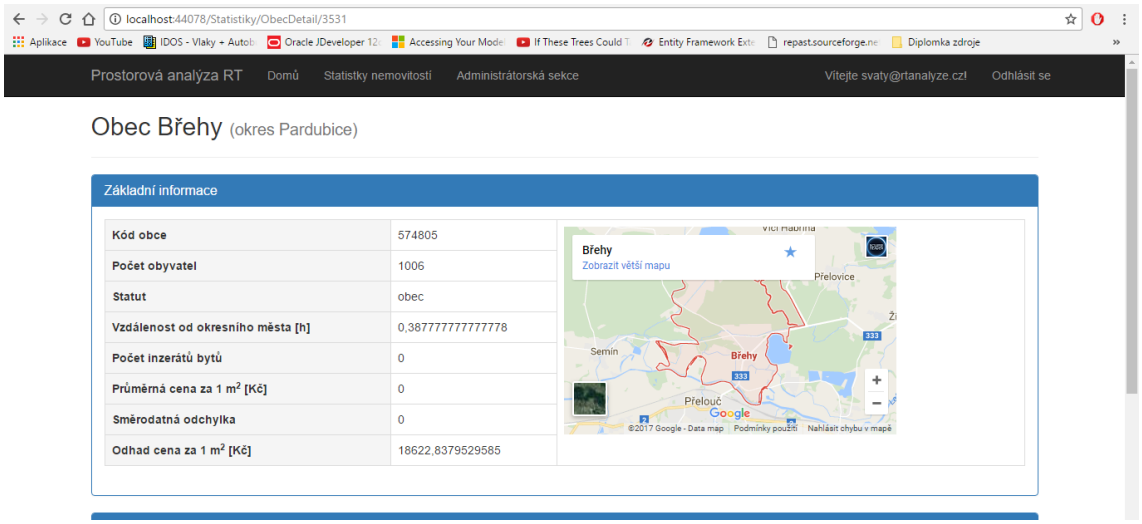
Obrázky Obrázek 3.16 a Obrázek 3.17 zobrazují detailní informace o konkrétním okrese v aplikaci, v tomto případě pro okres Pardubice. Je zde zobrazen vývoj průměrné cenu za 1 m^2 v tomto okrese a možnosti stažení vykreslených dat do CSV souboru. Dále stránka *detail okresu* zobrazuje tabulku nejpodobnějších okresů (včetně částí Prahy)

a v ní jsou seřazeny okresy od nejvíce podobného po neméně. Lze i kliknout na název okresu v tabulce a dojde k přesměrování na detail konkrétního okresu.

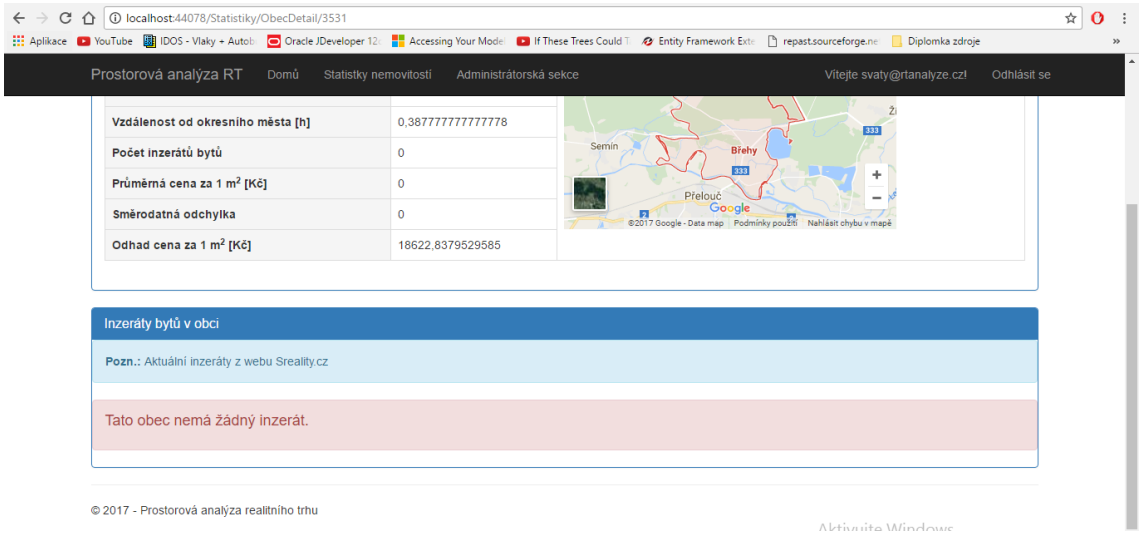


Obrázek 3.18 - Detail okresu Pardubice v aplikaci 3, zdroj: vlastní

Obrázek 3.18 zobrazuje v tabulce seznam obcí daného okresu a v této tabulce jsou zobrazeny základní informace o obci. Těmito informacemi jsou kód obce, název obce, počet obyvatel, statut a dojezd k okresnímu městu. Dále pak je možnost kliknutí na detail konkrétní obce a zobrazit detailní informace.

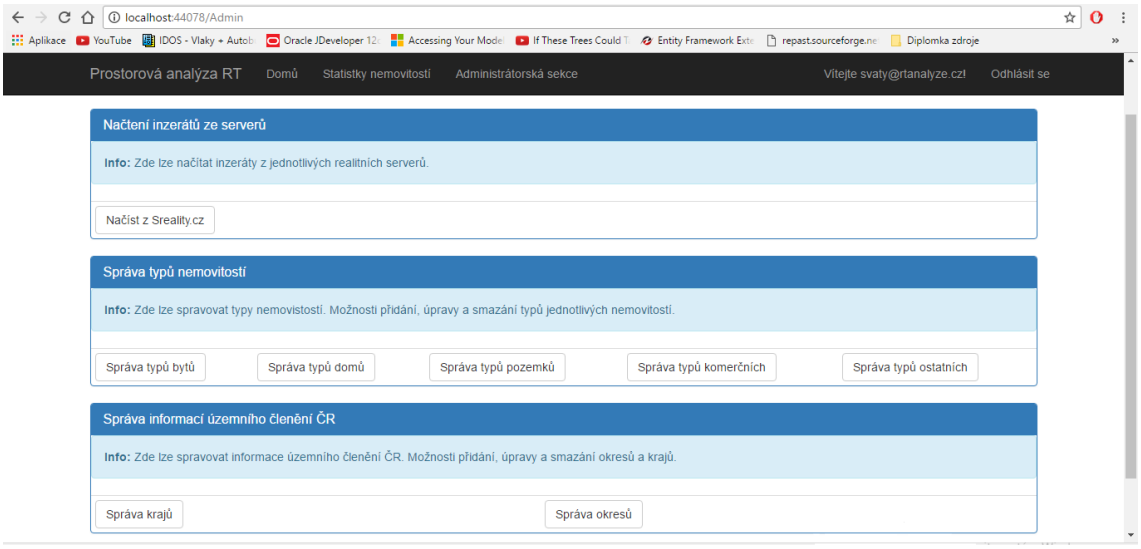


Obrázek 3.19 - Detail obce Břehy v aplikaci 1, zdroj: vlastní



Obrázek 3.20 - Detail obce Břehy v aplikaci, zdroj: vlastní

Na obrázcích Obrázek 3.19 a Obrázek 3.20 je vidět detail obce konkrétně detail obce Břehy, kde se zobrazují základní informace o obci. Kromě těchto informací se zobrazuje počet inzerátů v obci, průměrnou cenu za 1 m^2 vypočtenou z inzerátů bytů konkrétní obce, směrodatnou odchylku a odhad průměrné ceny za 1 m^2 pro danou obec. Dále pak se na této stránce zobrazují načtené inzeráty bytů ze serveru Sreality.cz. V tomto konkrétním případě obec Břehy žádné inzeráty bytů neobsahuje.



Obrázek 3.21 - Stránka administrátorská sekce, zdroj: vlastní

Na obrázku Obrázek 3.21 lze vidět stránku *Administrátorská sekce*, na které jsou tři panely a to panel *Načtení inzerátů ze serverů*, *Správa typů nemovitostí* a *Správa informací územního členění ČR*. Panel *Načtení inzerátů ze serverů* obsahuje tlačítko a po jeho spuštění dojde k načtení inzerátů typů nemovitostí ze serveru Sreality.cz. U panelu

Správa typů nemovitostí lze kliknout na správu jednotlivých typů nemovitostí, čili lze přidávat, upravovat a mazat číselníky jednotlivých typů nemovitostí. Poslední panel *Správa informací územního členění* slouží k přidání, úpravě a smazání informací u jednotlivých okresů či krajů ČR.

4 ZÁVĚR

Cílem práce bylo vytvořit aplikaci, která měla být schopna načítat inzeráty z realitního serveru. Nad těmito inzeráty provádět prostorové analýzy veřejně dostupných dat ekonomických subjektů České republiky vycházejících z Českého statistického úřadu. Dále vytvořit sadu časových řad zobrazující vývoj cen nemovitostí ve zvolených regionech ČR. Cílem práce také byla operace geokódování, kde k jednotlivým prvkům se přiřadí prostorové souřadnice a výpočet odhadu ceny nemovitostí v jednotlivých oblastech České republiky. Využít tak výpočtu faktorový model a vhodné vysvětlující proměnné (počet obyvatel, vzdálenost k okresnímu městu, nezaměstnanost v okrese).

Byla tedy vytvořena aplikace umožňující uživatelům zobrazit na mapě ČR průměrnou cenu za 1 m^2 v jednotlivých krajích, vypočtených z parsování webových stránek. Zde bylo zvoleno efektivnějšího načítání, a to načítání pomocí dotazů odesílajících se na API rozhraní serveru Sreality.cz. K vytvořené aplikaci byla navržena databáze uchovávající inzeráty jednotlivých typů nemovitostí a potřebných dat (například lexikon obcí, nezaměstnanosti v okrese) k výpočtu odhadu ceny.

Aplikace také umí zobrazit uživateli aktuální průměrnou cenu v okrese pro byty a její historický vývoj pomocí časové řady. K vykreslení jak mapy ČR, časové řady a výpočtu vzdálenosti od okresního města byly využity nástroje od společnosti Google. Pomocí Euklidovské vzdálenosti umí porovnat vývoj ceny mezi jednotlivými okresy a v obcích daného okresu umí zobrazit odhad ceny, kde je jeden nebo žádný inzerát. Využilo se faktorových modelů, nicméně musel být odebrán faktor nezaměstnanost v okrese, protože ten způsoboval při maticových počtech takovou hodnost matice, že vyšel determinant nula a nešla vypočítat inverzní matice a celý odhad v některých okresech. Aplikace by mohla mít do budoucna řadu rozšíření, co se týče výpočtu dalších sledovaných dat a pokud by se našel způsob pro porovnání ostatních nemovitostí, ne jenom bytů, tak počítat odhad i pro ně.

5 POUŽITÁ LITERATURA

[1] **MARKOVÁ, Hana**. *Ekonometrické modely*. Olomouc, 2011 [cit. 2017-05-15]. Dostupné z: http://theses.cz/id/yz58b8/Ekonometricke_modely_Markova_Hana.pdf. Diplomová práce. Univerzita Palackého v Olomouci, Přírodovědecká fakulta, katedra matematické analýzy a aplikací matematiky. Vedoucí práce Mgr. Jaroslav Marek, Ph.D.

[2] **HUŠEK, Roman**. *Ekonometrická analýza*. Vyd. 1. Praha: Oeconomica, 2007 [cit. 2017-05-15]. ISBN 978-80-245-1300-3.

[3] Regrese a kolerace: Index determinace. ŘEZÁNKOVÁ, Hana, Luboš MAREK a Michal VRABEC. *IASTAT: Interaktivní učebnice statistiky* [online]. 2001 [cit. 2017-05-15]. Dostupné z: http://iastat.vse.cz/regrese/Regrese9.htm

[4] **SVATOŠ, Jan**. *Monitorování realitního trhu*. Univerzita Pardubice, 2014. [cit. 2017-05-16] Bakalářská práce. Vedoucí práce Mgr. Jaroslav Marek, Ph.D.

[5] Co je GIS?. *Geoportal Praha* [online]. [cit. 16.05.2017] Dostupné z: http://www.geoportalpraha.cz/cs/clanek/11/co-je-gis#.WSLiCmjyjIW

[6] **KOREŇ, Milan**. *Geografický informačný systém ArcGIS* [online]. Technická univerzita vo Zvolene, 2014 [cit. 2017-05-16]. Dostupné z: http://gis.tuzvo.sk/tiki-download_file.php?fileId=437

[8] **ŠTORC, Ondřej**. Historie .NETu. *ITnetwork.cz: Sociální síť pro IT profesionály* [online]. 2017 [cit. 2017-05-16]. ISSN 2464-6326. Dostupné z: https://www.itnetwork.cz/csharp/historie-dotnetu

[7] Charts | Google Developers. *Google Developers* [online]. [cit. 2017-05-16] Dostupné z: https://developers.google.com/chart/

[9] Historie vývoj a aplikace dotazovacího jazyka linq. *Čečák.cz* [online]. 2017 [cit. 2017-05-16]. Dostupné z: http://www.cecak.cz/fel/dba/referaty/historie_vyvoj_a_aplikace_dotazovaciho_jazyka_linq

[10] LINQ a EF - Úvod. *Největší český web zaměřený na .NET framework* [online]. 2017 [cit. 2017-05-16]. Dostupné z: http://www.dotnetportal.cz/clanek/6413/LINQ-a-EF-Uvod

[11] 1. díl - Úvod do ASP.NET. *itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další.* [online]. 2017 [cit. 2017-05-16]. Dostupné z: https://www.itnetwork.cz/csharp/asp-net/tutorial-uvod-do-asp-dot-net

[12] Úvod do architektury MVC. *Zdroják - o tvorbě webových stránek a aplikací* [online]. [cit. 2017-05-16] Dostupné z: https://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/

[13] GitHub - aspnet/EntityFramework6: This is the codebase for Entity Framework 6 *The world's leading software development platform · GitHub* [online]. 2017 [cit. 2017-05-16]. Dostupné z: https://github.com/aspnet/EntityFramework6

[14] Toad Data Modeler Software and Data Modeling Tools. Quest | *IT Management* | *Mitigate Risk* | *Accelerate Results* [online]. 2017. [cit. 2017-05-16]. Dostupné z: https://www.quest.com/products/toad-data-modeler/

[15] Visual Studio – MSDN Blog CZ/SK. MSDN Blogs – Get the latest information, insights, announcements, and news from Microsoft experts and developers in the MSDN logs. [online]. 2017 Microsoft [cit. 16.05.2017]. Dostupné z: https://blogs.msdn.microsoft.com/vyvojari/p/visual-studio/

[16] Download SQL Server Management Studio. *Https://www.microsoft.com/* [online]. 2017 [cit. 2017-05-16]. Dostupné z: https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms

[17] Introduction. *Object moved* [online]. [cit. 2017-05-16] Dostupné z: http://www.newtonsoft.com/json/help/html/Introduction.htm

[18] Math.NET Numerics. *Math.NET Numerics* [online]. [cit. 2017-05-16] Dostupné z: https://numerics.mathdotnet.com/

[19] C# Bulk Operations: Bulk Insert, Update, Delete, Merge, and More!. *ZZZ Project* [online]. 2017 [cit. 2017-05-16]. Dostupné z: http://bulk-operations.net/

[20] Google Maps Distance Matrix API | Google Developers. *Google developers* [online]. [cit. 2017-05-16] Dostupné z: https://developers.google.com/maps/documentation/distance-matrix/