UNIVERZITA PARDUBICE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

BAKALÁŘSKÁ PRÁCE

2018 Matěj Grund

Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Doporučovací systém pro volbu restaurace

Matěj Grund

Bakalářská práce

2018

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2017/2018**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Matěj Grund**
Osobní číslo:	**I14092**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Doporučovací systém pro volbu restaurace**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Cílem práce bude vytvořit jednoduchý doporučovací systém, který dokáže uživatelovi podle jeho preferencí doporučit výběr obědu z denního menu podle aktuální nabídky lokálních restaurací. Aplikace bude implementována ve studentem zvolených technologiích s přihlédnutím na uživatelskou použitelnost. Text bakalářské práce bude kromě samotné problematiky obsahovat i přehled použitých technologií, rešerši o existujících alternativách a analýzu s realizací aplikace. Dále by neměl chybět nástin logiky rozhodování při výběru doporučení konkrétního jídla (restaurace).

Rozsah grafických prací:

Rozsah pracovní zprávy: **30-40 stran**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

PETR, Pavel. Data Mining. Vyd. 3. Pardubice: Univerzita Pardubice, 2010-. ISBN 978-80-7395-325-6.
AGGARWAL, Charu C. Recommender systems: The Textbook. New York, NY: Springer Science+Business Media, 2016. ISBN 978-3319296579.
KRUG, Steve. Don't make me think!: a common sense approach to Web usability. 2nd ed. Berkeley, Calif: New Riders Pub., c2006. ISBN 978-0321344755.

Vedoucí bakalářské práce: **Ing. Jan Merta**
Katedra softwarových technologií

Datum zadání bakalářské práce: **31. října 2017**

Termín odevzdání bakalářské práce: **12. května 2018**

Ing. Zdeněk Němec, Ph.D.
děkan

L.S.

Ing. Lukáš Čegan, Ph.D.
pověřený vedením katedry

V Pardubicích dne 20. března 2018

Prohlašuji:

Tuto práci jsem 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.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 5. 12. 2018

Matěj Grund

PODĚKOVÁNÍ

Tímto bych rád poděkoval panu Ing. Mertovi, který mi po dobu vypracovávání práce byl velice nápomocný a vždy byl ochoten se mnou diskutovat vhodnost jednotlivých bloků aplikace, formální úpravu a obsah bakalářské práce.

ANOTACE

Hlavním cílem bakalářské práce je vytvoření jednoduchého doporučovacího systému restaurace, který dokáže uživateli podle jeho preferencí doporučit výběr obědu z denního menu podle nabídky lokálních restaurací. Zároveň také umožní filtrovat jídla na denních menu do několika základních kategorií jídel. Tato aplikace je implementována v programovacích jazycích Java a JavaScript.

KLÍČOVÁ SLOVA

Doporučovací systém, JavaScript, Data mining.

TITLE

Restaurant recommender system.

ANNOTATION

The main goal of bachelor thesis is to create a simple restaurant recommender system, which is able to recommend user a selection of meals out of local restaurant daily menus, based on user's preferences. It is also able to filter a list of all daily menus to several basic food categories. This application is implemented in programming languages Java and JavaScript.

KEYWORDS

Recommender system, JavaScript, Data mining.

Obsah

Seznam obrázků

Seznam algoritmů

0 Úvod

Oběd je důležitou součástí dne každého z nás a existuje mnoho způsobů, jak se lidé mohou najíst. Jednou z možností je uvařit si jídlo doma a následně ho také doma sníst, nebo si jídlo zabalit a odnést s sebou do práce. Další možností je využití firemních kantýn, jejichž výhodou bývá levnější cena oproti restauracím a díky jejich častému umístění přímo ve firmách je také jejich dostupnost velkou výhodou. Třetím způsobem je možnost využití restaurací, které nabízejí rozvoz jídel do jejich blízkého okolí. Posledním způsobem je navštívení některé z lokálních restaurací. Jedná se o jeden z nejoblíbenějších způsobů, protože lidé nemusí doma řešit přípravu jídla na další den, mají větší výběr než ve firemních kantýnách a mohou se odreagovat tím, že na polední pauzu opustí areál jejich zaměstnání, a tím tento čas využijí k odpočinku od práce. Ve větších městech existuje velké množství restaurací, a tím pádem velký výběr jídel. Restaurace ve většině případů uvádějí na svých stránkách své denní menu, takže si lidé mohou vyhledat jídlo, na které mají chuť. To často může zabrat hodně času, než si lidé prohlédnou všechny webové stránky lokálních restaurací. Z toho důvodu dnes existují webové stránky, které seskupují veškerá dostupná menu restaurací na jednu přehlednou stránku, a tím usnadňují lidem výběr restaurace. Ovšem existuje ještě jedna možnost usnadnění výběru restaurace, která dosud není u nás využívána v tomto odvětví, a tou je využití doporučovacích systémů.

Doporučovací systémy jsou už dlouhou dobu důležitou součástí internetových služeb a jejich popularita stále stoupá. Jejich úlohami jsou například zvýšení prodeje produktů firem či podnikatelů nebo také slouží jako pomocník uživatelů, kterým na základě znalostí z jejich chování na internetu můžou doporučovat produkty, které uživatele zajímají.

Cílem této bakalářské práce je seznámit se se základními informacemi o doporučovacích systémech a navrhnout aplikaci, která pomocí metod data miningu a doporučovacích systémů bude schopna pomocí analýzy názvů jídel získat potřebné informace týkající se pokrmů, díky kterým budou uživatelům doporučována jídla na základě jejich preferencí. Zároveň tato webová aplikace nabídne možnost filtrace jídel do několika základních kategorií, což lidem velmi usnadní výběr hlavního jídla dne.

1 Doporučovací systémy

1.1 Historie

S růstem vlivu webových služeb na ekonomiku a obchodování začaly v 90. letech výrazně nabývat na důležitosti doporučovací systémy (Recommender Systems, dále jen RS). Brzy bylo znát, že internet nabízí nebývalé množství individuálních personalizací pro každého uživatele, na rozdíl od ostatních médií. Web nabízel hlavně jednoduchý sběr dat a vytvoření uživatelského rozhraní, které bylo možné použít pro doporučení položek uživateli takovým způsobem, aby mu to nebylo nepříjemné. [1]

Od té doby se RS značně rozvininuly z hlediska veřejného povědomí. Důkazem toho byl fakt, že se začly organizovat konference a semináře, které se věnovaly výhradně tomuto tématu. Téma RS je velice rozmanité, protože může používat různé typy uživatelských preferencí a dat za účelem získání doporučení. Mezi nejznámější metody RS patří kolaborativní doporučení (collaborative filtering), doporučení na základě obsahu (content-based) a doporučení založené na znalostech (knowledge-based). Tyto tři metody tvoří základní pilíře ve výzkumu RS. [1]

1.2 Úvod do doporučovacích sytému

Cílem RS je generovat uživatelům smysluplné doporučení položek nebo produktů, které by mohly zaujmout uživatele. Typickými příklady jsou doporučení filmů na platformě Netflix[1] nebo doporučení knih na stránce Amazon[2]. Návrh takových doporučovacích nástrojů závisí na oblasti daného problému a na specifických vlastnostech dostupných dat. Například uživatelé Netflix mohou filmy hodnotit na číselné škále v závislosti na tom, jak se jim film líbil. Taková data poté pomáhají k získání znalostí o vztahu mezi uživatelem a daným filmem. Dále je také možnost mít přístup k popisům položek a jejich atributům, nebo také k popisu uživatele a jeho preferencím. RS se liší ve způsobu, kterým analyzují zdroje dat, aby si vytvořily představu

[1] http://www.netflix.com/
[2] http://www.amazon.com/

o vztahu mezi uživatelem a položkou za účelem nalezení silného vztahu mezi těmito subjekty a vytvoření doporučení. [2]

Další formy získávání zpětné vazby nejsou tak explicitní jako předešlé, ale je jednodušší tyto data na webu získávat. Například pouhé zakoupení nějaké položky, nebo pouze její prohlížení může být považováno jako uživatelovo kladné hodnocení dané položky. Tyto formy zpětné vazby jsou používány zejména mezi internetovými obchody jako například Amazon. Tato forma sběru dat má výhodu, že nevyžaduje od zákazníka žádnou snahu a projevení jeho hodnocení. Základní myšlenka RS je využívat tyto různé zdroje dat k odvození zájmů zákazníků. Subjekt, kterému je doporučení poskytováno, je označován jako uživatel, a doporučovaný produkt je označován jako položka. Doporučovací analýza je často založena na předešlých interakcích mezi uživatelem a položkou, protože předešlé zájmy mezi těmito subjekty jsou často dobrým ukazatelem pro budoucí rozhodování. [1]

1.3 Role doporučovacích systémů

V předchozích kapitolách byly popisovány softwarové nástroje a techniky poskytující uživatelům návrhy na položky. Tato kapitola rozvine předchozí definice a nabídne několik příkladů rolí, které může RS nabývat. V tomto případě záleží na tom, jestli je na tento problém hleděno z pohledu zákazníka, nebo z pohledu subjektu, který nám poskytuje námi hledané položky. Například cestovní RS je zpravidla zaveden cestovním zprostředkovatelem (např. Expedia.com) nebo organizací turistické destinace (např. Visitfinland.com) za účelem zvýšení svého obratu, prodání více hotelových pokojů nebo nalákání více turistů do dané destinace. Zatímco zákazníkův hlavní účel, proč tyto služby navštívil je, aby si našel vhodný hotel a zajímavé události probíhající při návštěvě destinace. Ve skutečnosti je mnoho důvodů, proč by poskytovatelé služeb chtěli využívat těchto technologií. [3]

- *Zvýšení počtu prodaných položek.* Poskytovateli jde především o zvýšení počtu zákazníků, kteří si procházejí produkty na základě doporučení uživateli zajímavého produktu, které poté vede k jeho zakoupení, oproti zákazníkům, kteří si pouze procházejí náhodné produkty.

- *Prodej více rozdílných produktů.* Snaha nedoporučovat pouze nejpopulárnější produkty. Například uživatel platformy Netflix, který nesleduje populární filmy, pravděpodobně uvítá spíše doporučení na nějaký méně známý film.
- *Zvýšení spokojenosti zákazníka.* Kombinací efektivního a přesného doporučení může poskytovatel doporučit produkt, který se zákazníkovi zalíbí a produkt si zakoupí.
- *Zvýšení důvěry zákazníka.* Díky předchozím interakcím mezi poskytovatelem a zákazníkem (např. hodnocení položek, nákupy uživatele) získal RS potřebná data a nyní je schopen zákazníkovi doporučit produkty, které souvisí s předešlými nákupy zákazníka a jsou pro něj zajímavé.

Stejně užitečné jako pro poskytovatele služeb, jsou RS užitečné také pro uživatele. Některá doporučení mohou být považována jako hlavní úkoly, které jsou běžně spojovány s RS. To jsou například doporučení položek, které by mohly být uživateli užitečné. Ostatní doporučení se dají považovat spíše za příležitostné způsoby k využívání RS. S odlišností těchto dvou způsobů je možné se setkat například ve vyhledávači, jehož hlavní funkce je vyhledání dokumentů souvisejících s požadavky uživatele, ale může být také použit pro zjištění důležitosti webové stránky (sledování na jaké pozici se stránka ve výsledcích objeví). [3]

- *Doporučení některých položek.* Výběr pouze určitých položek, které se zobrazení jako seřazený seznam s možností předložení predikce, jak moc by se položka mohla uživateli líbit.
- *Doporučení všech položek.* Zobrazení všech položek, které by mohly uživatele zajímat. Tento způsob je využíván především, když není k dispozici velké množství výsledků.
- *Sekvenční doporučení*. RS se zaměřuje na doporučení položek, které jsou vhodné doporučit následně po předchozí položce. Jako příklad lze uvést televizní seriál, kde je vhodné po zhlédnutí jedné epizody doporučit následující epizodu nebo doporučit knihu o doporučovacích systémech po přečtení knihy o data miningu.
- *Doporučení balíčku.* Návrh skupiny položek, které se k sobě hodí. Například při nákupu fotoaparátu je možné nabídnout uživateli objektiv a stativ.
- *Vylepšení uživatelského profilu*. Toto souvisí se schopností uživatele podat dostatečné množství informací o jeho zájmech a s množstvím zadaných hodnocení. Bez těchto

znalostí není RS schopen podat personalizované doporučení. Je možné doporučit pouze položky, které se hodí pro průměrného uživatele.

1.4 Cíle doporučovacích systémů

Před diskutováním o cílech RS je potřeba představit různé způsoby jak zformulovat problém doporučování. Popisují ho následující dva primární modely.

- *Problém předpovězení hodnocení.* RS se pokouší předpovědět hodnocení v kombinaci uživatel-položka. Pro tento způsob je nutné mít tréninková data, která indikují preference uživatelů vůči položkám. Pro *m* uživatelů a *n* položek je definována matice *m x n,* kde jsou vyplněné hodnoty používány jako tréninková data. Chybějící hodnoty jsou poté doplněny navrženým algoritmem.
- *Problém seřazení položek.* Pro doporučení položky uživateli není vždy nutné předpovídat jejich hodnocení. V některých případech obchodníci raději doporučí top-k položek konkrétnímu uživateli nebo konkrétní položku pro top-k uživatele. [1]

Hlavním cílem RS je především zvýšení prodeje produktů a služeb, za účelem zvýšení profitu firem. RS se pokouší opatrnou selekcí významných položek upoutat pozornost uživatele pro tyto položky. Pro dosažení většího prodeje a profitu firem mají RS několik společných funkčních a technických cílů.

- *Relevance.* Nejdůležitějším cílem při funkčním doporučování je doporučit uživateli položku, která je pro uživatele nějakým způsobem zajímavá nebo užitečná. U takových položek je více pravděpodobné, že si uživatel tuto položku koupí.
- *Novinka.* RS jsou obzvláště nápomocné, když uživateli doporučí nějakou položku, která je pro něho nová a neznámá. Může se také stát, že opakovaným doporučováním pouze populárních položek může vést ke snížení profitu. Doporučením neznámých položek uživateli se zamezí snížení profitu.
- *Serendipita (označuje „příjemné překvapení“).* U doporučení položek, která jsou pro uživatele neznámé se vyskytuje prvek šťastného objevu. Tím se liší od doporučení, která jsou pro uživatele neznámá. Rozdíl od předchozího způsobu spočívá především v tom, že

u serendipity se jedná o opravdu překvapivá doporučení, na rozdíl od pouhého doporučení položky, o kterém uživatel dříve nevěděl. Tímto způsobem je možné zvýšit množství zájmů uživatele, ale může se často stávat, že tato doporučení budou pro uživatele nezajímavá.

- *Navýšení rozmanitosti doporučení.* RS typicky doporučují seznam top-k položek. Často se může stát, že se v tomto seznamu objeví hodně podobných položek, a tím se zvyšuje riziko, že se uživateli nebude líbit ani jedna z těchto položek. Zvýšením různorodosti položek v top-k seznamu se zvyšuje šance, že se uživateli zalíbí nějaká z těchto položek. [1]

1.5 Příklady

Mezi různými systémy je velká různorodost, týkající se typu doporučovaného produktu. Například Facebook[3] se zaměřuje místo doporučování produktu na doporučování přátelství. Uživatelé poté stráví na této sociální síti více času a zobrazí se jim na obrazovce více reklam. Pro lepší porozumění je zde vybráno několik historických a aktuálních RS z reálného světa. [1]

1.5.1 GroupLens

GroupLens byl průkopníkem mezi RS, který byl sestrojen jako výzkumný prototyp pro doporučování článků na síti Usenet[4]. Systém shromažďoval hodnocení čtenářů a použil je pro predikci, zda se uživatelům bude článek líbit. GroupLens je skupina akademiků z Minnesoty, kteří se podíleli na vývoji algoritmů pro původní kolaborativní filtrování. Později se také zaměřili na systémy pro doporučování knih a filmů, které dostaly názvy BookLens a MovieLens. [1]

1.5.2 Amazon

Amazon patřil mezi průkopníky RS, především v oblasti reklamy. Tato společnost původně prodávala knihy, ale později se rozšířila a nyní prodává produkty ve většině odvětví.

[3] http://www.facebook.com/
[4] http://www.usenet.com/

Doporučení jsou tvořena na základě explicitně zadaných hodnocení položek. Tato hodnocení jsou ve formě 5-ti hvězdičkové stupnice, kde počet hvězdiček znamená měřítko oblíbenosti dané položky. Doporučení se zobrazí na stránce po přihlášení uživatele. Často jsou také doporučení zobrazena současně s odůvodněním, proč byla daná položka doporučena. Dále tato síť používá implicitní hodnocení, která jsou získána chováním uživatele na webu. Například prohlížení položky nebo její zakoupení může být považováno jako implicitní kladné hodnocení. [1]

1.5.3 Netflix

Netflix je společnost, která pomocí internetového vysílání poskytuje uživatelům velkou kolekci filmů a televizních programů. Profit této firmy vzniká na základě měsíčního předplatného, které uživatelům umožňuje tuto službu používat. Uživatelé mohou předplatné kdykoliv ukončit, tudíž se Netflix snaží udržet zájem uživatelů, který obstarává primárně RS. [4]

Uživatelé mají možnost hodnotit filmy pomocí pěti hvězdičkové stupnice. Dále také Netflix uchovává informace o akcích uživatele v minulosti, například které filmy uživatel sledoval. Na základě těchto hodnocení a akcí uživatele jsou navržena doporučení, které jsou poskytnuta společně s vysvětlením, proč byla daná položka doporučena. Ve vysvětlení je popsáno na základě jakých filmů, které uživatel v minulosti viděl, je doporučení vytvořeno. Taková informace poté uživateli pomůže v rozhodnutí, zda by měl daný film zhlédnout. Podáním smysluplného vysvětlení doporučení se navyšuje šance, že si uživatel daný film vybere. Dále si tím také Netflix zajišťuje loajalitu a udržení zákazníků. [1]

1.5.4 Google News

Google News[5] je online portál, který shromažďuje články z několika tisíců zdrojů a zobrazuje je přihlášeným uživatelům v personalizované formě. Doporučení je založeno na způsobu kolaborativního filtrování a vstupní data pro doporučení jsou získávána pomocí kliknutí uživatelů na nějaký článek. [5]

[5] http://www.news.google.com/

Kliknutí jsou spojena s konkrétními uživateli, kteří byli identifikováni pomocí účtu Gmail. Kliknutí uživatele na článek se považuje za kladné hodnocení konkrétního článku. Tento způsob hodnocení se považuje za unární, jelikož uživatel kliknutím vyjádří kladné hodnocení článku, ale neexistuje žádný mechanismus pro uživatele jak vyjádřit negativní hodnocení. Pro personalizované doporučení článků uživateli jsou na shromážděná data aplikovány algoritmy kolaborativního filtrování. [1]

1.5.5 Facebook

Pro sociální sítě je často velmi důležité podávat uživatelům návrhy na přátelství, aby zvýšili počet sociálních spojení na síti. Facebook je v tomto ohledu perfektním příkladem. Doporučování na sociálních sítích má odlišné cíle na rozdíl od doporučování produktů, kde se firmě zvyšuje profit tím, že uživatelé nakupují dané produkty. U sociálních sítí je snaha, aby uživatelé měli větší požitek z návštěvy sítě a díky tomu strávili delší čas na sociální síti. Jelikož poskytovatelé těchto služeb jsou výrazně závislí na příjmu z reklam umístěných na jejich platformách, strávený čas uživatelů na jejich sítích je pro ně klíčový. RS Facebooku se zaměřuje na doporučení přátel, které je založeno spíše na sociálních vztazích než na hodnocení. [1]

1.6 Základní modely

Základní modely pro RS používají dva druhy dat. Prvním typem dat jsou interakce mezi uživateli a položkami, což mohou být například hodnocení nebo nákupní chování uživatele. Druhým typem jsou samotné popisky a atributy uživatelů nebo produktů, které mohou být ve formě textového popisu nebo ve formě klíčových slov. Pro zpracování těchto dat mají RS několik základních modelů, které mají odlišný přístup k těmto datům. Kolaborativní filtrování seskupuje hodnocení všech uživatelů, z kterých je tento model schopen odhadnout chybějící hodnocení konkrétního uživatele. RS založený na obsahu přihlíží především na hodnocení jednoho aktuálního uživatele, na rozdíl od kolaborativního filtrování, které hodnocení odhaduje na základě všech uživatelů. RS založený na znalostech vyhodnocuje doporučení na základě explicitně vyjádřených

požadavcích uživatele. Hybridní RS kombinují sílu jednotlivých modelů, což umožňuje vytvořit doporučení za více různých okolností. [1]

1.6.1 Kolaborativní filtrování

Hlavní myšlenkou kolaborativního filtrování je využívání informací o chování uživatelů, nebo vyjadřování jejich hodnocení za účelem odhadnutí, které položky by se konkrétnímu uživateli mohly pravděpodobně líbit. Tyto systémy mají v dnešní době široké využití, zejména jako nástroj u online maloobchodních sítí pro přizpůsobení obsahu ke konkrétním potřebám zákazníka. [5]

Kolaborativní filtrování si vytváří matici *a x b*, kde *a* jsou uživatelé a *b* jsou položky. Matice má vyplněné hodnoty tam, kde jsou známá hodnocení uživatele vůči položce. Prázdné hodnoty v matici jsou poté dopočítávány na základě podobnosti uživatelů nebo položek. Jako příklad lze uvést uživatele Boba a Alici, kteří mají podobný vkus. Pomocí algoritmu je možné zjistit podobnost mezi těmito uživateli. Při zjištění vysoké podobnosti mezi těmito dvěma uživateli, můžeme předpokládat, že u položek, u kterých má například pouze Bob vyplněné hodnocení, bude pravděpodobně hodnocení Alice velmi podobné. [1]

Kolaborativní metody lze podle knihy *Recommender Systems : The Textbook* [1] rozdělit následovně :

- Paměťové
    - Doporučení založené na podobnosti uživatelů (User-based)
    - Doporučení založené na podobnosti položek (Item-based)
- Modelové

Doporučení založené na podobnosti uživatelů (User-based)

Tento způsob využívá uživatele, kteří v minulosti hodnotili stejné položky podobným hodnocením jako uživatel A, za účelem vytvoření odhadu hodnocení uživatele A u položky, kterou tento uživatel dosud nehodnotil. Hlavní myšlenkou je tedy najít skupinu podobných uživatelů

s uživatelem A a poté pomocí váženého průměru této skupiny vypočítat doporučené hodnocení. Například pokud uživatelé Bob a Alice v minulosti podobně hodnotili stejné filmy, může být použito hodnocení Alice u filmu Terminátor pro předpovězení neznámého hodnocení u Boba pro stejný film. Podobnosti jsou vypočítávány z řádků matic hodnocení pro zjištění podobných uživatelů. [1]

Tento postup může být všeobecně shrnut do následujících kroků: [2]

1. Přiřazení váhy všem uživatelům s ohledem na podobnost s aktivním uživatelem.
2. Výběr *k* uživatelů (sousedů), kteří mají největší podobnost s aktivním uživatelem.
3. Výpočet předpovězeného hodnocení z vážené kombinace hodnocení podobných uživatelů

V prvním kroku se vypočítá váha $w_{a,u}$, která měří podobnost mezi uživatelem u a aktivním uživatelem a. Nejpoužívanější způsob pro výpočet váhy je použití Pearsonova korelačního koeficientu, jehož vzorec vypadá následovně:

$$w_{a,u} = \frac{\sum_{i \in I}\left(r_{a,i} - \bar{r_a}\right)\left(r_{u,i} - \bar{r_u}\right)}{\sqrt{\sum_{i \in I}\left(r_{a,i} - \bar{r_a}\right)^2 \sum_{i \in I}\left(r_{u,i} - \bar{r_u}\right)^2}}$$

kde I je množina položek hodnocených oběma uživateli, $r_{u,i}$ je hodnocení položky i uživatelem u, a $\bar{r_u}$ je průměr hodnocení uživatele u. [2]

Předpovězené hodnoty hodnocení jsou obvykle vypočítány jako vážený průměr odchylek od průměru souseda:

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u \in K}\left(r_{u,i} - \bar{r_u}\right) \times w_{a,u}}{\sum_{u \in K} w_{a,u}}$$

kde $p_{a,i}$ je předpověď hodnocení aktivního uživatele a pro položku i, $w_{a,u}$ je podobnost mezi uživateli a a u v množině podobných uživatelů K. [2]

Tento model patří ve srovnání s ostatními modely RS mezi ty přesnější. Je snadné ho naimplementovat a nepotřebuje znát žádné informace o položkách. Jelikož je tento model závislý

na hodnocení uživatelů, může se vyskytnout problém při malém počtu hodnocení nebo při zavedení webové služby či nové položky, kde nejsou zadána žádná hodnocení. [6]

Doporučení založené na podobnosti položek (Item-based)

Při používání doporučování založeném na podobnosti uživatelů nastává problém u velkých společností, které mají na svých sítích velké množství uživatelů a položek. Zjišťování možných podobností se sousedy je poté prakticky nemožné spočítat v reálném čase, z důvodu výpočtů milionů různých kombinací uživatelů. Z toho důvodu bylo zavedené doporučení na základě podobnosti položek. Hlavní myšlenkou je výpočet hodnocení na základě podobnosti položek místo uživatelů. [5]

Pro předpovězení hodnocení cílové položky B uživatelem A je prvním krokem určit množinu položek S, které jsou podobné položce B. Hodnocení v množině S, která jsou zadána uživatelem A, jsou použita pro předpovězení, zda se položka B bude uživateli A líbit. Například pokud uživatel hodnotil sci-fi filmy Vetřelec a Predátor, tak tato hodnocení mohou být použita pro odhadnutí hodnocení filmu Terminátor. [1]

Podobnosti položek jsou tímto způsobem off-line počítány pomocí Pearsonovy korelace následujícím způsobem:

$$w_{i,j} = \frac{\sum_{u \in U}(r_{u,i} - \overline{r_i})(r_{u,j} - \overline{r_j})}{\sqrt{\sum_{u \in U}(r_{u,i} - \overline{r_i})^2}\sqrt{\sum_{u \in U}(r_{u,j} - \overline{r_j})^2}}$$

kde U je množina všech uživatelů, kteří hodnotili obě položky i a j, $r_{u,i}$ je hodnocení uživatele u položky i, a $\overline{r_i}$ je průměrné hodnocení položky i. [2]

Hodnocení položky i od uživatele a může být předpovězeno výpočtem váženého průměru:

$$p_{a,i} = \frac{\sum_{j \in K} r_{a,j} w_{i,j}}{\sum_{j \in K} |w_{i,j}|}$$

kde K je množina k položek hodnocených uživatelem a, které jsou nejvíce podobné položce i. [2]

Tento model využívá společnost Amazon, jejichž databáze obsahuje více než 29 miliónů uživatelů a několik miliónů katalogových položek. Amazon používá model kolaborativního filtrování na základě podobnosti položek. Pro vylepšení výkonu používá Amazon pro doporučování dvě části RS. Pro vytváření matic a jejich náročných výpočtů je používán off-line RS. On-line RS používá tyto matice pro vytváření doporučení a je závislý na počtu zakoupených nebo ohodnocených položek aktivním uživatelem. [7]

Modelové doporučovací systémy

Při velkém množství dat mohou procesy odhadování hodnocení u paměťových RS trvat déle, než bychom chtěli. Z toho důvodu existují modelové RS, které si vytváří vlastní zjednodušený model z originálního souboru dat. Jinými slovy jsou z původního souboru dat vytaženy pouze nějaké informace, z kterých je vytvořen nový model. Poté při každém dalším předpovídání hodnocení je místo celého souboru dat používán pouze tento zjednodušený model. Po vytvoření modelu je postup předpovězení hodnocení stejný jako u paměťových RS. [8]

Díky výraznému zmenšení souboru dat a použití vytvořeného modelu pomůže tento způsob zjednodušit systému práci s daty, výrazně zrychlí proces odhadu hodnocení a také zamezí přetěžování systému. Jelikož vytváření modelu zabírá dost času a využívání dostupných zdrojů, je obtížné přidávat nová data do těchto modelů a to tvoří tyto systémy nepřizpůsobivé. Jestliže je pro doporučování využíván pouze zjednodušený model namísto celého souboru dat, předpovědi hodnocení jsou často méně přesnější než paměťové RS. [8]

1.6.2 Doporučovací systémy založené na obsahu

Tento způsob vytváří doporučení na základě popisujících atributů položek, které uživatel ohodnotil. Tyto znalosti jsou poté zkombinovány za účelem předpovědi doporučení. Pojem „obsah" odkazuje právě na tyto atributy. Například pokud uživatel ohodnotil vysokým hodnocením nějaký film, ale nejsou k dispozici hodnocení ostatních uživatelů, tak není možné

využít metody kolaborativního filtrování. Naopak, pokud máme k dispozici popis a atributy daného filmu, je možné na základě podobnosti atributů doporučit podobný film. [1]

Jelikož RS založené na obsahu nejsou závislé na hodnocení ostatních uživatelů, jsou obzvlášť vhodné pro nově zavedené položky do systému, protože je mohou doporučit na základě podobnosti atributů s položkami, které uživatel v minulosti ohodnotil. Na druhou stranu mohou tyto RS vytvářet předvídatelná doporučení, protože k vybírání doporučených položek využívají data o položkách, které uživatel v minulosti ohodnotil, tudíž uživatele nikdy nepřekvapí nečekaným doporučením nějaké nové položky. Dalším problémem může být vytvoření doporučení novým uživatelům, kteří nezadali dostatečný počet hodnocení pro vytvoření smysluplného doporučení. [1]

Proces doporučení položky u RS založených na obsahu se skládá z několika základních komponent. Jelikož pracují s mnoha různými popisy položek a informacemi o preferencích uživatelů, je třeba tato neuspořádaná data konvertovat do strukturovaných popisů, které mají většinou formu klíčových slov. Postup při doporučování se skládá ze tří hlavních komponent.

- Extrakce vlastností položek.
- Získání znalostí o profilu uživatele.
- Vytvoření doporučení. [1]

Extrakce vlastností položek

V první fázi je nutné získat z popisků položek jejich vlastnosti. V některých případech jsou k dispozici textové popisky, ze kterých je potřeba extrahovat klíčová slova, reprezentující klíčové vlastnosti položky. V druhém případě je možné se setkat již se strukturovanými popisky dat, kde jsou přehledně vypsány typy vlastností a jejich hodnoty. Těmto získaným vlastnostem je přidělena váha, která odpovídá důležitosti dané vlastnosti pro aktivního uživatele. [1]

Jedním z nejpoužívanějších způsobů pro extrakce vlastností je Gini index, který má následující tvar:

$$Gini(w) = 1 - \sum_{i=1}^{t} p_i\,(w^2)$$

kde *t* je množina všech možných hodnot hodnocení a $p(w^2)$ je frakce položek obsahujících danou vlastnost, které byly ohodnoceny hodnocením *t*.

Jelikož ve většině případů mají některé vlastnosti větší význam než ostatní, může být těmto vlastnostem přidělena jejich váha následujícím způsobem:

$$g(w) = a - Gini(w)$$

kde *a* je parametr určující váhu dané vlastnosti a je vždy větší než 1. [1]

Získání znalostí o profilu uživatele

Protože doporučení v tomto modelu úzce souvisí s požadavky a zájmy uživatele, je potřeba tyto data získat. Získání znalostí o uživateli lze dosáhnout pomocí zjištění historie hodnocení položek nebo z chování uživatele na dané službě. Tato zpětná vazba je poté používána ve spojení s atributy položek za účelem vytvoření tréninkového modelu. Z tréninkového modelu je poté sestrojen výsledný model, na který je často odkazováno pojmem uživatelský profil, protože koncepčně souvisí se zájmy uživatele vůči atributům položek. [1]

Vytvoření doporučení

Po extrakci vlastností a získání znalostí o zájmech uživatele jsou k dispozici dva sety tréninkových dat. První set obsahuje položky, které uživatel ohodnotil a druhý set obsahuje zbytek položek. Mezi nejjednodušší metody výpočtu patří klasifikace nejbližšího souseda, kterého lze zjistit pomocí kosinové podobnosti:

$$Cosine(\bar{X}, \bar{Y}) = \frac{\sum_{i=1}^{d} x_i y_i}{\sqrt{\sum_{i=1}^{d} {x_i}^2}\sqrt{\sum_{i=1}^{d} {y_i}^2}}$$

kde $\bar{X}(x_1 \dots x_d)$ *a* $\bar{Y}(y_1 \dots y_d)$ jsou páry dokumentů, kde jsou normalizované frekvence vlastností i-tého slova dány pomocí x_i a y_i. [1]

1.6.3 Doporučovací systémy založené na znalostech

RS založené na znalostech jsou především využívány v případech, kdy položky nejsou příliš často kupovány. To může být způsobeno vysokou cenou nebo také třeba velkým množstvím různých položek. Mezi takové kategorie patří například drahé šperky, finanční služby nebo automobily. U těchto položek je pro vytvoření doporučení předchozími způsoby nedostatečné množství hodnocení. Z tohoto důvodu RS založené na znalostech nezahrnují do procesu doporučování hodnocení položek. Preference uživatele mohou být zadány přímo v profilu uživatele nebo jsou preference zadávány přímo při procházení katalogu. Tento typ RS může být dále rozdělen na dvě části.

- *Doporučení na základě omezení (Constraint-based recommender systems).* Výsledky jsou vyhodnocovány pomocí zadaní požadavků uživatele při vyhledávání. Tyto požadavky mají většinou nějaké rozmezí (např. cena, rok výroby) nebo konkrétní hodnotu atributu.
- *Doporučení na základě požadavků (Case-based recommender systems).* Uživatel při procházení katalogu najde položku, která se mu líbí. Poté tuto položku zvolí jako jeho požadavek (case) a systém poté vyhledá podobné položky. Uživatel má také možnost upravit nějaký atribut jeho požadavku. Například pokud nalezne dům, který se mu líbí, může vyhledat podobné domy s jinou adresou. [1]

V obou případech je uživateli poskytována možnost zadat či změnit požadavky. Ovšem zadání těchto požadavků se v každém případě liší. U doporučení na základě požadavků je vybrána konkrétní položka (požadavek), která je použita jako kotevní bod pro následné doporučení. U druhého způsobu jsou explicitně zadány pravidla, podle kterých se má proces doporučení řídit. Zadání těchto požadavků může být implementováno několika způsoby.

- *Konverzační systémy.* Uživatelské požadavky jsou iterativně určeny v kontextu zpětné vazby.

- *Systémy založené na vyhledávání.* Požadavky pro doporučení jsou zjištěny pomocí série otázek, týkajících se preferencí uživatele.
- *Doporučení založené na navigaci*. Uživatel specifikuje několik změn na aktuálně doporučované položce. Po každé změně je uživateli doporučená nová položka. Účelem tohoto způsobu je dostat uživatele po sérii změn k nejvíce vhodné položce. [1]

Jelikož RS založené na znalostech silně závisí na atributech položek a nejsou do doporučení zahrnuty hodnocení, mohou být tato doporučení často předvídatelná. V tomto ohledu jsou velice podobné RS založeným na obsahu. Hlavním rozdílem však je, že RS založené na obsahu přihlíží na historii hodnocení konkrétního uživatele. [1]

Doporučení na základě omezení

Tento způsob umožňuje uživatelům zadat konkrétní požadavky na atributy položek. Tyto požadavky mohou mít formu rozmezí, výčtu hodnot nebo konkrétní hodnoty. Například pokud uživatel prochází katalog domů, může poté zadat cenové rozmezí, konkrétní počet místností, minimální rozlohu nemovitosti nebo také seznam lokalit. Uživatel má také možnost upravit svůj vlastní profil, kde je možné vyplnit vlastní preference. Pro vyhledávání jsou poté použity dva základní vstupy.

- Doporučení je vytvořeno na základě zadaných požadavků při vyhledávání, které jsou zkombinovány s informacemi o preferencích z uživatelského profilu.
- Vytvoření doporučení závisí na zadaných atributech uživatelem v moment vyhledávání. Tyto preference je možné získat dvěma způsoby.
    - *Přímé zadání.* Uživatel zadá seznam požadavků.
    - *Odvození.* Ze zadaného seznamu požadavků je možné pomocí všeobecných znalostí vytvořit nová pravidla. Například pokud uživatel hledá byt a zadá, že jeho rodina se skládá ze čtyř členů, systém díky této znalosti může předpokládat, že tato rodina potřebuje byt s minimálně třemi místnostmi. [1]

Poté co systém nabídne uživateli seznam položek, může uživatel seznam položek seřadit. Řazení je vhodné pokud uživatel dostane velké množství výsledků, nebo pokud má nějaký

primární atribut, podle kterého chce seznam výsledků seřadit. Pro seřazení položek je využívána následující funkce:

$$U(\bar{V}) = \sum_{j=1}^{d} w_j \times f_j(v_j)$$

kde $\bar{V}(v_1 \ldots v_d)$ je vektor hodnot atributů doporučených položek o rozměru d. Každá položka může mít zadanou váhu w_j a její podíl, který popisuje funkce $f_j(v_j)$. [1]

Doporučení na základě požadavků

U doporučení na základě požadavků jsou používány funkce podobnosti, které doporučí uživateli podobnou položku k položce, kterou zvolil. Pro doporučení další položky jsou použity hodnoty atributů zvolené položky a uživatel má také možnost nějaké atributy změnit. Pro nalezení takového výsledku jsou používány funkce podobnosti. Pokud chce uživatel doporučit podobnou položku, může u některých parametrů specifikovat jiné hodnoty a u některých parametrů chce, aby hodnota zůstala stejná. Při změně hodnot hledá systém asymetrickou podobnost, v druhém případě jde o symetrickou podobnost. [1]

V případě symetrické podobnosti mezi položkami t_i a x_i je pro výpočet podobnosti $Sim(t_i,x_i)$ použit následující vzorec:

$$Sim(t_i, x_i) = 1 - \frac{|t_i - x_i|}{max_i - min_i}$$

kde max_i, min_i reprezentují maximální a minimální možnou hodnotu atributu i. V případě asymetrické podobnosti atributů uživatel hledá větší nebo menší hodnotu atributu. V tomto případě je k původnímu vzorci přičtena tzv. asymetrická odměna. Pokud jsou vyšší hodnoty atributu pro uživatele vhodnější, vzorec vypadá takto:

$$Sim(t_i, x_i) = 1 - \frac{|t_i - x_i|}{max_i - min_i} + \alpha_i \times I(x_i > t_i) \times \frac{|t_i - x_i|}{max_i - min_i}$$

kde $\alpha_i \geq 0$ je parametr zadaný uživatelem, určující o jakou hodnotu se mají parametry lišit a funkce $I(x_i > t_i)$, která kontroluje, zda se jedná o vyšší hodnotu než hodnota původního parametru. V případě, že pro uživatele jsou vhodnější nižší hodnoty, vypadá výpočet následovně:

$$Sim(t_i, x_i) = 1 - \frac{|t_i - x_i|}{max_i - min_i} + \alpha_i \times I(x_i < t_i) \times \frac{|t_i - x_i|}{max_i - min_i}$$

Funkce pro výpočet podobnosti dvou položek vypadá takto:

$$f(\overline{T}, \overline{X}) = \frac{\sum_{i \in S} w_i \times Sim(t_i, x_i)}{\sum_{i \in S} w_i}$$

kde $\overline{T}, \overline{X}$ reprezentují dvourozměrné vektory cílové a původní položky, S je množina atributů a $Sim(t_i, x_i)$ je podobnost *i*-tého atributu s váhou w_i. [1]

1.6.4 Hybridní doporučovací systémy

Každý RS, který tu byl probrán, má nějaké výhody a nevýhody. Mezi nevýhody patří například problém studeného startu u RS, které jsou závislé na hodnocení uživatelů, nebo problém lehce předvídatelných doporučení u systémů zanedbávajících hodnocení uživatelů. Tyto problémy pomáhají odstranit hybridní RS, které kombinují předešlé modely za účelem efektivnějšího doporučování. Hybridní RS se dají rozdělit do několika kategorií.

- *Vážené*. Skóre ze všech zkombinovaných RS je zkombinované do jednotného skóre, pomocí výpočtu vážených souhrnů skóre z jednotlivých komponent.
- *Přepínací*. Algoritmus přepíná mezi různými RS v závislosti na aktuálních potřebách. Tento systém například používá RS založený na znalostech při startu webové služby. Po

dosažení dostatečného počtu hodnocení může zase přepnout na model kolaborativního filtrování.

- *Kaskádové*. V tomto případě mají RS určené pořadí, ve kterém vyhodnocují doporučení. Následující RS vždy vylepší doporučení předešlého RS. Při vytváření tohoto doporučení je vždy přihlíženo na hodnotu doporučení předešlého RS. Poté jsou tyto hodnoty zkombinovány do jednoho výstupu.
- *Rozšíření vlastností*. Tento způsob funguje na podobném principu jako kaskádové RS, ale místo vylepšování hodnocení předešlého RS je hodnota doporučení předešlého RS brána jako vlastnost a je použita jako vstupní parametr pro další RS.
- *Kombinace vlastností*. Pokud je k dispozici více zdrojů dat, tak tento způsob kombinuje vlastnosti ze všech dostupných zdrojů.
- *Meta-úroveň*. Výsledný model jednoho RS je použit jako vstupní model do dalšího RS.
- *Smíšený*. Uživateli jsou nabídnuty výsledky ze všech RS najednou. [1] [3]

2 Návrh modulu pro doporučovací systém

V dnešní době je mnoho, kteří během práce navštěvují restaurace v době obědu. Ve velkých městech mají tito lidé desítky restaurací, ze kterých si můžou vybírat. Pro usnadnění výběru existují na internetu portály, které nabízejí seznam restaurací včetně jejich denních menu, a to na jedné přehledné stránce. Tyto portály ovšem nabízejí pouze seznam jídel a restaurací bez jakékoliv možnosti personalizace. Cílem této práce je získání potřebných dat pro demonstraci funkčnosti využití doporučovacích systémů v tomto odvětví a jejich následná analýza využívající metody data miningu. Tato metoda získává informace o jídlech, potřebné k vytváření doporučení na základě uživatelských preferencí. Zároveň je třeba využít znalost doporučovacích systémů pro výběr správného modelu. Ten je zvolen na základě dostupných informací o jídlech a preferencích uživatelů. Využitím výše zmíněných metod by aplikace měla mít možnost vyhodnotit základní vlastnosti jídel a jejich ingredience, což aplikaci umožní nabídnout uživatelům filtraci výpisu jídel na základě vlastních preferencí. Hlavním cílem aplikace je pomocí uživatelské selekce oblíbených kategorií získat informace o uživatelských preferencích. Díky tomu je poté možné doporučit uživatelům jídla, která jsou použitím metod doporučovacích systémů vybrána jako nejvhodnější pro daného uživatele. Pro splnění těchto cílů bude implementována aplikace, která je složena z několika modulů, do kterých patří aplikace pro data mining a analýzu dat, databáze a webová aplikace, která použitím analyzovaných dat z databáze vytváří webovou stránku a obstarává všechny její dynamické prvky.

2.1 Analýza aplikace

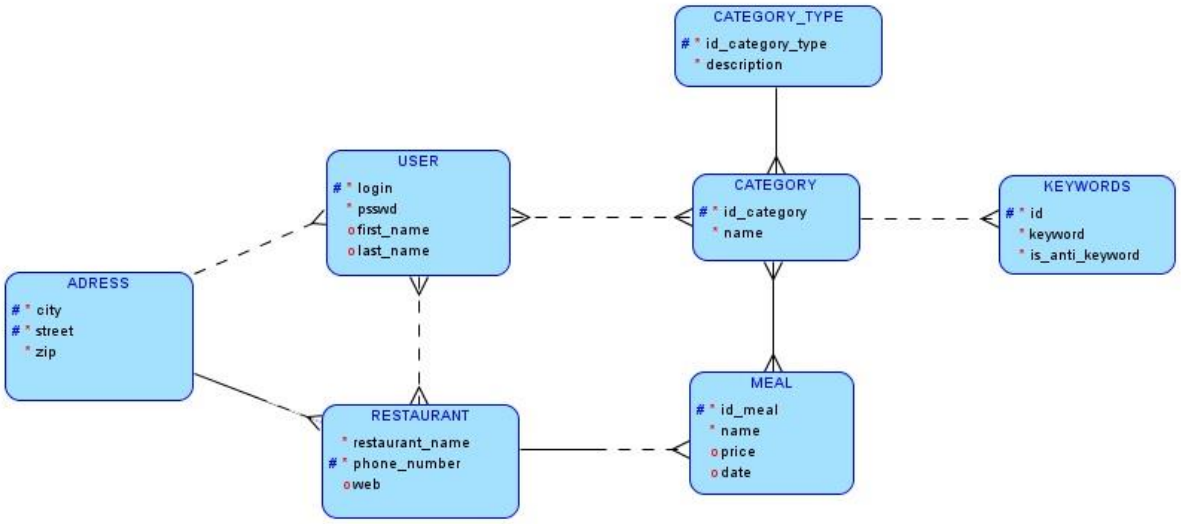
Pro funkčnost a dosažení požadovaných výsledků je aplikace složena z několika základních modulů. Základem pro tuto aplikaci jsou data jednotlivých restaurací a seznamy jídel na denním menu. Aby bylo možné simulovat funkčnost této aplikace, data jsou pro simulaci získána ze serveru Meníčka[6]. Veškeré informace o restauracích a jejich menu jsou získány z HTML kódu této služby. O získání a analýzu těchto dat se stará aplikace napsaná v programovacím jazyce Java. Získaná data jsou analyzována algoritmy, které jednotlivé pokrmy porovnávají s klíčovými slovy

[6] http://www.menicka.cz/

kategorií jídel. Tento proces přiřadí každému jídlu seznam kategorií, které aplikace u daného pokrmu rozpoznala. Tato data jsou ukládána do MySQL databáze, aby je mohla získat aplikace, která se stará o provoz webové služby. Webová aplikace je napsána v jazyce JavaScript a komunikuje se serverem založeném na Node.js.

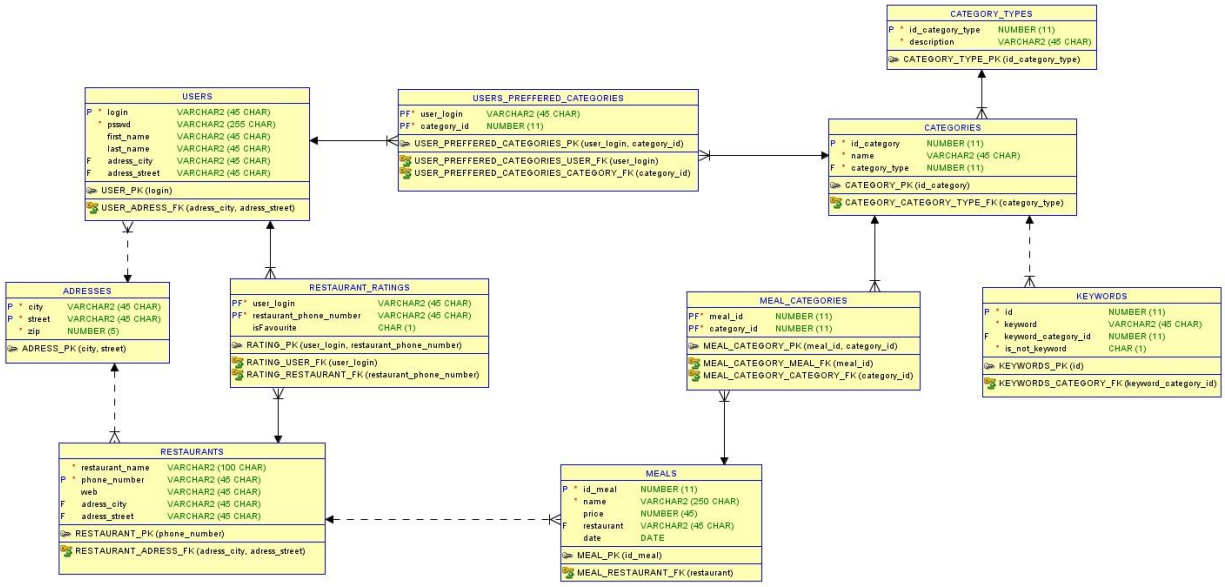
2.1.1 Databáze

Pro uchování dat této aplikace byla zvolena databáze MySQL. Logický model databáze se skládá ze sedmi základních entit. Pro uchování informací o samotných jídlech slouží entita *MEAL*, kde je jako primární klíč využíváno *id_meal*. Dále obsahuje povinný atribut obsahující informaci o názvu jídla, nepovinné atributy ceny jídla a datumu, kdy se jídlo nacházelo na menu z určitých restaurací. Entita *RESTAURANT* používá jako primární klíč telefonní číslo restaurace a druhým povinným atributem je název restaurace. Nepovinným atributem restaurace je její webová adresa. Jelikož by více entit využilo adresu jako jeden z atributů, byla pro ni vytvořena samostatná entita *ADRESS*. Primární klíč je složen z atributů město a ulice. Ulice obsahuje informaci o názvu ulice i čísle popisném. Dále je v této entitě povinný atribut poštovního směrovacího čísla. Informace o uživatelích uchovává entita *USER* s povinnými atributy přihlašovací jméno a heslo a nepovinnými atributy jméno a příjmení. Přihlašovací jméno je používáno jako primární klíč. Kategorie jídel popisuje entita *CATEGORY*, obsahující informaci o názvu kategorie a umělý primární klíč *id_category*. Tyto kategorie jsou poté rozděleny do několika typů, které popisuje entita *CATEGORY_TYPE*. Obsahuje atributy popis a primární klíč *id_category_type*. Každou kategorii popisuje seznam několika klíčových slov, které popisuje entita *KEYWORD*. Obsahuje atributy klíčové slovo a atribut *is_anti_keyword*, indikující zda dané slovo popisuje danou kategorii či ne. Na následujícím obrázku č. 1 je zobrazen logický model databáze.



Obrázek 1 *- Logický model databáze [autor, 2018]*

Jelikož některé relace mají vazbu M:N, jsou pro ně v relačním modelu databáze vytvořeny tabulky. Tabulka *RESTAURANT_RATINGS* obsahuje vztah mezi uživatelem a restaurací, kde je přidán atribut hodnocení dané restaurace uživatelem. Každý uživatel si může vybrat seznam oblíbených kategorií jídel, který poté slouží pro vytvoření doporučení. Tento seznam je ukládán do tabulky *USERS_PREFFERED_CATEGORIES*. Každý název jídla je analyzován pomocí klíčových slov. Díky tomuto procesu je zjištěno, do kterých kategorií jednotlivá jídla patří. Tento seznam kategorií je poté uložen do tabulky *MEAL_CATEGORIES*. Tento relační model je zobrazen na následujícím obrázku č. 2.



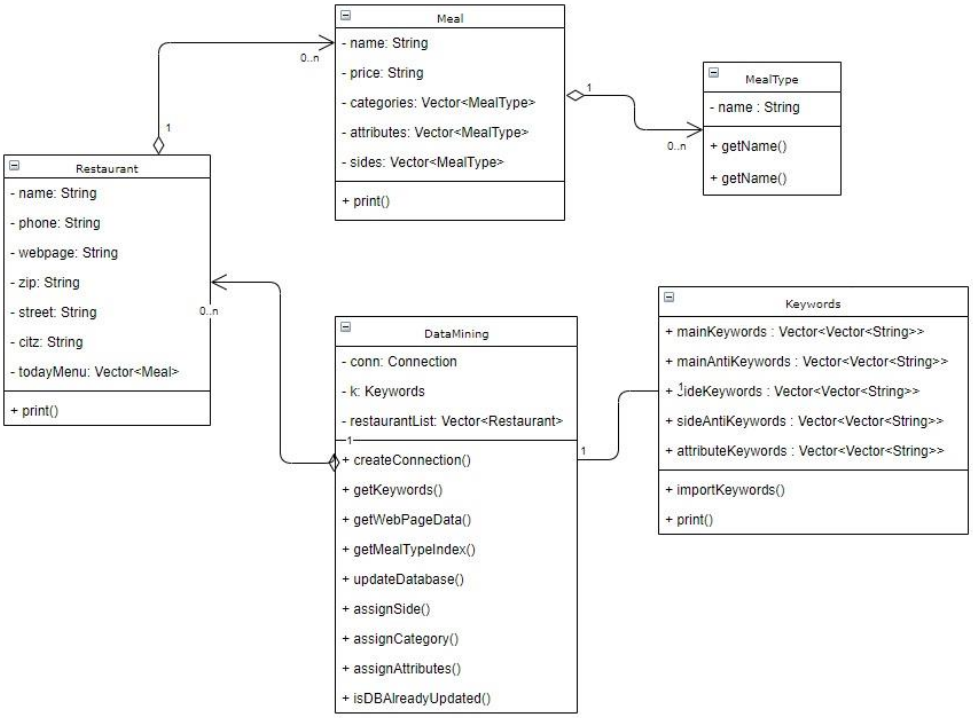
Obrázek 2 - *Relační model databáze [autor, 2018]*

2.1.2 Data mining

Základem pro funkčnost této webové aplikace je získání a analýza dat. Jelikož tato aplikace pouze demonstruje funkčnost doporučovacího systému, jsou data pro tuto demonstraci získána z HTML kódu webového serveru Meníčka. Tento web poskytuje denní menu restaurací po celé České republice. Tato aplikace používá pro demonstraci pouze menu Pardubických restaurací. Po získání dat jsou analyzovány názvy jídel a následně vyhodnoceny kategorie, do kterých tato jídla patří. Tato data jsou poté odeslána do databáze. Pro práci s informacemi o jídlech používá aplikace pět základních tříd:

- DataMining.java - hlavní třída aplikace, která importuje klíčová slova z databáze a seznamy jídel restaurací, kterým přiřazuje atributy a výsledky poté odesílá do databáze
- Keywords.java – třída sloužící pro uchování seznamů klíčových slov jednotlivých kategorií
- Meal.java – uchovává popis, cenu a seznam kategorií jídla
- Restaurant.java – třída restaurace, kde jsou v parametrech uchovány všechny informace o restauraci včetně jejího aktuálního menu

- MealType.java – výčtový typ pro kategorie pokrmů



Obrázek 3 - *UML diagram Java aplikace [autor, 2018]*

Získání dat

Potřebná data pro práci s touto aplikací jsou získána pomocí metod web scrapingu z HTML adresy *https://www.menicka.cz/padrubice.html*, kde se nachází seznam menu pardubických restaurací. Jednotlivé restaurace mají svůj vlastní HTML blok. Po nahrání HTML dokumentu do aplikace je pomocí jednotného identifikátoru bloků restaurací vytvořen seznam elementů, obsahujících informace o restauracích. Z těchto bloků jsou poté získána data o restauracích a jejich aktuální seznam jídel na menu. Proces získání dat je zobrazen v algoritmu č. 1. Tato data jsou uložena do pole objektů *Restaurant.* Tento objekt obsahuje následující atributy:

- String name
- String phone

- String webpage
- String zip
- String street
- String city
- Vector<Meal> todayMenu

```
static void getWebPageData(Vector<Restaurant> restaurantList) throws IOException{
    Restaurant tmpRest;
    Meal tmpMeal;
    final Document doc = Jsoup.connect("https://www.menicka.cz/pardubice.html").get();
    Elements temp = doc.select("div.menicka_detail");
    for (Element element : temp) {
        Elements restaurantName = element.select("div.hlavicka div.info div.nazev");
        Elements restaurantPhone = element.select("div.hlavicka div.info div.telefon");
        String webpage = element.select("div.hlavicka div.info div.nazev a").attr("href");

        final Document restaurantPage = Jsoup.connect(webpage).get();
        String street = restaurantPage.select("div.info span.adr p.street-address").text();
        String zip = restaurantPage.select("div.info span.adr p span.postal-code").text();
        String city = restaurantPage.select("div.info span.adr p span.locality").text();
        String webAdress = restaurantPage.select("div.info p a.url").attr("href");

        tmpRest = new Restaurant(restaurantName.text(), restaurantPhone.text(), webAdress, city, street, zip);
        Elements foodList = element.select("div.menicka div.nabidka_1");
        Elements foodList2 = element.select("div.menicka div.nabidka_2");
        for (Element element1 : foodList) {
            if (element1.attr("class").equals("nabidka_1 capitalize") == false) {
                tmpMeal = new Meal(element1.text(), selectNthElementAfter(element1, "div", 1).text());
                tmpRest.addMenuItem(tmpMeal);
            }
        }
        for (Element element1 : foodList2) {
            if (element1.getElementsByClass("more").size() == 0) {
                tmpMeal = new Meal(element1.text(), selectNthElementAfter(element1, "div", 1).text());
                tmpRest.addMenuItem(tmpMeal);
            }
        }

        if (tmpRest.getPhone().length() == 18) {
            restaurantList.add(tmpRest);
        }
    }
}
```

***Algoritmus 1** - Získání dat pomocí Web scrapingu [autor, 2018]*

Analýza dat

Jelikož tato aplikace využívá doporučovací systém založený na znalostech, je potřeba získat z názvů jídel jejich atributy a vlastnosti. Atributy označují například z jakého masa je daný pokrm připraven, zda se jedná o vegetariánské jídlo nebo jestli je to sladký pokrm. Dalšími atributy mohou být například způsob přípravy jídla, příloha, použité ingredience nebo také konkrétní název jídla, které patří mezi populární české pokrmy. Pro analýzu pokrmů se používají klíčová slova popisující jednotlivé kategorie, která jsou uložena v databázi. Klíčová slova jsou rozdělena do dvou typů. První typ obsahuje slova která indikují, zda dané jídlo patří do kategorie jídel a druhý typ vylučuje jídlo z dané kategorie. Příkladem by mohla být analýza jídla, který má následující popis:

"Svíčková z vepřové kýty s houskovým knedlíkem".

Jelikož mezi klíčová slova hovězích jídel patří slovo „*svíčková*", protože ve velké většině případů se jedná o pokrm z hovězího masa, program by výše uvedené jídlo zařadil mezi pokrmy z hovězího masa. Z toho důvodu existuje druhý typ klíčových slov pro každou kategorii, které vylučují, že se jedná o danou kategorii. V tomto případě je mezi takovými slovy u kategorie hovězích jídel klíčové slovo "*vepř",* které v tomto případě zajistí, aby výše uvedený pokrm nebyl zařazen do kategorie hovězích jídel.

Poté, co jsou importována aktuální menu restaurací a získána klíčová slova z databáze, aplikace zavolá funkce pro přiřazení atributů všem pokrmům aktuálního dne. Jelikož jídla mohou mít několik přiřazených atributů, probíhá proces porovnávání s klíčovými slovy u všech kategorií. Po přiřazení atributů pokrmům jsou všechna data odeslána do databáze. Při nahrávání dat do databáze aplikace kontroluje, zda se všechny restaurace nacházejí v databázi. Pokud ano, nahrávána jsou pouze nová jídla s jejich atributy, v opačném případě je do databáze přidána nová restaurace s příslušnými informacemi. Algoritmus č. 2 popisuje analýzu přílohy jídla.

```
public static void assignSide(Meal meal, Keywords k){
    boolean isCategory = false;
    for (int i = 0; i < k.sideKeywords.size(); i++) {
        for (String key : k.sideKeywords.get(i)) {

            if (meal.getName().matches("(?i:(.*)" + key + "(.*))")) {
                isCategory = true;

                for (String antiKey : k.sideAntiKeywords.get(i)) {
                    if (meal.getName().matches("(?i:(.*)" + antiKey + "(.*))")) {
                        isCategory = false;
                        break;
                    }
                }

                if (isCategory) {
                    meal.addSide(MealType.values()[i+47]);
                    break;
                    }
                break;
            }
        }
    }
}
```

***Algoritmus 2** – Analýza přílohy jídla [autor, 2018]*

2.1.3 Webová aplikace

Webová aplikace importuje analyzovaná data z databáze a výsledná data vypíše na webovou stránku s možností filtrace jídel do několika základních kategorií. Pro přihlášené uživatele jsou vyhodnocena doporučená jídla. Uživatelům také aplikace nabízí vybírat si oblíbené restaurace a pomocí adresy v uživatelském profilu zobrazovat vzdálenosti restaurací.

Import dat z databáze

Data jsou z databáze importována pomocí SQL dotazu, závislém na dvou atributech. Prvním atributem je *onlyFavouriteRestaurants*, který zajistí, zda databáze vrátí data všech restaurací, nebo pouze oblíbené restaurace přihlášeného uživatele. Druhý atribut *category* určuje, zda jsou vrácena všechna jídla dostupná k aktuálnímu dni, nebo pouze jídla, která patří do specifikované kategorie, jejíž atribut *id_category* v databázi odpovídá hodnotě atributu *category*. Následuje algoritmus č. 3, na kterém je uveden způsob získávání seznamu požadovaných jídel.

```
app.get('/getMeals', function (req, res) {
    console.log('GET');
    var category = req.query.categories;
    var onlyFavouriteRestaurants = req.query.onlyFavouriteRestaurants;

    if (onlyFavouriteRestaurants == "true") {
        if (category === "0") {
            var query =
                "SELECT n.name, price, restaurant, restaurant_name, adress_city, adress_street, web" +
                " FROM recommender_system.meal_categories m join meals n on m.meal_id=n.id_meal join categories c on m.category_id=c.id_category " +
                "join restaurants r on r.phone_number=n.restaurant  join restaurant_ratings rr on n.restaurant=rr.restaurant_phone_number " +
                "where date=(select curdate()) and user_login='" + loggedInUser.username + "' group by n.name";
        } else {
            var query = "SELECT n.name, price, restaurant, restaurant_name, adress_city, adress_street, web" +
                " FROM recommender_system.meal_categories m join meals n on m.meal_id=n.id_meal join categories c on m.category_id=c.id_category " +
                "join restaurants r on r.phone_number=n.restaurant  join restaurant_ratings rr on n.restaurant=rr.restaurant_phone_number " +
                "where id_category=" + category + " and date=(select curdate()) and user_login='" + loggedInUser.username + "'";
        }
    } else {
        if (category === "0") {
            var query =
                "SELECT n.name, price, restaurant, restaurant_name, adress_city, adress_street, web" +
                " FROM recommender_system.meal_categories m join meals n on m.meal_id=n.id_meal join categories c on m.category_id=c.id_category " +
                "join restaurants r on r.phone_number=n.restaurant where date=(select curdate()) group by n.name";
        } else {
            var query = "SELECT n.name, price, restaurant, restaurant_name, adress_city, adress_street, web" +
                " FROM recommender_system.meal_categories m join meals n on m.meal_id=n.id_meal join categories c on m.category_id=c.id_category " +
                "join restaurants r on r.phone_number=n.restaurant where id_category=" + category + " and date=(select curdate())";
        }
    }

    con.query(query, function (err, result) {
        if (err) {
            res.setHeader('Content-Type', 'application/json');
            res.send(JSON.stringify(err));
        }
        res.setHeader('Content-Type', 'application/json');
        res.send(JSON.stringify(result));
    });
});
```

Algoritmus 3 - *Získání seznamu požadovaných jídel [autor, 2018]*

Tvorba bloků restaurací

Po importu dat z databáze jsou na webovou stránku vykreslovány jednotlivé bloky restaurací. Jelikož jsou z databáze nahrána všechna jídla restaurací v jedné neuspořádané proměnné, aplikace v metodě *createTables()* tento seznam jídel filtruje. Následně výsledky ukládá do uspořádaného pole restaurací, kde každá restaurace má proměnnou *meals*, do které je pomocí filtrování uložen seznam jídel odpovídající dané restauraci. Jednotlivé proměnné restaurací jsou poté předány metodě *createTable()*, která vytvoří HTML blok restaurace, zobrazí její název, telefon, adresu a tabulku s pokrmy. Zároveň je do daného bloku přidáno tlačítko pro přidání restaurace mezi oblíbené, vzdálenost restaurace (pokud je známá adresa uživatele) a tlačítko pro zobrazení celého menu v případě, že si uživatel zobrazuje pouze jídla patřící do zvolené kategorie. Algoritmus č.4 zobrazuje postup při tvorbě jednotlivých bloků restaurací.

```
var createTables = function (restaurantNames, mealsList, isRecommended, isCategory) {
    if (!getMealsOnce && !isRecommended) {
        restaurantsDetails = restaurantNames;
        getMealsOnce++;
    }

    restaurantNames.forEach(function (item) {
        item.meals = [];
        item.meals = mealsList.filter(function (elem) {
            if (elem.restaurant_name === item.restaurant_name) {
                return true;
            }
            return false;
        });
        if (item.meals.length == 0) {
            console.log("");
        }
    });

    var leftOrRight = true;
    restaurantNames.forEach(function (item) {
        if (item.meals.length > 0) {
            createTable(item.meals, leftOrRight, isCategory, isRecommended);
            leftOrRight = !leftOrRight;
        }

    });
    displayOnlySpecified();
};
```

Algoritmus 4 - *Dynamická tvorba bloků jednotlivých restaurací [autor, 2018]*

Registrace uživatele

Po zadání uživatelských údajů aplikace kontroluje, zda jsou zadány všechny povinné atributy. Zároveň probíhá také kontrola, zda je uživatelské jméno již zabrané a jestli heslo obsahuje minimální počet znaků. Při nesplnění nějaké z výše uvedených podmínek aplikace informuje uživatele z jakého důvodu není možné dokončit registraci. Pokud jsou všechny podmínky splněny, je nový uživatel přidán do databáze. Pokud uživatel vyplnil adresu, je nejdřív kontrolováno, zda se adresa nachází v databázi. Pokud ano, uživatel je přidán do databáze a jeho adresa odkazuje na již existující adresu v databázi. V opačném případě je vložena nová adresa do tabulky adres. Následně je uživatel přesměrován na přihlašovací obrazovku.

Šifrování hesla

Pro šifrování hesla je využívána techlogie šifrování BCrypt. Tato technologie je založena na symetrické blokové šifře nazývající se Blowfish, která v tomto případě nabývá formy adaptivní hashovací funkce. BCrypt využívá KeyFactor, pomocí kterého je tato technologie schopná ovlivňovat náročnost hashování. Díky tomu je BCrypt zařazován mezi nejlepší šifrovací technologie hesel. Využití této technologie je zobrazeno v následujícím algoritmu č. 5. [20]

```
app.post('/addUser', function (req, res) {
    console.log("POST ");
    var query = "INSERT INTO users VALUES (?, ?, ?, ?, ?, ?)";
    var login = req.body.login;
    var password = req.body.password;
    var firstName = req.body.firstName;
    var lastName = req.body.lastName;
    var city = req.body.city;
    var street = req.body.street;
    var arr = [req.body.login, req.body.password, req.body.firstName, req.body.lastName, req.body.city, req.body.street];
    console.log(arr);

    bcrypt.hash(password, saltRounds, function (err, hash) {
        con.query(query, [login, hash, firstName, lastName, city, street], function (err, result) {
            if (err) {
                res.setHeader('Content-Type', 'application/json');
                res.send(JSON.stringify(err));
            } else {
                res.setHeader('Content-Type', 'application/json');
                res.send(JSON.stringify("Uzivatel pridan!"));
            }

        })
    });
});
```

Algoritmus 5 - *Přidání uživatele do databáze s šifrováním hesla [autor, 2018]*

Při vytváření uživatele jsou veškerá hesla před komunikací s databází šifrována metodou *bcrypt.hash()*, která zajistí odeslání hesla do databáze v zašifrované podobě. V případě ověřování shody hesel při přihlašování uživatele je heslo kontrolováno metodou *bcrypt.compare()*, která je použita v algoritmu č. 6.

```
app.post('/validateOldPassword', function (req, res) {
    console.log("POST");
    var pass = req.body.psswd;
    var query = "SELECT * FROM users WHERE login = ?";
    var arr = [loggedInUser.username];

    con.query(query, arr, function (err, result) {
        if (err) throw err;
        if (result.length > 0) {
            bcrypt.compare(pass, result[0].password, function (err, resPass) {
                if (resPass == true) {
                    res.setHeader('Content-Type', 'application/json');
                    res.send(JSON.stringify("success"));
                } else {
                    res.setHeader('Content-Type', 'application/json');
                    res.send(JSON.stringify("fail"));
                }
            });
        } else {
            res.setHeader('Content-Type', 'application/json');
            res.send(JSON.stringify("fail"));
        }
    });

});
```

Algoritmus 6 - *Kontrola shody zadaného hesla s heslem uloženým v databázi [autor, 2018]*

Výběr doporučených jídel

V první kapitole byly probrány základní modely doporučovacích systémů. Jelikož tato webová služba slouží pouze jako informační portál pro uživatele, nelze se v tomto případě spoléhat na explicitní hodnocení uživatelů. Jelikož všechny informace o denních menu jsou na domovské stránce této webové služby, uživatel nemusí klikat na žádná konkrétní jídla, tudíž lze také vyloučit implicitní hodnocení z chování uživatele při procházení webových stránek. Z toho důvodu nelze pro tyto účely využít kolaborativní filtrování, jelikož vytváření doporučení závisí na hodnocení ostatních uživatelů. V případě doporučovacích systémů založených na obsahu je proces doporučování závislý na vlastním hodnocení uživatele a vlastnostech položek, což je také z důvodu nepřítomnosti hodnocení vyloučeno. Tudíž doporučovací systém založený na znalostech je v tomto případě nejvhodnější variantou, protože jídla mohou být doporučena na základě vyplněných oblíbených kategorií uživatele. Tato webová aplikace při doporučování jídel upřednostňuje jídla, která obsahují nejvíce shodujících se kategorií mezi jídlem a oblíbenými

kategoriemi uživatele. Jelikož si uživatel může vybrat své oblíbené restaurace, mají jídla vyskytující se v restauraci, která patří mezi uživatelovi oblíbené, přiřazenou větší váhu než ostatní jídla. Tímto způsobem je zajištěno, aby jídla z oblíbených restaurací měla větší šanci dostat se na seznam doporučených jídel. Pro každé jídlo je spočítána hodnota důležitosti X_u pro daného uživatele u, pomocí váhy ω_u dané restaurace pro uživatele a počtu atributů a, které se nacházejí zároveň mezi atributy daného jídla a oblíbenými atributy uživatele.

$$X_u = \omega_u \times a$$

Pokud uživatel vyplnil alespoň jednu oblíbenou kategorii jídel, jsou mu na domovské stránce doporučena čtyři jídla s nejvyšší hodnotou důležitosti. Pro doporučení jídel je nejdříve nutné zjistit, které oblíbené kategorie si uživatel zvolil. Metoda *getFavourites()* zjišťuje seznam oblíbených kategorií uživatele a následně ho uloží do proměnné *favouriteRestaurants*. Pro získání doporučených jídel je poslán do databáze SQL dotaz, do kterého se dosazuje proměnná *favouriteList*. Tato proměnná obsahuje řetězec ve formě SQL zahrnující seznam oblíbených kategorií jídel uživatele. Tento postup popisuje následující algoritmus č. 7.

```
app.get('/getRecommendedMeals', function (req, res) {
    console.log('GET');
    var favouriteList = req.query.favouriteList;
    var onlyFavouriteRestaurants = req.query.onlyFavouriteRestaurants;

    if (onlyFavouriteRestaurants == "true") {
        var query = "select m.name, price, restaurant, restaurant_name, adress_city, adress_street, web, " +
            "(count(category_id)*if(isFavourite is null,1,1.5)) as weighted_score from meals m " +
            "join (select * from restaurant_ratings where user_login='" + loggedInUser.username + "') " +
            "rr on m.restaurant=rr.restaurant_phone_number " +
            "join restaurants r on r.phone_number=m.restaurant " +
            "join meal_categories mc on mc.meal_id=m.id_meal " +
            "where date=curdate() and (" + favouriteList + ")  " +
            "group by name order by weighted_score desc limit 4;";
    } else {
        var query = "select m.name, price, restaurant, restaurant_name, adress_city, adress_street, web, " +
            "(count(category_id)*if(isFavourite is null,1,1.5)) as weighted_score from meals m " +
            "left join (select * from restaurant_ratings where user_login='" + loggedInUser.username + "') " +
            "rr on m.restaurant=rr.restaurant_phone_number " +
            "join restaurants r on r.phone_number=m.restaurant " +
            "join meal_categories mc on mc.meal_id=m.id_meal " +
            "where date=curdate() and (" + favouriteList + ")  " +
            "group by name order by weighted_score desc limit 4;";
    }

    con.query(query, function (err, result) {
        if (err) {
            res.setHeader('Content-Type', 'application/json');
            res.send(JSON.stringify(err));
        } else {
            res.setHeader('Content-Type', 'application/json');
            res.send(JSON.stringify(result));
        }
    });
});
```

Algoritmus 7 - *Získání seznamu nejvhodnějších jídel pro doporučení [autor, 2018]*

Tento SQL dotaz vrací seznam všech jídel, která obsahují alespoň jednu z kategorií oblíbených uživatelem. Tento seznam jídel je seřazen podle počtu kategorií konkrétního jídla, které se shodují s oblíbenými kategoriemi uživatele. Jelikož jsou uživateli doporučeny vždy čtyři jídla, z databáze jsou vybrána čtyři jídla s největším počtem shodujících se kategorií. Seznam jídel seřazený podle jejich důležitosti je zobrazen na následujícím obrázku č. 4.

```
37 • select (count(category_id)*if(isFavourite is null,1,1.5)) as weighted_score, m.name, price, restaurant,
38   restaurant_name, adress_city, adress_street, web from meals m
39   join (select * from restaurant_ratings where user_login='user') rr on m.restaurant=rr.restaurant_phone_number
40   join restaurants r on r.phone_number=m.restaurant
41   join meal_categories mc on mc.meal_id=m.id_meal
42   where date='2018-04-12' and (category_id=2 or category_id=7 or category_id=14 or category_id=16 or category_id=18
43   or category_id=22 or category_id=27 or category_id=33 or category_id=46 or category_id=48)
44   group by name order by weighted_score desc;
45
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

weighted_score	name	price	restaurant	restaurant_name
6.0	150g Pikantní masová směs svpaná sýrem. bramboráčky - 95	95 Kč	(+420) 732 663 999	Garáž
4.5	Pečené kuře na másle s vařeným bramborem a raičatovým salátkem Ľ /	94 Kč	(+420) 466 535 300	Evropa Restaurant & Pub
4.5	Pečená brambora plněná žampionovým ragú a směsí sýrů	87 Kč	(+420) 775 483 828	Gekon boulder bar
3.0	Trhaný listový salát se zeleninou. limetkovo-bylinkovou zálivkou a grilovaným hermelínem 120g /	94 Kč	(+420) 466 535 300	Evropa Restaurant & Pub
3.0	Vídeňský telecí řízek. bramborový salát 1371011	95 Kč	(+420) 466 613 118	Jídelna Karlovina
3.0	Bramborový salát našich kuchařů		(+420) 774 575 033	Restaurace Na Rozkoši u Kulatý Bábv...
3.0	150g Plněný paprikový lusk v raiské omáčce. houskový knedlík - 89	89 Kč	(+420) 732 663 999	Garáž
3.0	Smažený hermelín. brambor MM. tatarská omáčka 1371011	85 Kč	(+420) 466 613 118	Jídelna Karlovina
3.0	Zapečené uzenářské brambory		(+420) 774 575 033	Restaurace Na Rozkoši u Kulatý Bábv...
3.0	Hovězí burger s raičetem. okurkou. sázeným veicem a dijonským dresinkem. Coleslaw salát 120g /	115 Kč	(+420) 466 535 300	Evropa Restaurant & Pub
3.0	250g Sedlák Burger s flákotou uzené sekané. hranolky - 99	99 Kč	(+420) 732 663 999	Garáž
3.0	Vepřové žebírko na pivu a zázvoru. šťouchaný brambor 1612	85 Kč	(+420) 466 613 118	Jídelna Karlovina
3.0	Steak z marinované krkovice. mačkané brambory.		(+420) 604 740 780	Dobrá kantýna
3.0	Dřevorubecká vepřová kotleta se slaninou. cibulí. americkým bramborem a chilli dresinkem 120g /	96 Kč	(+420) 466 535 300	Evropa Restaurant & Pub
3.0	300g Studený těstovinový salát s hermelínem a vaiíčkem - 89	89 Kč	(+420) 732 663 999	Garáž
3.0	Hermelín v bramboráku		(+420) 774 575 033	Restaurace Na Rozkoši u Kulatý Bábv...
3.0	150g Smažený sýr se šunkou. vařený brambor. tatarská omáčka - 97	97 Kč	(+420) 732 663 999	Garáž
3.0	PO CELÝ TÝDEN Smažený sýr s vařeným bramborem a domácí tatarkou 120g /	96 Kč	(+420) 466 535 300	Evropa Restaurant & Pub
1.5	300g Zeleninový salát s tuňákem a vařeným veicem	85 Kč	(+420) 466 746 913	Restaurace Pivovarka
1.5	Vařený brambor s máslem		(+420) 774 575 033	Restaurace Na Rozkoši u Kulatý Bábv...
1.5	Kuřecí / vepřový řízek s vařeným bramborem / hranolkami 120g /	96 Kč	(+420) 466 535 300	Evropa Restaurant & Pub

Obrázek 4 - *Seznam jídel seřazený podle hodnoty důležitosti pro doporučení [autor, 2018]*

2.2 Použité technologie

2.2.1 Java

Technologie Java byla vynalezena v roce 1995 Jamesem Goslingem, který pracoval pro Sun Microsystems. Je to programovací jazyk pro různé platformy, který odvozuje syntaxi z programovacích jazyků C a C++. Java byla původně navržena pro využití u mobilních telefonů, ale později se hlavní zaměření přesunulo na internet, kde poskytovala uživatelům interaktivitu pomocí animovaných webových stránek. Kompilovaný kód Java je možný spustit na různých operačních systémech, jako například Windows, Linux a Mac OS. Java patří mezi nejpoužívanější programovací jazyky a je několik důvodů proč využívat právě Javu.

- *Jednoduchost.* Základy Javy vzešly z programovacího jazyka C++. Jelikož je syntaxe jazyka C++ složitější, tak se Java zaměřila na vylepšení syntaxe a tím dosáhla větší jednoduchosti při práci s Javou.
- *Spolehlivost*. Zavedením objektově-orientovaného programování se Java snaží předejít fatálním chybám programátorů.

- *Bezpečnost*. Jelikož byla Java původně vytvořena pro telefony, které přenášely přes síť data, byl při jejím vývoji kladen velký důraz na bezpečnost. Do dnešní se Java považuje za nejbezpečnější jazyk.
- *Nezávislost na platformě.* Java byla navržena tak, aby byla přenosná mezi platformami a aby jí nezáleželo na použitém operačním systému, hardwaru nebo na kterých zařízeních běží. [9] [10] [11]

2.2.2 NetBeans

NetBeans začalo jako studentský projekt v České republice v roce 1996. Cílem bylo napsat vývojové prostředí v Javě. V dnešní době je NetBeans open-source software poskytující vývojové prostředí pro několik programovacích jazyků a frameworků a umožňuje jednoduchou implementaci Java desktopových, mobilních a webových aplikací nebo také HTML5, CSS a JavaScript aplikací. Existuje několik důvodů proč využívat právě vývojové prostředí NetBeans.

- *Nejlepší podpora pro nejnovější technologie Java.* Jelikož je NetBeans oficiálním vývojovým prostředím Javy, je možnost rychle a jednoduše aktualizovat aplikace do novějších verzí Javy.
- *Rychlá a chytrá úprava kódu.* Editor NetBeans pomáhá programátorům s doplňováním kódu, kontrolou sémantiky a syntaxe, poskytováním vzorů kódu nebo automatickým generováním kódu.
- *Snadná tvorba uživatelského prostředí GUI.* Pro vytváření uživatelského prostředí má NetBeans vlastní editor, kde jednotlivé objekty GUI lze vkládat pomocí drag-and-drop a editor se poté postará, aby daný objekt měl správné ohraničení a mezery mezi dalšími objekty. [12]

2.2.3 MySQL

MySQL je díky svému výkonu, spolehlivosti a jednoduchosti nejpoužívanější open-source databáze na světě, používaná nejznámějšími internetovými sítěmi jako například Facebook, Twitter, YouTube nebo Yahoo. MySQL je nezávislá na operačním systému a může mít využití

v široké škále aplikací, ale nejčastěji je tato databáze používána pro webové aplikace. MySQL je také součástí vývojové platformy LAMP, která používá Linux jako operační systém, Apache jako webový server, databázi MySQL a objektově orientovaný skriptovací jazyk PHP. MySQL je napsána v jazyce C a C++ a pro práci s databází jsou používané standartní SQL dotazy. Do roku 2016 byla hlavním rozdílem mezi SQL a MySQL kompatibilita mezi operačními systémy, kde SQL databáze byla do tohoto roku kompatibilní pouze s Windows. [13] [14]

2.2.4 JavaScript

JavaScript je programovací jazyk určený pro vytváření interaktivních webových stránek. Byl vytvořen v roce 1995 společností Netscape a jeho původní jméno bylo LiveScript. Jelikož Java nabývala na popularitě, tak byl kvůli marketingovému tahu změněn název na JavaScript. Velkou výhodou JavaScriptu je možnost využívání API (Application Programming Interface). API jsou připravené bloky kódu, které umožňují vývojáři implementovat programy, které by pro něj mohly být velice obtížné, nebo dokonce nemožné implementovat. JavaScript je navrhnut, aby pracoval se značkovacím jazykem HTML, kde se HTML stará o základní strukturu stránky a JavaScript o její funkčnost. [15] [16] [17]

2.2.5 Node.js

Node.js je platforma, která pomáhá rychlému a snadnému vývoji serverových a webových aplikací. Node.js se skládá z JavaScript engine od společnosti Google a několika standartních knihoven. Je to open-source multiplatformní software. Aplikace v Node.js jsou psány v programovacím jazyce JavaScript a mohou být spuštěny na OS X, Windows a Linuxu. Existuje několik důvodů, proč v dnešní době častěji programátoři využívají tuto platformu.

- *Asynchronní*. Všechny API z knihoven Node.js jsou asynchronní. To znamená, že server založený na Node.js nikdy nečeká, než mu API vrátí data. Zde je používán mechanismus událostí, které zajišťují předání odpovědí z předešlých API.
- *Rychlost*. Díky tomu, že je tato platforma sestavená na JavaScript Engine od společnosti Google, jsou knihovny Node.js schopny velmi rychle přeložit kód.

- *Žádné bufferování*. Aplikace Node.js nikdy nebufferují data, nýbrž odesílají výstupová data po částech. [18]

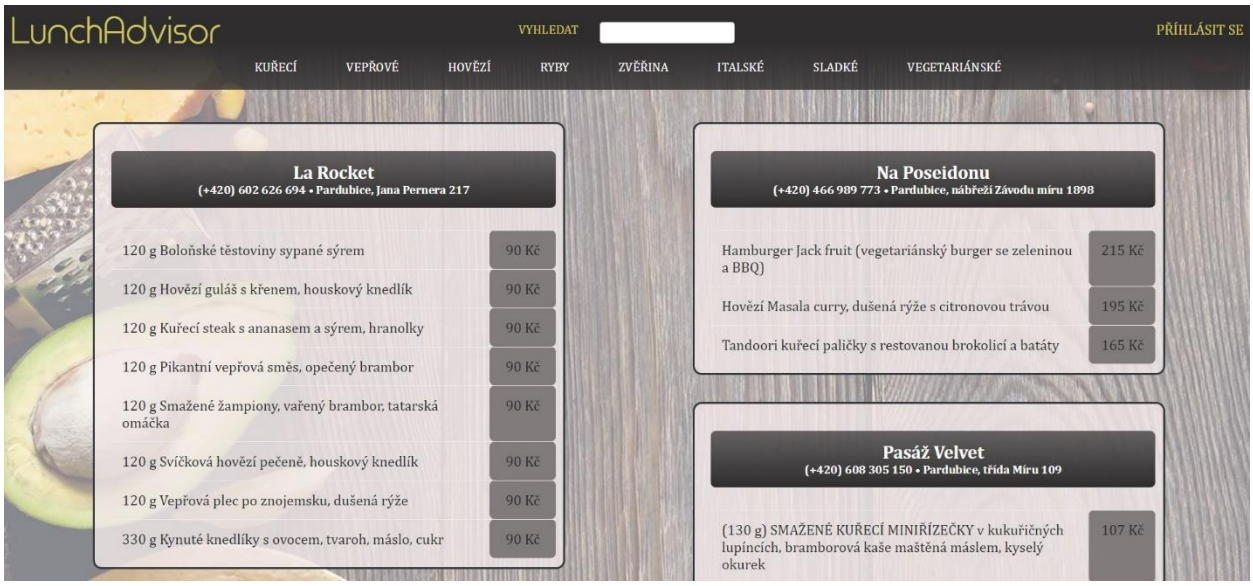
2.2.6 JSON

JSON (JavaScript Object Notation) je typ formátu pro přenos dat, který je nezávislý na programovacím jazyce a má přehledný a čitelný formát pro člověka i pro systém. JSON využívá konvence známých jazyků z rodiny C (C, C++, Java, JavaScript a další). JSON je založený na dvou strukturách, kde první je kolekce párů název/hodnota (objekt, struktura, hash tabulka a další) a druhá struktura je seřazený seznam hodnot (pole, vektor, seznam, posloupnost). Tyto univerzální struktury jsou podporovány většinou programovacích jazyků. [19]

2.3 Funkce webové aplikace

2.3.1 Domovská stránka

Domovská stránka nabízí kompletní seznam dostupných restaurací s jejich aktuálním menu. Restaurace jsou zobrazeny v jednotlivých blocích, kde je vypsán název restaurace, telefonní číslo a adresa restaurace. Ve výpisu denního menu jsou zobrazeny názvy jídel a jejich cena, pokud byla zadána poskytovatelem. Pro zjednodušení vyhledávání jídel pro uživatele je na navigačním menu, které slouží pro filtrování jídel, umístěno osm základních kategorií jídel. Tyto kategorie slouží například pro vypsání jídel, která spadají do zvolené kategorie. Kliknutím na logo webové služby umístěné v levém horním rohu obrazovky se uživatel dostane zpět na domovskou stránku, zobrazující kompletní výpis všech jídel. Zároveň má uživatel možnost vyhledávat jídla pomocí fulltextového vyhledávače v případě, že nemá zájem o výpis jídel dané kategorie, ale vyhledává konkrétní pokrm. Vyhledávač je nezávislý na diakritice. Náhled domovské stránky uvádí obrázek č. 5.



Obrázek 5 - *Domovská stránka [autor, 2018]*

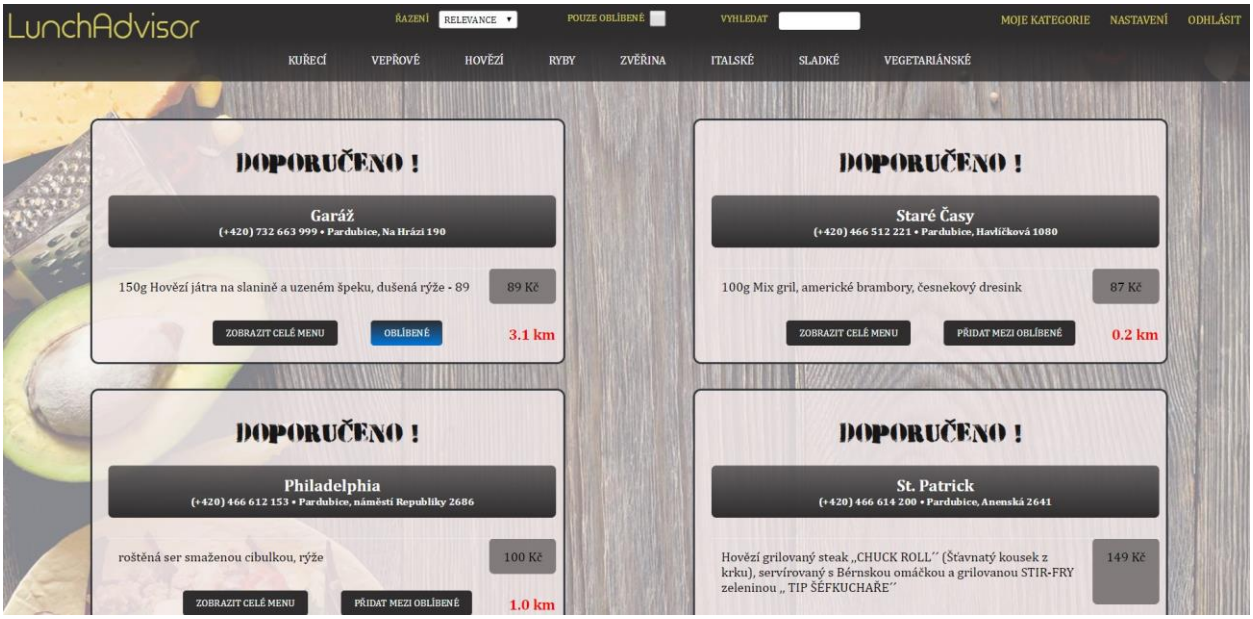
2.3.2 Vytvoření nového uživatele

Při vytváření nového uživatele je nutné vyplnit formulář, který obsahuje několik polí s povinnými a nepovinnými údaji. Mezi povinné údaje patří přihlašovací jméno a heslo, které musí obsahovat minimálně osm znaků z důvody bezpečnosti hesla. Heslo se zadává ve dvou odlišných polích, za účelem kontroly shody zvoleného hesla. Tímto krokem se také předchází případnému chybnému zápisu zvoleného hesla, jelikož existuje malá pravděpodobnost, že by uživatel zadal své heslo špatně dvakrát. Mezi nepovinné údaje patří jméno, příjmení a adresa uživatele. Pokud má systém zadanou adresu uživatele, tak výpis restaurací obsahuje zároveň vzdálenosti daných restaurací od zadané adresy uživatele. Posledním nepovinným údajem je výběr oblíbených kategorií jídel, jejich přípravy či přísad, které daná jídla obsahují.

2.3.3 Doporučení jídel

Pokud uživatel vyplnil při registraci oblíbené kategorie jídel nebo si tento seznam kategorií definoval v nastavení oblíbených kategorií, jsou každému přihlášenému uživateli na domovské stránce nabídnuta čtyři jídla, která byla zvolena na základě podobnosti mezi oblíbenými

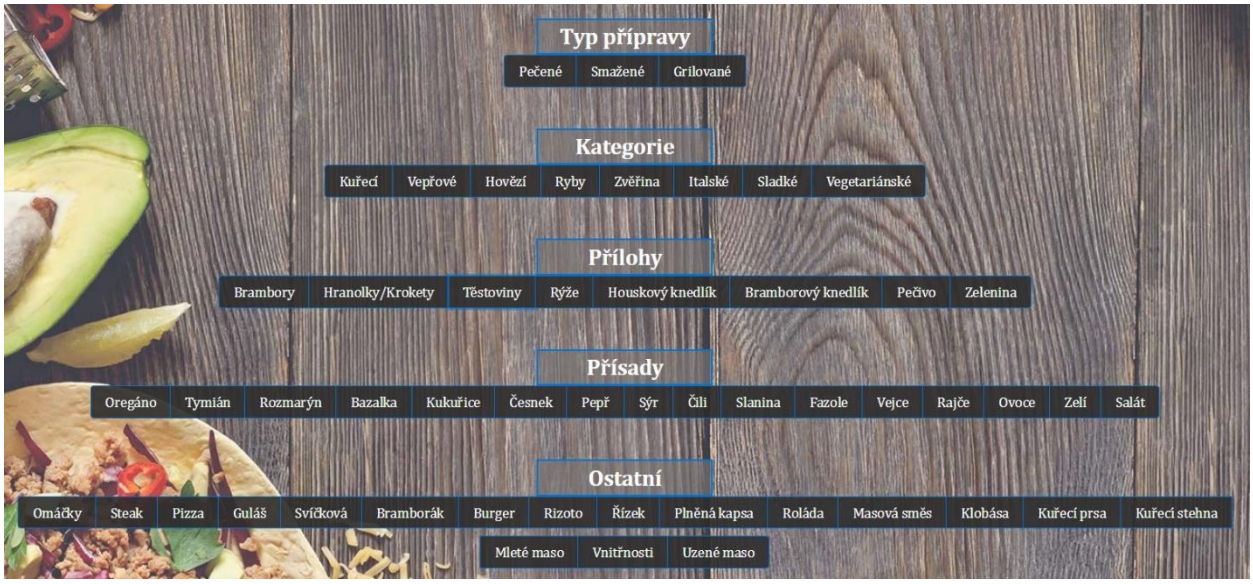
kategoriemi uživatele a atributy jídel. Každé doporučené jídlo je zobrazeno v samostatném bloku restaurace, kde nejsou vypsána ostatní jídla dané restaurace. Pro zobrazení všech aktuálních jídel dané restaurace slouží tlačítko "*ZOBRAZIT CELÉ MENU*" ve spodní části bloku. Náhled obrazovky přihlášeného uživatele je zobrazen na obrázku č. 6.



Obrázek 6 *- Obrazovka přihlášeného uživatele se selekcí doporučených jídel [autor, 2018]*

2.3.4 Selekce oblíbených kategorií jídel

Seznam oblíbených kategorií má přihlášený uživatel možnost editovat kliknutím na tlačítko "*MOJE KATEGORIE*" v pravé horní části domovské obrazovky. Zde je mu zobrazen kategorizovaný výpis všech kategorií. Modře označené položky indikují, že daná kategorie patří mezi oblíbené kategorie uživatele. Kliknutím na danou kategorii uživatel určuje, zda kategorii chce mezi svými oblíbenými kategoriemi či ne. Pro uložení úprav kategorií slouží tlačítko ve spodní části obrazovky "*ULOŽIT ZMĚNY*". Náhled na selekci oblíbených kategorií jídel uvádí obrázek č. 7.



Obrázek 7 *- Selekce oblíbených kategorií jídel [autor, 2018]*

2.3.5 Úprava uživatelských údajů

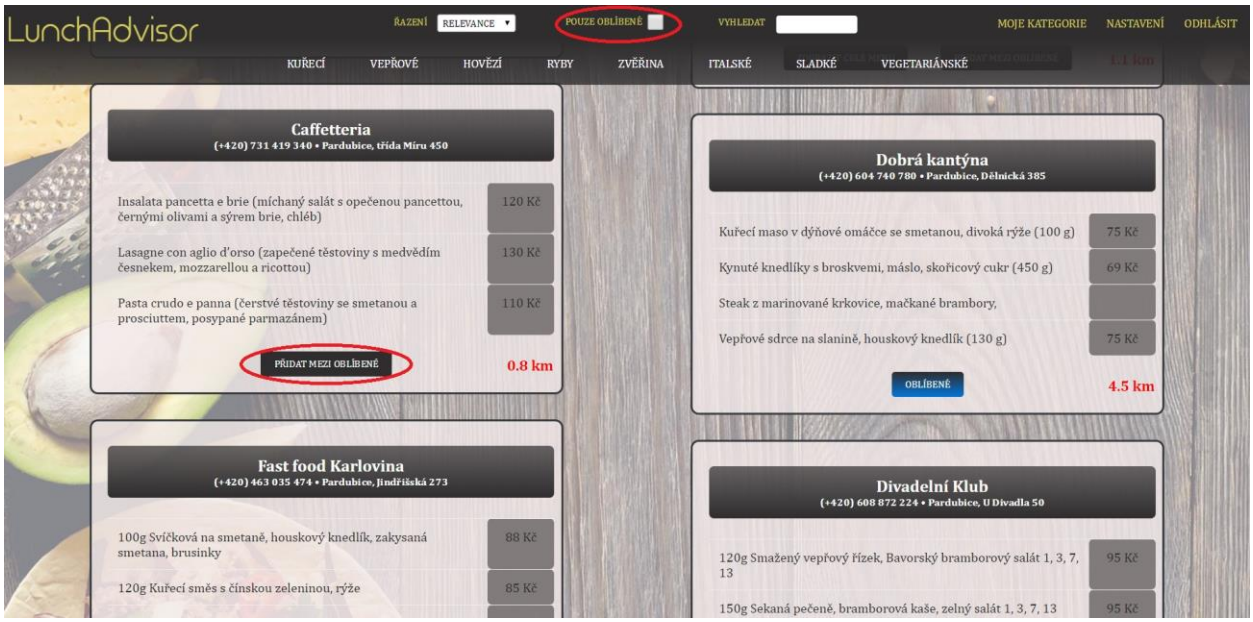
Do formuláře pro editaci uživatelských údajů se přihlášený uživatel dostane kliknutím na tlačítko "*Nastavení*". V první části formuláře nazvané "*Osobní údaje*" může uživatel změnit svou adresu, jméno a příjmení. Druhá část slouží pro změnu stávajícího hesla, kde musí uživatel vyplnit své původní heslo a nové heslo zároveň s jeho druhým zápisem pro kontrolu. Tlačítko "*Uložit změny*" na spodní části obrazovky slouží pro uložení editace uživatelského profilu.

2.3.6 Řazení restaurací

Jelikož je ve výpisu restaurací velké množství těchto bloků, má uživatel možnost zobrazené výsledky seřadit dvěma způsoby. První způsob je řazení podle relevance, který restaurace seřadí podle jejich celkové oblíbenosti mezi všemi uživateli. Druhou možností je řazení podle vzdálenosti, které seřadí restaurace podle vzdálenosti od adresy zadané v uživatelském profilu.

2.3.7 Oblíbené restaurace

Každý uživatel má možnost zvolit si své oblíbené restaurace. Na jeho domovské obrazovce může poté seznam restaurací filtrovat pouze na výpis menu jeho oblíbených restaurací pomocí zaškrtnutí tlačítka "*POUZE OBLÍBENÉ*". Jednotlivé restaurace si uživatel může přidávat nebo odebírat ze seznamu oblíbených restaurací tlačítkem "*PŘIDAT MEZI OBLÍBENÉ*", které je umístěné ve spodní části každého bloku restaurace. Tyto operace jsou naznačeny na obrázku č. 8.



Obrázek 8 *- Možnosti práce s oblíbenými restauracemi [autor, 2018]*

3 Závěr

Jedním z cílů této práce bylo seznámení se základní teorií doporučovacích systémů, tvořící důležitou část na moderním internetovém průmyslu, díky jejich pozitivnímu dopadu na uživatele internetu. Těm tyto systémy pomáhají při vyhledávání produktů. Zároveň pomáhají webovým poskytovatelům služeb, kteří díky doporučování produktů zákazníkům dosahují efektivnějších obchodních výsledků. V práci byla nejprve stručně vysvětlena historie doporučovacích systémů a jejich příklady v praxi zároveň s konkrétními příklady využití těchto systému. Dále byly představeny základní modely doporučovacích systémů a podrobně popsány jednotlivé metody, používáné při návrhu doporučených položek. Pro představu v jakých případech tyto systémy používat, bylo u každého modelu popsáno, za jakých podmínek je vhodné konkrétní model použít současně s jeho výhodami a nevýhodami.

Tyto znalosti byly následně využity pro návrh a implementaci webové aplikace, která byla hlavním cílem této práce. Tato práce měla za úkol vytvořit webovou stránku, která slouží pro usnadnění výběru jídel z denních menu restaurací v Pardubicích pomocí možnosti filtrování jídel podle různých atributů a doporučováním jídel uživatelům na základě jejich preferencí. Aby byl tento cíl splněn, byla navržena aplikace složená ze dvou modulů. První modul využívá metod data miningu, získává a analyzuje jednotlivé názvy jídel pomocí klíčových slov. Díky tomu jsou získány atributy popisující informace o analyzovaných pokrmech. Získání těchto atributů bylo klíčové pro možnost využití doporučovacích systémů. Druhý modul aplikace vytvořil webovou stránku a obstarává všechny její dynamické prvky. Zajišťuje výpis denních menu a umožňuje jejich filtraci podle vybraných atributů. Zároveň poskytuje přihlášeným uživatelům doporučení jídel, která jsou vytvořena na základě jejich preferencí. Po prozkoumání teorie doporučovacích systémů byl vybrán jako nejvhodnější model doporučovací systém založený na znalostech. Tento model byl zvolen z důvodu nezávislosti na hodnocení uživatelů, s využitím uživatelem zadaných preferencích. Získání implicitních či explicitních hodnocení by na této webové stránce bylo prakticky nemožné, protože tato stránka slouží pro uživatele pouze jako informační, tudíž se nedalo očekávat, že by uživatelé hodnotili jednotlivá jídla.

Tato aplikace slouží pouze jako základní model pro znázornění využití doporučovacích systémů v této oblasti. Existuje několik způsobů, jak tuto práci v budoucnosti vylepšit. V dnešní době je nejjednodušší spojení s internetem pomocí mobilního telefonu, tudíž implementace mobilní aplikace by byla značným zjednodušením přístupu uživatelům k této aplikaci. Jelikož uživatelé mají pouze možnost vyplnit své oblíbené kategorie, uživatelům jsou doporučovány jídla bez ohledu na to, zda obsahují ingredience, které nepatří mezi uživatelovi oblíbené. Implementace selekce kategorií neoblíbených jídel a jejich následné zohlednění při doporučování jídel by vylepšilo kvalitu této aplikace. Dalším krokem vylepšení by byla implementace mapy, která by zobrazovala veškeré dostupné restaurace současně s aktuální adresou uživatele.

4 Zdroje

[1] AGGARWAL, Charu C. *Recommender Systems: The Textbook*. Switzerland: Springer International Publishing, 2016. ISBN 978-3-319-29659-3.

[2] MELVILLE P., SINDHWANI, V., Recommender Systems [online]. In Encyclopedia of Machine Learning, 2010. [cit. 20. 11. 2018]. Dostupné z WWW:<http://www.prem-melville.com/publications/recommender-systems-eml2010.pdf>.

[3] RICCI, Francesco, Lior ROKACH, Bracha SHAPIRA a Paul B. KANTOR. *Recommender systems handbook*. Springer US. New York: Springer, c2011. ISBN 978-0-387-85820-3.

[4] MOLINA FERNÁNDEZ, Leidy Esperanza. *Recommendation System for Netflix* [online]. Amsterdam, 2018 [cit. 2018-11-20]. Dostupné z WWW: https://beta.vu.nl/nl/Images/werkstuk-fernandez_tcm235-874624.pdf. Reserch paper. VRIJE UNIVERSITEIT AMSTERDAM.

[5] JANNACH, D., ZANKER, M., FELFERNIG, A., FRIEDRICH, G., *Recommender Systems: An Introduction*. Cambridge University Press, 2010. ISBN: 978-0521493369.

[6] PINELA, Carlos. *Recommender Systems—User-Based and Item-Based Collaborative Filtering* [online]. 6.11.2017 [cit. 2018-11-20]. Dostupné z WWW: https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f

[7] ALMAZRO, D., SHAHATAH, G., ALBDULKARIM, L., KHEREES, M., MARTINEZ, R., NZOUKOU, W., A Survey Paper on Recommender Systems [online]. Datum poslední revize 24. 2. 2010 [cit. 2018-11-20]. Dostupné z WWW: https://arxiv.org/pdf/1006.5278v4.pdf

[8] *Model-based recommendation systems* [online]. [cit. 2018-11-20]. Dostupné z WWW: http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/modelbased.html

[9] LEATHY, Paul. *What Is Java?: Java is built on C++ for a simple-to-use language* [online]. 9.4.2018 [cit. 2018-11-20]. Dostupné z WWW: https://www.thoughtco.com/what-is-java-2034117

[10] *What is Java technology and why do I need it?* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://java.com/en/download/faq/whatis_java.xml

[11] Java. *Techopedia* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://www.techopedia.com/definition/3927/java

[12] *Welcome to the Netbeans Community* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://netbeans.org/about/index.html

[13] ROUSE, Margaret. *MySQL* [online]. 2018 [cit. 2018-11-20]. Dostupné z WWW: https://searchoracle.techtarget.com/definition/MySQL

[14] *About MySQL* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://www.mysql.com/about/

[15] CHAPMAN, Stephen. *Introduction to JavaScript* [online]. 14.6.2018 [cit. 2018-11-20]. Dostupné z WWW: https://www.thoughtco.com/what-is-javascript-2037921

[16] *What is JavaScript: Introduction to JavaScript* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://www.dofactory.com/tutorial/what-is-javascript

[17] *What is JavaScript* [online]. 18.11.2018 [cit. 2018-11-20]. Dostupné z WWW: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript

[18] *Node.js - Introduction* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm

[19] *Introducing JSON* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://www.json.org/

[20] BOTERHOVEN, Daniel. *Why You Should Use Bcrypt to Hash Passwords* [online]. 7.12.2016 [cit. 2018-11-20]. Dostupné z WWW: https://medium.com/@danboterhoven/why-you-should-use-bcrypt-to-hash-passwords-af330100b861

[21] *Why you should use Bcrypt to hash passwords* [online]. [cit. 2018-11-20]. Dostupné z WWW: https://viewport-tech.com/blog/why-you-should-use-bcrypt-to-hash-passwords/