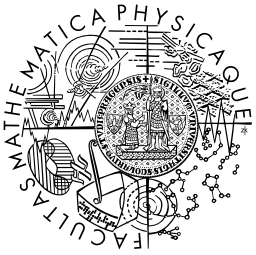


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Kristýna Lhoťanová

Webová aplikace pro vyhledávání receptů

Katedra softwarového inženýrství

Vedoucí bakalářské práce: doc. Mgr. Martin Nečaský, Ph.D.
Studijní program: Informatika
Studijní obor: Programování a vývoj software

Praha 2022

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Poděkování.

Název práce: Webová aplikace pro vyhledávání receptů

Autor: Kristýna Lhoťanová

Katedra: Katedra softwarového inženýrství

Vedoucí bakalářské práce: doc. Mgr. Martin Nečaský, Ph.D., Katedra softwarového inženýrství

Abstrakt: Cílem této práce je vyvinout webovou aplikaci pro vyhledávání receptů založenou na agregaci datových sad z existujících webových stránek s recepty a jejich obohacení o data ze znalostních grafů. Znalostní grafy byly zastoupeny projekty DBpedia a Wikidata, z nichž byla získána data o ingrediencích a kategoriích jednotlivých receptů. Data byla extrahována s pomocí knihovny Apify a v dokumentovém modelu uložena do databázového systému Apache CouchDB. Aplikace uživateli poskytuje různé možnosti filtrování výsledků včetně fasetového vyhledávání, k čemuž využívá platformu Apache Solr. Zaměřuje se zejména na vyhledávání dle ingrediencí. Jedná se o tzv. single-page aplikaci implementovanou pomocí JavaScriptové knihovny React pro uživatelské rozhraní a frameworku Express.js na straně serveru. Obě části aplikace jsou psány staticky typovaným jazykem TypeScript a komunikují spolu prostřednictvím REST API.

Klíčová slova: webová aplikace, recept, znalostní graf, propojená data

Title: Web application for searching recipes

Author: Kristýna Lhoťanová

Department: Department of Software Engineering

Supervisor: doc. Mgr. Martin Nečaský, Ph.D., Department of Software Engineering

Abstract: Abstract.

Keywords: web app, recipe, knowledge graph, linked data

Obsah

Úvod **3**

1 Analýza **6**
1.1 Požadavky aplikace . . . 6
1.1.1 Funkční požadavky . . . 6
1.1.2 Nefunkční požadavky . . . 7
1.1.3 Uživatelské příběhy . . . 9
1.1.4 Obrazovky uživatelského rozhraní . . . 10
1.2 Dostupné datové sady . . . 11
1.2.1 Recipe1M+ . . . 11
1.2.2 Open Recipes . . . 15
1.2.3 FoodKG . . . 15
1.2.4 Food.com Recipes and Interactions . . . 15
1.2.5 Generování vlastního datasetu . . . 16
1.2.6 DBpedia . . . 18
1.2.7 Wikidata . . . 19
1.3 Úspěšné webové aplikace s recepty . . . 19
1.3.1 Allrecipes . . . 20
1.3.2 Supercook . . . 20
1.3.3 MyFridgeFood . . . 21
1.3.4 Taste of Home . . . 21
1.3.5 Food.com . . . 22

2 Architektura řešení **23**
2.1 Příprava dat . . . 23
2.1.1 Extrakce dat . . . 23
2.1.2 Čištění dat . . . 26
2.2 Databázový model . . . 28
2.3 Indexy . . . 28
2.4 Backend . . . 28
2.5 Frontend . . . 28
2.6 Aplikační logika . . . 28

3 Implementace návrhu **29**
3.1 Zpracování vstupních dat . . . 29
3.2 Databáze Apache CouchDB . . . 29
3.3 Vyhledávání pomocí Apache Solr . . . 29
3.4 Rozhraní REST API . . . 29
3.5 Middleware . . . 29
3.6 Single-page aplikace . . . 29

4 Testování **30**

Závěr **31**

Seznam použité literatury **32**

Seznam obrázků **33**

Seznam tabulek **34**

A Přílohy **35**
A.1 První příloha 35

Úvod

Vyhledávání relevantního obsahu je spolu s elektronickou komunikací jednou z klíčových funkcí internetu. S rostoucím množstvím dostupných informací se filtrování nalezených výsledků stává stále obtížnějším. Tvůrci webových stránek se často zaměřují spíše na uživatelsky přívětivé interaktivní rozhraní, zatímco optimalizace strojového vyhledávání jde stranou. Pro webové vyhledávače, jmenovitě např. Google, Bing nebo Yahoo, je pak náročné analyzovat obsah těchto stránek po sémantické stránce a tedy vyhodnotit, zda obsahují užitečné informace k zodpovězení dotazu uživatele.

V reakci na tuto problematiku vznikl tzv. *Sémantický Web* neboli Web dat jakožto rozšíření původního Webu dokumentů, tak jak jej známe z platformy *World Wide Web*. Sémantický Web lze vnímat jako globální databázi, nad kterou se lze pomocí speciálního jazyka *SPARQL* dotazovat podobně jako nad tradičními databázovými systémy. Data jsou poskytována v různých serializacích formátu RDF a mohou být přímo vnořena do HTML dokumentů nebo zpřístupněna v samostatných souborech. Tato strukturovaná data nazýváme *propojená* (v originále *Linked Data*). Umožňují snadnější hledání souvislostí mezi entitami z různých zdrojů na základě společných slovníků neboli ontologií [1].

V posledních letech termín Sémantický Web ustupuje do pozadí a často je místo něj zmiňován tzv. *znalostní graf* (anglicky *Knowledge Graph*). Začátky fenoménu znalostních grafů bychom mohli datovat do roku 2012, kdy společnost Google představila svůj znalostní graf pro vyhledávání obsahu na webu. K technologii znalostních grafů se brzy poté přihlásily další velké společnosti včetně firem Microsoft, IBM, Facebook, LinkedIn, Amazon, eBay, Airbnb nebo Uber. Grafový model totiž oproti tradičnímu relačnímu modelu nabízí flexibilnější správu dat z oblasti sociálních sítí, dopravních spojení, bibliografických citací a řady dalších odvětví. Výše zmíněné příklady znalostních grafů všechny spadají do kategorie komerčních znalostních grafů, které jsou určeny pro interní využití v rámci dané firmy. Protikladem jim jsou otevřené znalostní grafy poskytující data k volnému využití všem uživatelům internetu. Nejvýznamnějšími představiteli otevřených znalostních grafů jsou aktuálně DBpedia, Wikidata, Freebase a YAGO [2]. První dva zmíněné projekty si představíme v této práci a integrujeme je s aplikací na vyhledávání receptů.

Oblast gastronomie je rozvěž vhodným kandidátem k zapojení do sítě znalostních grafů a propojených dat. Pro tvůrce webových aplikací je poměrně jednoduché publikovat obsah svých stránek ve formátu strukturovaných dat. Vhodným způsobem je např. vložení RDF reprezentace daných entit (receptů, uživatelů, recenzí) ve formátu JSON-LD[1] přímo do hlavičky jednotlivých HTML dokumentů. V takovém případě je žádoucí použít existující ontologie raději než definovat vlastní, byť by mohly být lépe strukturované a uzpůsobené dané doméně. Využití standardizovaných slovníků usnadňuje webovým vyhledávačům interpretaci stránky a je větší šance, že se aplikace dostane na vyšší příčky vyhledávaných výsledků.

Cílem této bakalářské práce je prozkoumat možnosti využití otevřených dat v doméně receptů, propojit je s daty publikovanými na různých webových strán-

[1]Koncovka *LD* v názvu JSON-LD odkazuje na pojem Linked Data.

kách shromažďujících recepty a prezentovat tyto výsledky uživateli ve formě vlastní webové aplikace. Zároveň v rámci této aplikace poskytnout užitečné možnosti filtrování agregovaných výsledků včetně fasetového vyhledávání. Proces sběru, konverze a uložení dat by měl být co nejvíce automatizovaný a snadno zreprodukovatelný. Práce se nevěnuje přidávání nových receptů prostřednictvím uživatelského rozhraní. Existujících webové stránky totiž obsahují velké množství dat, které lze díky bohaté historii v podobě hodnocení a recenzí lépe filtrovat. Navíc by bylo potřeba se vypořádat s automatickou kalkulací nutričních hodnot receptu z obsažených surovin, přičemž ne všechny ingredience dokážeme automaticky identifikovat a získat jejich nutriční hodnoty. V budoucnu by funkce nahrávání nových receptů měla být přidána spolu s více lokalizacemi aplikace, registrací uživatelů a celkovou personalizací obsahu pro přihlášené uživatele. Dále se práce v této fázi nezabývá nasazením, neboť by vyžadovalo větší výpočetní kapacitu a skupinu IP adres na získání dostatečně velkého množství dat a také poměrně robustní databázi pro uložení extrahovaných dat.

Volba tématu

Příprava jídla je tématem každodenního života a na webových stránkách, které se této oblasti věnují, má velmi silnou komunitu. Většina z nás se chystání domácích pokrmů z ekonomických důvodů nevyhne, takže se hodí mít po ruce sadu receptů pro inspiraci. Typicky máme na recepty různé požadavky — někdo preferuje rychlejší postup, jiný se dívá po ceně ingrediencí nebo nutričních hodnotách. Občas dostaneme chuť na recept z řecké nebo italské kuchyně a jindy zkrátka chceme experimentovat a najít recept kombinující našich 5 oblíbených surovin. Některé ingredience z receptu nám mohou být neznámé, nebo si podle samotného názvu nejsme jistí, zda máme na mysli tu správnou. V takovém případě musíme stránku s receptem opustit a dodatečné informace k ingredienci vyhledat jinde, pokud na ně aplikace přímo neodkazuje. Zde je příležitost zapojit otevřená data a namapovat názvy ingrediencí na jejich odpovídající entity ve znalostních grafech. Data pak můžeme začlenit do aplikace a nabídnout uživateli informace nad rámec samotného receptu, např. popisy a glykemické hodnoty surovin, ilustrační obrázky a podobně. Také můžeme identifikovat ingredience a tranzitivně recepty ze stejných kategorií. Oproti původní datové sadě tak vytvoříme nové vazby a poskytneme uživateli rozmanitější filtrování výsledků.

Doména receptů navíc poskytuje spoustu prostoru pro zajímavá rozšíření se zapojením moderních technologií. Uplatnění by zde našlo například počítačové vidění s rozpoznáváním obrázků. S dostatečně velkou databází bychom díky němu mohli analyzovat fotografii hotového pokrmu a nalézt příslušný recept. Usnadnili bychom tak uživateli práci v situacích jako je návštěva restaurace, při které návštěvníkovi zachutnalo servírované jídlo a chtěl by si jej později připravit v domácích podmínkách. Dalším uplatněním strojového učení by mohlo být vyhledávání na základě příkazů v přirozeném jazyce. Namísto zdlouhavého zadávání nejrůznějších filtrů by stačilo aplikaci položit dotaz: „Jaké recepty z italské kuchyně mohu vyrobit z kuřete, rajčat a parmazánu?“. Této problematice se věnuje například projekt FoodKG konstruující nad recepty a ingrediencemi znalostní graf [3]. Ruku v ruce s touto funkcionalitou jde hlasové zadávání, které by se hodilo zapojit nejen ve fázi vyhledávání receptů, ale také například pro hands-free ovládání aplikace.

Uživatel by měl možnost diktovat příkazy k přečtení části receptu, pokud zrovna pracuje na jeho přípravě a nemá volné ruce k listování obsahem. Využití by našlo i populární *full-text* vyhledávání, pomocí kterého lze snadno objevit recepty na základě klíčových slov v popisku receptu, postupu či recenzích. V komerční sféře by se nabízelo propojení s online supermarkety, konkrétně zrychlení nákupu pomocí vyhledávání surovin k vybranému receptu. S tímto konceptem již na svých stránkách pracuje firma rohlik.cz, nabídka receptů a možnosti filtrování jsou ale omezené. Nepochybně by se hodilo integrovat také doporučovací systém pro ještě snadnější nalezení relevantních výsledků. Aplikace má velký prostor pro škálování objemu dat, přičemž datasety mohou být následně použity jako podklad pro strojové učení.

1. Analýza

V této kapitole si zadefinujeme požadavky na funkcionalitu naší aplikace. Také se v kontextu požadavků podíváme na existující webové stránky s recepty a provedeme diskuzi nad jejich funkcemi, možnými vylepšeními a rozšířeními. Následně si rozebereme různé alternativy dostupných datových sad a srovnáme jejich výhody i nevýhody vzhledem k požadavkům aplikace.

1.1 Požadavky aplikace

Nyní si rozebereme požadavky na naši aplikaci, které můžeme rozdělit do skupin funkčních a nefunkčních požadavků. Funkční požadavky popisují konkrétní funkcionalitu systému, zabývají se vstupem od uživatele a prezentací výstupu. Díky tomu je lze poměrně snadno definovat a testovat jejich naplnění v hotové aplikaci. Nefunkční požadavky se naopak na konkrétní vstup nevážou a místo toho popisují vlastnosti a omezení, které by měl systém splňovat. Zjednodušeně lze říci, že funkční požadavky popisují, co má systém dělat, zatímco nefunkční požadavky specifikují, jaký má systém být [4].

1.1.1 Funkční požadavky

Následuje výčet funkcionalit, které by aplikace svým uživatelům měla nabídnout. Uživatelé mohou mít různé role od běžného návštěvníka stránky po administrátora nebo vývojáře integrujícího data do jiného systému.

Běžný uživatel

1. Aplikace poskytuje uživatelské rozhraní pro vyhledávání receptů na základě ingrediencí, klíčových slov, času přípravy, hodnocení a nutričních hodnot.
2. Aplikace umožňuje kombinovat libovolné množství vyhledávácích filtrů.
3. Aplikace podporuje zadávání vlastních i předdefinovaných ingrediencí prostřednictvím našeptávače.
4. Aplikace podporuje fasetové vyhledávání, tedy u nabízených možností zobrazuje počet receptů, které se po zvolení daného filtru zobrazí.
5. Aplikace poskytuje možnost smazání všech vyhledávacích filtrů jedním kliknutím, ale také mazání po jednom filtru.
6. Aplikace zobrazuje uživateli všechny nalezené výsledky bez omezení na maximální počet výsledků.
7. Aplikace při otevření vyhledávací obrazovky bez zadaných filtrů zobrazuje všechny recepty, které má v databázi.
8. Aplikace umožňuje zobrazení detailu receptu rozkliknutím nalezeného výsledku.

9. Aplikace zobrazuje pouze recepty s titulní fotografií.
10. Aplikace na vyhledávací stránce pro každý nalezený recept zobrazuje jeho název, popis, obrázek, čas přípravy, hodnocení a počet recenzí.
11. Aplikace nabízí náhledy všech ingrediencí u vyhledaných receptů a zvýrazňuje aktuálně vyhledávané ingredience.
12. Aplikace umožňuje listování nalezenými výsledky prostřednictvím systému stránkování, nikoli nekonečným posouváním stránky.
13. Aplikace plně podporuje navigaci v rámci historie prohlížeče včetně přidávání a odebírání filtrů i listování více stranami výsledků.
14. Aplikace na detailní stránce každého receptu zobrazuje název, hodnocení, počet recenzí, popis, čas přípravy, fotografii, ingredience, postup přípravy a nutriční hodnoty.
15. Aplikace zvýrazňuje ingredience na detailní stránce receptu, ke kterým má dodatečné informace.
16. Aplikace přesměrovává na obrazovku s detailem ingredience po kliknutí na zvýrazněnou ingredienci.
17. Aplikace zobrazuje na detailní stránce ingredience následující informace nebo jejich podmnožinu: název, popis, obrázek, nutriční hodnoty, náhrady, kategorie a níže recepty obsahující tuto ingredienci, které lze otevřít stejně jako z vyhledávací obrazovky.
18. Aplikace má nezávisle na otevřené stránce viditelný ovládací panel s možností navigace na vyhledávací obrazovku.

Externí systém

1. Aplikace poskytuje REST API endpointy pro získání dat k receptům a ingrediencím.
2. Aplikace zpřístupňuje JSON-LD reprezentaci dat v hlavičkách dokumentů s recepty a ingrediencemi.
3. Aplikace podporuje navigaci a vyhledávání receptů přes url adresy s query parametry.

1.1.2 Nefunkční požadavky

Požadavky z této kategorie lze dále dělit podle jejich zaměření. Některé se věnují výkonu aplikace, jiné spolehlivosti, přenositelnosti, bezpečnosti, využitým technologiím, vývojovému prostředí nebo platformě, testovatelnosti či rozšiřitelnosti. Oblastí je zde skutečně mnoho, uvedeme proto pouze výčet konkrétních požadavků na naši aplikaci.

1. Backend aplikace je postaven na frameworku Express.js pro Node.js prostředí.

2. Frontend aplikace je implementován pomocí knihovny React.
3. Backend i frontend aplikace jsou psány staticky typovaným jazykem TypeScript.
4. Aplikace využívá dokumentovou databázi Apache CouchDB pro uložení dat o receptech a ingrediencích.
5. Aplikace využívá systém Apache Solr pro implementaci vyhledávání receptů.
6. Aplikace využívá program Silk Workbench pro objevování linků mezi dvěma entitami.
7. Aplikace je implementovaná jako single-page aplikace s podporou routingu mezi více obrazovkami.
8. Aplikace integruje data z aspoň 2 různých veřejných znalostních grafů.
9. Aplikace pro komunikaci mezi klientem a serverem používá REST API v kombinaci s asynchronními requesty.
10. Uživatelské rozhraní aplikace je založeno na knihovně Material UI poskytující sadu univerzálních komponent pro React aplikace.
11. Uživatelské rozhraní aplikace je responzivní pro desktopová i mobilní zařízení.
12. Databáze obsahuje v prvotní fázi přes 50 000 receptů z aspoň 2 různých zdrojů.
13. Aplikace je škálovatelná co do množství poskytovaných dat.
14. Aplikace je škálovatelná z pohledu nových lokalizací a jejich distribuce.
15. Aplikace je připravena pro implementaci nových rozšíření bez nutnosti výrazné změny stávajícího kódu.
16. Vyhledávání receptů je pro nového uživatele přímočaré a filtrování zvládne nastavit v řádu vteřin až minut v závislosti na počtu požadovaných filtrů.
17. Zdrojový kód aplikace je open-source a verzovaný na platformě GitHub.
18. Zdrojový kód aplikace je přehledný a snadno rozšiřitelný dalšími vývojáři.
19. Komponenty aplikace jsou znovupoužitelné v rámci projektu i mimo něj.
20. Získání receptů pro jednu stránku výsledků trvá méně než 500 ms (200 ms dotaz na server, 200 ms doručení odpovědi klientovi a 100 ms rezerva).
21. Rendering jedné stránky vyhledaných receptů trvá méně než 1 000 ms od načtení dat do paměti.
22. Nově extrahovaná data se uživatelům aplikace zobrazí nejpozději do druhého dne.
23. Aplikace je kompatibilní s webovými prohlížeči Google Chrome, Mozilla Firefox a Microsoft Edge.

1.1.3 Uživatelské příběhy

Požadavky aplikace lze méně formálním způsobem popsat pomocí tzv. uživatelských příběhů, které vyjadřují přání a očekávání uživatele vůči aplikaci. Uživatel vyžaduje konkrétní funkcionalitu pro dosažení vybraného cíle. Uživatelské příběhy jsou důležitou součástí agilního vývoje, neboť kladou důraz na potřeby uživatele, které se v průběhu vývoje mohou vyvíjet a měnit. Obvykle příběhy zapisujeme v jednoduchém formátu: „Jako (role) chci (funkce)[, abych (cíl)]“ [5]. Následují příklady uživatelských příběhů v kontextu naší aplikace:

1. Jako kuchař, který má vybraných několik hlavních ingrediencí, chci najít všechny recepty obsahující tyto suroviny.
2. Jako milovník řecké a italské kuchyně chci použít filtrování receptů z těchto oblastí, abych nemusel procházet detaily receptů a hledat jejich původ v popisu.
3. Jako uživatel, který rád šetří čas, chci znát všechny ingredience daného receptu ještě před otevřením jeho detailu, abych se vyhnul čtení receptů s příliš mnoha dodatečnými ingrediencemi.
4. Jako zaneprázdněný student chci snadno najít recepty, které lze připravit za méně než 30 minut.
5. Jako celiak chci hledat pouze recepty bez obsahu lepku, abych nemusel procházet spousty receptů, které si nemohu připravit.
6. Jako vrcholový sportovec chci snadno najít recepty s vysokým obsahem bílkovin.
7. Jako nutriční poradce chci u receptů vidět podrobný rozpis nutričních hodnot, abych daný recept mohl doporučit svým klientům dle jejich stravovacích potřeb.
8. Jako hostitel očekávající návštěvu potřebuji znát počet porcí, které se základním množstvím surovin připravím, abych toto množství mohl přizpůsobit počtu hostů.
9. Jako zvídavý uživatel se chci při čtení receptu dozvědět zajímavosti o jeho ingrediencích.
10. Jako uživatel s vytříbeným vkusem chci hledat pouze recepty s maximálním hodnocením a s co největším počtem kladných recenzí.
11. Jako nerozhodný uživatel chci mít možnost rychlé změny vyhledávacích filtrů.
12. Jako uživatel, který našel zajímavý recept před několika dny, chci využít historii prohlížeče a najít recept dle názvu, abych nemusel vzpomínat na vyhledávací filtry, pomocí nichž jsem recept původně objevil.
13. Jako kuchař spokojený s připraveným pokrmem chci najít autora receptu a vyhledat jeho další recepty.

14. Jako uživatel s preferencí vzhledu Material Design bych rád pracoval s aplikací, která je na tomto stylu založená.

15. Jako vývojář externí aplikace s recepty bych rád jednoduše získal strukturovaná data receptů, abych každou informaci nemusel extrahovat přes jednotlivé CSS selektory.

16. Jako webový vyhledávač potřebuji informace k receptům ve strukturovaném formátu pro Linked Data, ideálně popsané dle entity `Recipe` z ontologie Schema.org.

1.1.4 Obrazovky uživatelského rozhraní

Přestože vyvíjíme single-page aplikaci, počítáme s více obrazovkami pro pohodlnější navigaci. S využitím knihovny React Router dokážeme simulovat existenci libovolného množství obrazovek a zároveň zůstat na jedné stránce bez potřeby opětovného načítání. Tím se odlišíme od tradičních statických aplikací, kterým při každé změně url včetně query parametrů musí server v odpovědi poslat odpovídající HTML obsah. Náš přístup má ovšem nevýhodu z pohledu strojového zpracování, neboť pro vygenerování obsahu stránky potřebujeme v prohlížeči spustit JavaScript kód. Tím znemožníme zpracování naší aplikace prostřednictvím pouhých HTTP requestů, což je podstatně jednodušší a ekonomičtější varianta ve srovnání s automatizací celého webového prohlížeče. Tento nedostatek ale kompenzujeme transparentním REST API, přes které si lze vyžádat strukturovaná data přímo přes HTTP requesty. Zároveň usnadníme automatické zpracování vyhledávačům, které automatizaci prohlížeče využívají, neboť v detailech receptů a ingrediencí zahrneme jejich JSON-LD reprezentaci.

Aplikaci složíme ze 3 základních uživatelských obrazovek: vyhledávání receptů, detail receptu a detail ingredience. Všechny obrazovky musí být responzivní a poradit si s proměnlivou velikostí obrazovky.

Vyhledávání receptů

Domovskou stránku bude tvořit vyhledávání receptů na základě různých kritérií. Primárně bude k dispozici výběr požadovaných ingrediencí, sekundárně filtry klíčových slov, kategorií, času přípravy, hodnocení a nutričních hodnot. Všechny filtry bude možné odstranit samostatně i najednou pomocí společného tlačítka pro smazání. Pro získání přesnějších výsledků bude při vyplňování filtrů k dispozici našeptávač, který zobrazí známé možnosti a spolu s nimi počty receptů, které jsou při výběru tohoto nastavení k dispozici. Výsledky budou zobrazovány na stránkách s 24 nebo 30 kartami receptů. Přepínání stránek bude umístěno standardně ve spodní části stránky a zároveň bude aktuální stránka figurovat v query parametrech pro přímočarou podporu navigace v historii prohlížeče. Karta receptu bude obsahovat název, popis, obrázek, hodnocení, počet recenzí, čas přípravy a počet instrukcí. Navíc bude možné rozbalit seznam ingrediencí, ve kterém budou zvýrazněny aktuálně vyhledávané ingredience. Uživatel bude přesměrován na obrazovku s detailem receptu při stisknutí tlačítka `View` nebo při kliknutí na obrázek receptu. Konkrétní rozložení obrazovek viz obrázek 1.1 pro desktopová zařízení a 1.2 pro mobilní zařízení.

Detail receptu

Na obrazovce s konkrétním receptem bude obsažen název, popis, obrázek, autor, datum publikování, hodnocení s počtem recenzí, nutriční hodnoty a samozřejmě ingredience a postup přípravy. Rozložení pro větší obrazovky viz obrázek 1.3, pro menší obrazovky se všechny karty zobrazí v 1 sloupci analogicky k rozložení 1.2. Jako budoucí rozšíření by bylo možné implementovat modul recenzí. V první fázi by recenze byly pouze extrahovány ze zdrojových datasetů, v další fázi by uživatelé mohli nové recenze přidávat prostřednictvím naší aplikace.

Detail ingredience

Na obrazovce detailu ingredience budou prezentována data z otevřených znalostních grafů. Zaměříme se primárně na jméno, popis a obrázek ingredience, které dle dostupných informací doplníme o nutriční hodnoty, kategorie, místo původu a další zajímavosti.

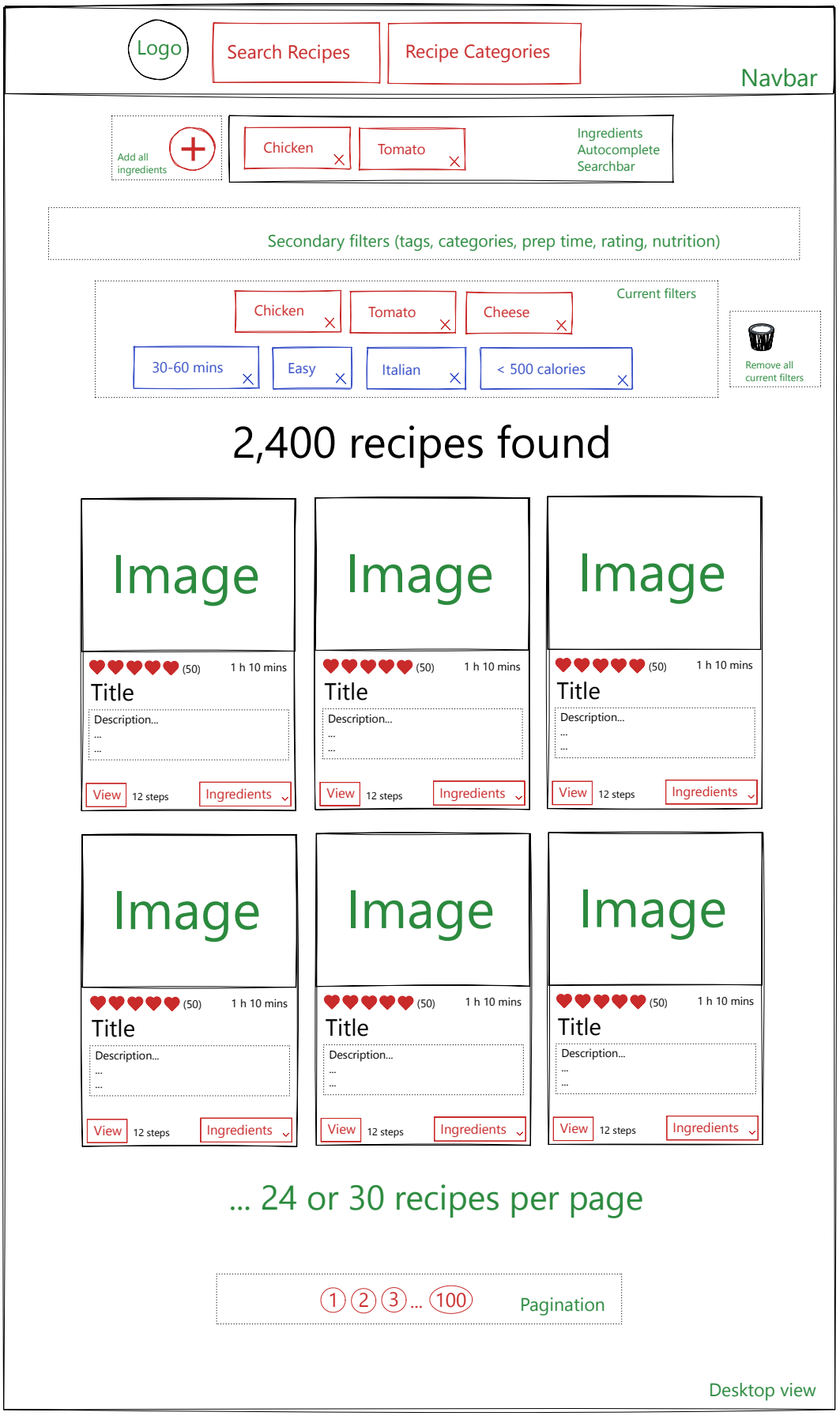
1.2 Dostupné datové sady

V této sekci je vyhrazen prostor pro analýzu různých veřejně dostupných datasetů z domény receptů. Nejedná se ani zdaleka o kompletní výčet, měly by ale být představeny nejznámější alternativy, které by mohly být vybrány jako podklad pro obsah aplikace.

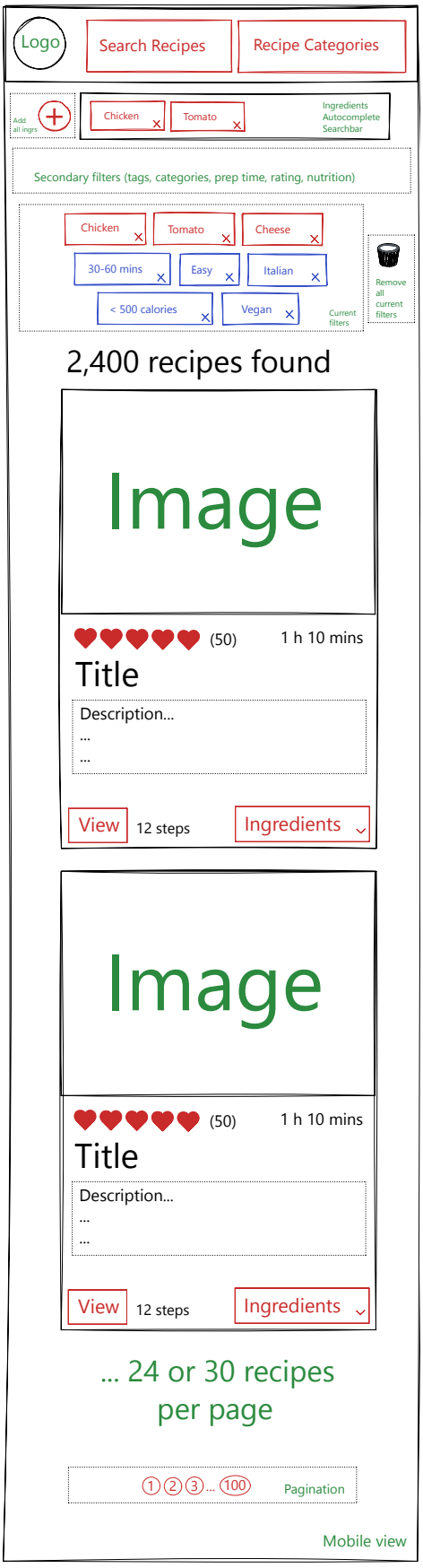
1.2.1 Recipe1M+

Jedním z nejdůležitějších projektů v této oblasti je *Recipe1M+*, strukturovaný korpus obsahující přes 1 milion receptů a 13 milionů souvisejících obrázků jídla. Aktuálně se jedná o největší veřejně dostupnou sadu receptů. Dataset je dostupný pouze přihlášeným uživatelům z ověřené organizace a je povoleno jej využívat výhradně pro účely studia a výzkumu. Pro registraci lze využít univerzitní email. Z celkového počtu 1 milionu receptů obsahuje 50 000 receptů s nutričními informacemi [6]. V naší aplikaci preferujeme nutriční hodnoty zahrnout, pokud jsou dostupné na zdrojové stránce receptu. Měli bychom tedy k dispozici 50 000 dokumentů s touto informací. Ostatní data jsou určena přednostně pro strojové zpracování prostřednictvím trénování modelů.

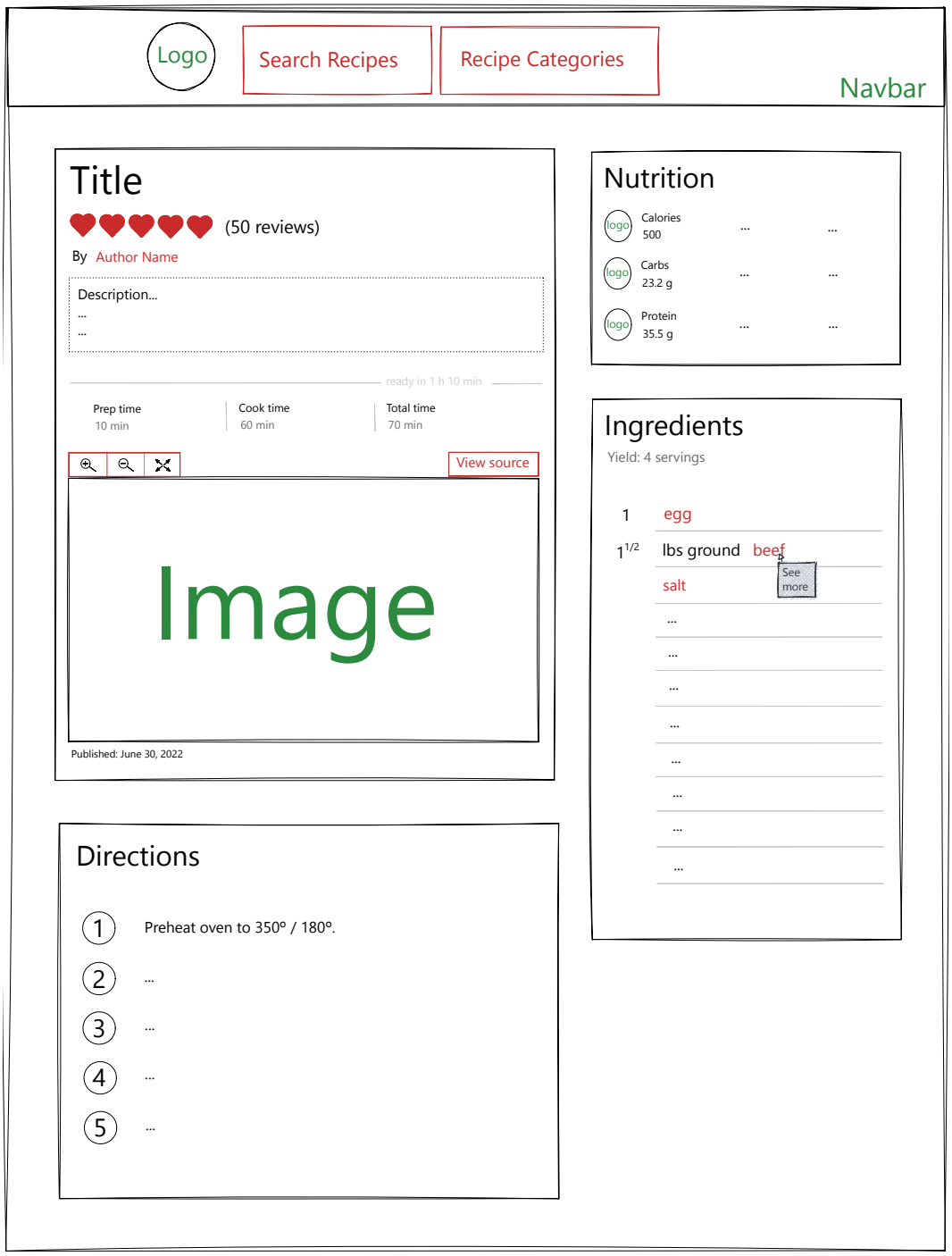
Celková velikost datové sady se pohybuje v řádu stovek gigabytů, samotné JSON dokumenty se strukturovanými recepty z adresáře `layers` se ale vejdou do 2 GiB, tudíž by byly vhodné pro potřeby této práce limitované omezenou výpočetní kapacitou. Lze odtud využít 1 029 720 receptů obsahujících název, url, ingredience a postup přípravy. Odkazy na ilustrační fotografie jsou u 402 760 z těchto receptů. Pro příjemnější uživatelský zážitek se omezujeme pouze na recepty s obrázky, takže jsme z datasetu Recipe1M+ schopni použít přibližně 400 000 receptů, pokud akceptujeme absenci nutričních hodnot. Bylo by spíše obtížnější z tohoto datasetu identifikovat názvy ingrediencí, neboť jsou suroviny uloženy včetně jejich množství a jednotek měření v rozmanitém formátu.



Obrázek 1.1: Obrazovka vyhledávání receptů pro desktopová zařízení



Obrázek 1.2: Obrazovka vyhledávání receptů pro mobilní zařízení



Obrázek 1.3: Obrazovka detailu receptu

1.2.2 Open Recipes

Dalším významným aktérem na poli volně dostupných receptů je iniciativa *Open Recipes*. Autoři Finkler, Shiflett a Birkebæk projekt představují jako otevřenou databázi záložek s recepty. Pojem záložky je použit z důvodu absence instrukcí k přípravě receptu. Dataset má sloužit pouze k vyhledání receptu a pro detailní informace má být uživatel přesměrován na zdroj s kompletním receptem [7]. Tohoto přístupu úspěšně využívají některé z vyhledávačů receptů, např. populární aplikace *SuperCook*. Naše aplikace si ale klade za cíl zpracovat i stránky s detaily receptů, ze kterých lze dále pokračovat na detaily ingrediencí s informacemi ze znalostních grafů. Projekt Open Recipes tedy pro náš scénář nebude vhodnou volbou.

1.2.3 FoodKG

Přímo v oblasti znalostních grafů figuruje projekt *FoodKG*, který je postaven nad sadou receptů z již zmíněného datasetu Recipe1M+. Recepty doplňuje o podrobnější data k ingrediencím ze stránky The Cook's Thesaurus a definuje vlastní ontologii. Model ontologie je navržen pro zodpovídání dotazů na recepty dle ingrediencí s přihlédnutím k individuálním potřebám uživatele, jako jsou alergie a intolerance na určité složky potravin.

Vývojáři projektu FoodKG zpřístupňují skripty k extrakci dat z encyklopedie The Cook's Thesaurus a k vytvoření znalostního grafu. Neposkytují ale žádné nové recepty nad rámec datové sady Recipe1M+, naší horní hranicí by tedy bylo 50 000 receptů s nutričními hodnotami (viz sekce *Recipe1M+*). Ontologie publikovaná na webových stránkách projektu obsahuje 75 entit ingrediencí, které kromě obecného popisu poskytují informace o glykemickém indexu, obsahu lepku a možných náhradách dané ingredience. Výhodou je připravený RDF formát, nad kterým se lze snadno dotazovat pomocí jazyka SPARQL. Autoři Chen a kol. uvádějí ukázky dotazů, vyberme například dotaz vracející recepty, které obsahují banán a zároveň neobsahují vlašské ořechy [3]:

```
@PREFIX food: <http://purl.org/heals/food/>
@PREFIX ingredient: <http://purl.org/heals/ingredient/>
SELECT DISTINCT ?recipe
WHERE {
    ?recipe food:hasIngredient ingredient:Banana .
    FILTER NOT EXISTS {
        ?recipe food:hasIngredient ingredient:Walnut .
    }
}
```

1.2.4 Food.com Recipes and Interactions

Rozsáhlý dataset *Food.com Recipes and Interactions* s téměř 200 000 recepty extrahovanými z webové stránky Food.com (původního GeniusKitchen) je publikován na portálu *Kaggle*, který shromažďuje podklady pro strojové učení. Datová sada pokrývá 18 let interakce uživatelů včetně hodnocení, počtu recenzí i konkrétních reakcí [8]. Kromě základních informací obsahuje také nutriční hodnoty

receptů, datum publikování a rovněž normalizovaná jména ingrediencí. Ta byla získána parsováním originálního textu surovin, kvůli čemuž nejsou vždy zcela spolehlivě přesná (např. ve jménech často zůstala jednotka měření z původního textu). Unikátních ingrediencí je k dispozici kolem 8 000, což by měl být dostačující základ pro hledání linků s entitami otevřených znalostních grafů. Zároveň ve srovnání s předchozími projekty nabízí nejbohatší informace k jednotlivým receptům.

Nevýhodou datasetu je jeho primární určení pro strojové zpracování. Byl vytvořen jako podklad pro generování personalizovaných receptů na základě dřívějších preferencí uživatele [9]. Syrová data nejsou zamýšlena pro přímou prezentaci, což se negativně odráží na jejich přesnosti a estetice. Slova jsou občas zařazena do špatných kategorií a problematický je zejména plně *lowercase* formát textu, ze kterého nejsme schopni zpětně zrekonstruovat originální text receptu. Dataset bychom tedy nemohli použít samostatně, ale pouze v kombinaci s vlastní extrakcí dat, která by respektovala velikost písma a lépe se vypořádala s parsováním jednotlivých kategorií.

Tento problém je poměrně snadno řešitelný díky struktuře stránky Food.com. Z id receptu lze jednoduše složit url ve formátu `www.food.com/recipe/id` a navíc aplikace podporuje koncept propojených dat, tedy poskytuje recepty ve strukturovaném RDF formátu. Do HTML hlaviček všech dokumentů s recepty vkládá JSON-LD serializaci dle ontologie *Schema.org*. Z připraveného datasetu bychom tedy mohli využít identifikátory receptů a normalizované ingredience, pro každý recept extrahovat jeho JSON-LD a spojit informace dohromady. Zároveň bychom si ušetřili práci s převáděním receptů do JSON-LD formátu a připravené soubory rovnou vložili do hlaviček dokumentů. Nevytvářeli bychom nové entity receptů, pouze bychom změnili prezentační vrstvu RDF dat. Identifikátory entit v podobě IRI by tedy zůstaly nezměněné.

1.2.5 Generování vlastního datasetu

Pokud se nespokojíme s žádnou z dostupných datových sad, případně potřebujeme data rozšířit a posbírat je přímo ze zdroje, využijeme metodu zvanou *web scraping*. V rámci tohoto procesu musíme analyzovat cílovou stránku z pohledu získávání a prezentace dat. S využitím vývojářským nástrojů ve webovém prohlížeči můžeme přes panel `Network` sledovat požadavky, které aplikace odesílá na svůj server a v mnoha případech se na toto interní API dokážeme napojit a získat data ve strukturované podobě. Aplikace typicky pracují s REST API, GraphQL API nebo jejich kombinací a standardně data poskytují ve formátu JSON. Pokud žádný fetch request pro získávání potřebných dat neobjevíme, musíme informace extrahovat přímo z HTML dokumentu prostřednictvím CSS selektorů. V obou případech budeme aplikaci posílat GET requesty, ať už na její backend pro strukturovaná data nebo na frontend pro HTML dokumenty k následnému parsování.

Problematická je kategorie aplikací, které data nezískávají s využitím transparentních fetch requestů a zároveň potřebují spouštět JavaScript kód pro vygenerování obsahu. Zde nestačí pouhé poslání GET requestu přes HTTP, neboť odpověď neobsahuje žádná relevantní data uvnitř HTML. Pro zvládnutí tohoto typu stránek potřebujeme zapojit automatizaci webového prohlížeče. Nejznámějšími pro-

jekty, které se této automatizaci věnují, jsou Selenium[1], Puppeteer[2], Playwright[3] a Cypress[4] pro automatizaci testování [10]. Všechny ze zmíněných projektů jsou open-source.

Během posílání requestů můžeme rovněž narazit na různé formy blokování, od limitu maximálního počtu requestů z jedné IP adresy přes povinné autorizační tokeny až po captcha testy řešitelné pouze s využitím umělé inteligence. Některé aplikace navíc kontrolují tzv. otisk webového prohlížeče. Jedná se o sadu informací k zařízení uživatele, jmenovitě data o konkrétním hardwaru, operačním systému a webovém prohlížeči včetně konfigurace [11]. Také se při neopatrnosti může stát, že server aplikace zahltíme příliš velkým množstvím paralelních requestů, čímž prodloužíme dobu odezvy nebo zpracování dalších requestů dočasně zcela znemožníme.

Stejně jako v jiných oblastech se hodí využít nástroj, který co nejvíce běžných problémů vyřeší za nás. Na poli open-source nástrojů pro extrakci dat si vedoucí pozici drží knihovna Scrapy[5] psaná v jazyce Python, která nabízí celou řadu pokročilých funkcí proti blokování requestů. Pro potřeby této práce by ale vzhledem k rozsáhlejší osobní zkušenosti byla vhodnou volbou knihovna Apify[6] pro Node.js. V arzenálu má zpracování HTTP requestů s následným parsováním HTML pomocí knihovny Cheerio[7], ale také automatizaci webového prohlížeče s využitím knihoven Puppeteer nebo Playwright, včetně generování otisků webového prohlížeče. Navíc zajišťuje rotaci IP adres, čímž snižuje množství zablokovaných requestů. IP adresy lze v rámci placeného účtu získat přímo od firmy Apify, nebo na vstupu poskytnout seznam vlastních. Obecně preferujeme program nespouštět z osobní IP adresy, neboť riskujeme, že nás stránka někdy i natrvalo zablokuje, případně se naše IP adresa dostane na veřejný seznam adres doporučených k blokování.

S dostatkem času, výpočetních prostředků, IP adres pro rotování a s velikou kapacitou úložiště bychom byli schopni zpracovat většinu vybraných aplikací s recepty. Pro každou stránku bychom napsali dedikovaný program a postupně extrahovali data z celé stránky. Recepty z různých aplikací bychom uložili ve sjednoceném formátu a výsledkem by byl kvalitní dataset s maximálním množstvím dat, které lze od zdrojových stránek získat. Práce ovšem necílí na datovou sadu takovéto velikosti. Místo toho se zaměřuje na vytvoření infrastruktury nad podmnožinou receptů, kterou bude možné libovolně škálovat dle možností dalšího vývoje. V případě vlastní extrakce dat bychom si tedy vybrali dva až tři zástupce aplikací, navrhli pro ně jednoduché řešení extrakce dat a omezili počet sesbíraných výsledků na rozumnou hodnotu. Dle požadavků aplikace musíme zároveň splnit spodní limit více než 50 000 receptů. Vhodným kandidátem by jednoznačně byla zmíněná stránka Food.com, která v době psaní této práce obsahuje přes 500 000 receptů a pro cca 200 000 z nich máme k dispozici unikátní identifikátory skrze dataset z platformy Kaggle. Navíc dokumenty s recepty obsahují JSON-LD reprezentaci v hlavičce HTML. Pro každý recept se známým id by tedy

[1] https://github.com/SeleniumHQ/selenium
[2] https://github.com/puppeteer/puppeteer
[3] https://github.com/microsoft/playwright
[4] https://github.com/cypress-io/cypress
[5] https://github.com/scrapy/scrapy
[6] https://github.com/apify/apify-js
[7] https://github.com/cheeriojs/cheerio

stačilo vytvořit url, poslat na něj GET request a z HTML odpovědi extrahovat JSON-LD data. Podobně bychom mohli zpracovat recepty ze stránky Allrecipes, kde jsou v detailech receptů rovněž publikována JSON-LD data. Url receptů by mohl objevit přímo náš program během procházení stránky nebo bychom mohli využít nasbírané url adresy z datasetu Recipe1M+.

1.2.6 DBpedia

Znalostní graf DBpedia by se nám hodil pro extrakci rozšiřujících informací k ingrediencím nasbíraným z jednotlivých receptů. Prvním krokem by byla identifikace názvů ingrediencí z datasetu s recepty. Ideální by bylo mít k dispozici již extrahované ingredience, což není samozřejmostí, neboť recepty jsou často poskytovány bez strukturovaného textu surovin. Ten kromě názvu ingredience může obsahovat také její množství a jednotku měření. Není pak snadné spolehlivě určit, která část textu není součástí jména ingredience, zejména kvůli rozmanitým názvům jednotek měření. Nicméně i jednotek je jen konečné množství a s dostatečným úsilím bychom je měli být schopni identifikovat. Pro každou novou lokalizaci bychom ale problém řešili znova. Strukturovaným ingrediencím s odděleným názvem, množstvím a jednotkou měření bohužel nenahrává standardizovaný formát receptu dle ontologie Schema.org[8]. Ten definuje typ vlastnosti `recipeIngredient` jako prostý text, tedy včetně množství a jednotky.

Jakmile by se nám podařilo získat určitou skupinu jmen surovin, vytvořili bychom jednoduché entity ingrediencí v RDF formátu. Stačilo by každé surovině přiřadit unikátní IRI a vlastnost typu `rdfs:label` odpovídající názvu dané ingredience. Z těchto informací bychom sestavili RDF dataset a nahráli jej do aplikace Silk Workbench. Následně bychom provedli konfiguraci DBpedia SPARQL endpointu a nalezli shody s hodnotami `rdfs:label` ve znalostním grafu DBpedia. Abychom se vyhnuli prohledávání celého grafu, potřebujeme nastavit omezení na povolený typ entit. Po zběžné analýze konkrétních instancí surovin na DBpedia můžeme využít např. následující jednoduché omezení, kde proměnná `?a` znázorňuje dle konvence Silk Workbench hledanou entitu:

```
{
    {
        ?recipe <http://dbpedia.org/ontology/ingredient> ?a
    } UNION {
        ?a <http://dbpedia.org/ontology/ingredient> ?anotherIngredient
    }
}
```

Výše uvedený fragment dotazu cílí na všechny entity, které vystupují jako ingredience jiných entit a zároveň z opačného směru hledá všechny entity, které obsahují nějakou ingredienci. Z dat na DBpedia totiž můžeme vypozorovat, že obsahuje nejen základní ingredience, ale také suroviny složené z více přísad. Takové ingredience by bylo možné považovat za recept, nicméně i ony mohou být dále použity v rámci komplexnějšího postupu přípravy pokrmu. Dobrým příkladem složené ingredience je guacamole, které bývá často uváděno jako přísada

[8] https://schema.org/Recipe

(např. u hamburgerů), ale samo je produktem z avokáda, rajčat, cibule, česneku a limetky.

Dále si potřebujeme zvolit podmnožinu informací, které chceme extrahovat a uložit do vlastní databáze. Opět provedeme pozorování konkrétních instancí surovin a identifikujeme názvy vlastností pro jméno, popis, obrázek, nutriční hodnoty, kategorie a místo původu. Je třeba mít na paměti, že množství dat pro jednotlivé ingredience bude velmi proměnlivé a i jména vlastností nebo typ hodnot mohou být do určité míry odlišné. Do budoucna je zde prostor pro získání kvalitních dat pro rozdílné lokalizace aplikace, neboť ve znalostním grafu lze snadno filtrovat literály v požadovaném jazyce.

1.2.7 Wikidata

Práce se znalostním grafem Wikidata by probíhala velmi podobně jako u výše popsaného projektu DBpedia. Pomocí Silk Workbench bychom vytvořili vazby mezi entitami ingrediencí a následně extrahovali data k nalezeným přísadám. Ve srovnání s obsahem grafu DBpedia je zde většinou k dispozici menší množství textu a spíše je kladen důraz na odkazy do jiných zdrojů. Dle zběžného pozorování ale graf Wikidata často poskytuje relevantnější obrázky, kategorie a místo původu. Také mnohdy uvádí obsažené složky (např. smetanu a mléko u másla) a u vybraných ingrediencí zobrazuje barvu nebo dokonce Unicode znak.

Pokusíme se získat z DBpedia i Wikidata co nejvíce ze zmíněných informací a sloučit je dohromady. Tím bychom měli vylepšit poměr ingrediencí, ke kterým najdeme větší množství zajímavých informací. Dotaz na stejnou vlastnost dané entity může totiž na DBpedia i Wikidata dopadnout úplně jinak, přestože oba čerpají z projektu Wikipedia.

1.3 Úspěšné webové aplikace s recepty

V této sekci projdeme konkrétní příklady úspěšných webových aplikací z domény vyhledávání receptů a uvedeme, v čem se od nich plánujeme odlišit a které rozšiřující funkce naše aplikace nabídne. Stránek v oblasti gastronomie a přípravy pokrmů existuje velké množství, vybereme tedy podmnožinu těch nejznámějších na základě jejich pozice ve vyhledávaných výsledcích. Pro přibližnou analýzu návštěvnosti využijeme kombinaci dotazu `recipes by ingredients` ve vyhledávači Google, dále položíme stejný dotaz platformě Spyfu[9] a nakonec zkontrolujeme kategorii `Cooking and Recipes` na stránce Similarweb[10]. Vzhledem k anglické lokalizaci naší aplikace zkontrolujeme vyhledávané výsledky na Google přes IP adresu amerického původu a kategorii `Cooking and Recipes` vyhodnotíme rovněž v kontextu Spojených Států Amerických, které mají oproti Velké Británii větší zastoupení z pohledu počtu uživatelů.

Dle výše definované metriky jsou v době psaní této práce na předních příčkách následující aplikace pro vyhledávání receptů dle ingrediencí: Supercook, Allrecipes, MyFridgeFood, Taste of Home a Recipeland. Z obecné domény receptů pak vysoké umístění mají ještě stránky Simply Recipes, Food Network,

[9] https://www.spyfu.com/

[10] https://www.similarweb.com/

The Spruce Eats nebo i Food.com — stránka zmíněná v kapitole o dostupných datových sadách.

1.3.1 Allrecipes

Aplikace Allrecipes nabízí různé kategorie receptů pro rychlou inspiraci a také funkci vyhledávání dle ingrediencí. Po přesměrování na vyhledávací stránku zobrazí seznam všech receptů, který se aktuálně pohybuje kolem 55 000. V naší aplikaci bychom dokázali nabídnout více výsledků, navíc z různých zdrojů včetně Allrecipes. Vyhledávací filtry umožňují nastavit požadované ingredience, klíčová slova či název receptu, ale také ingredience, které hledaný recept nesmí obsahovat. Je zde možnost odstranit všechny filtry najednou, kterou určitě plánujeme poskytnout i v naší aplikaci. Dle funkčních požadavků uživateli předložíme širší možnosti filtrování včetně času přípravy, hodnocení a nutričních hodnot. Navíc budeme podporovat fasetové vyhledávání, které se bude průběžně aktualizovat a uživatel vždy předem uvidí, kolik receptů se mu zobrazí při výběru daného filtru. Vyhledávání v aplikaci Allrecipes navíc postrádá našeptávač, uživatel tedy nedostává žádnou zpětnou vazbu před explicitním stisknutím tlačítka `Show results`. V naší aplikaci budeme výsledky aktualizovat průběžně po přidání každého nového filtru. Také není podporováno řazení výsledků dle hodnocení nebo počtu recenzí, takže není jasné, jaká kritéria určují pořadí receptů. Načítání nových receptů je řešeno nekonečným posouváním stránky, což při prohlížení působí plynule a proces renderování je velmi rychlý. Detail receptu je ale potřeba otevřít v novém okně, jinak při navigaci zpět ztratíme pozici prohlížení.

Detailní stránka receptu je interaktivní, nabízí přizpůsobení množství surovin dle počtu porcí, označování jednotlivých kroků postupu za splněné a také lze přidat ingredience na nákupní seznam. Tato možnost je ale k dispozici pouze přihlášeným uživatelům. V naší aplikaci bychom mohli jako rozšíření implementovat přidávání surovin na nákupní seznam i nepřihlášeným uživatelům. Stav seznamu by byl uložen v prohlížeči, stejně jako tomu bývá u nákupních košíků v online obchodech.

1.3.2 Supercook

Vyhledávání dle ingrediencí je v aplikaci Supercook implementováno rozdílně od většiny ostatních aplikací. Recepty jsou řazeny podle toho, zda je lze připravit pouze ze zadaných surovin. To může být v praxi velmi užitečný přístup, neboť nemusíme nakupovat žádné dodatečné ingredience a vystačíme si s domácími zásobami. Pro větší přehlednost aplikace počítá, že základní ingredience typu sůl nebo voda má k dispozici každý a vůbec je proto nezařazuje do vyhledávání. S každou přidanou ingrediencí se zvětšuje počet dostupných receptů. Pro zjištění celkového počtu receptů bychom museli aplikaci sdělit, že máme k dispozici všechny navrhované ingredience. Např. po přidání všech surovin z kategorie zeleniny se zobrazí téměř 2 000 receptů. Ingredience lze vybírat z přehledných kategorií nebo pomocí vyhledávání s našeptávačem. Kategorie surovin bychom v budoucnu chtěli v naší aplikaci zavést také. Bylo by snadné kategorie extrahovat přímo ze stránky Supercook, chceme se ale vyhnout kopírování vzhledu. Aplikace rovněž nabízí řadu doplňujících filtrů včetně kategorií, typu kuchyně, diety, hodnocení, času přípravy

nebo maximálního počtu ingrediencí. Také lze vyhledávat na základě názvu receptu. Vyřazení dané ingredience z vyhledávání by mohlo být považováno za lehce zmatečné. Nejprve je potřeba ingredienci přidat do vyhledávání, poté se zobrazí v seznamu surovin, které lze vyřadit, a odtamtud ji lze označit jako zakázanou.

Celkově se stránka Supercook snaží budit dojem desktopové aplikace, což se jí poměrně daří díky perzistovanému výběru ingrediencí a neměnné url adrese. Oproti naší aplikaci implementuje pouze vyhledávání receptů a pro zobrazení detailu receptu poskytuje odkaz na zdrojovou stránku.

1.3.3 MyFridgeFood

Aplikace MyFridgeFood řeší vyhledávání ingrediencí pouze pomocí definovaných kategorií s konkrétními surovinami. Ingredience nelze vyhledávat dle jména s přímou podporou aplikace, pouze skrze hledání na stránce v rámci prohlížeče. Přidané ingredience lze nicméně snadno odstraňovat po jedné, nebo všechny najednou. Výsledky je potřeba zobrazit stisknutím tlačítka `Find Recipes`. Aplikace u vyhledaných receptů zobrazuje chybějící ingredience, které lze snadno přidat do vyhledávání. Také ve výsledcích prezentuje kategorie receptů, čas přípravy a nutriční hodnoty. Po přihlášení navíc umožňuje vytváření záložek s recepty. Vyhledané výsledky lze filtrovat na základě fasetových kategorií a počtu kalorií. Zajímavostí je také dialog pro personalizovanou volbu receptu. Lze jej nalézt pod záložkou `Decider`, bere v potaz aktuálně zadané ingredience a pokládá uživateli několik upřesňujících otázek ohledně požadovaného času přípravy, počtu porcí nebo množství kalorií. Na závěr dle odpovědí vybere nejvhodnější recept, případně nabídne výchozí volbu, pokud vzhledem k zadaným omezením žádný neobjevil.

1.3.4 Taste of Home

Webová aplikace Taste of Home poskytuje sjednocené rozhraní pro zadávání filtrů, ať už se jedná o ingredience nebo klíčová slova, typ kuchyně či obtížnost receptu. Zároveň nabízí vybrané kategorie, které lze rozkliknout a dostat se přes ně ke konkrétním filtrům. Při každém přidaném filtru se obnoví seznam nalezených receptů a také se aktualizuje vyhledávání filtrů. Nabízejí se pouze ty filtry, při jejichž výběru se stále zobrazí aspoň 1 recept. Tento přístup rovněž aplikujeme na naší stránce v rámci fasetového vyhledávání, tedy nabídneme jen ty filtry, pro které máme k dispozici 1 a více výsledků. Aplikace Taste of Home ovšem pro každou změnu kategorie nebo filtru načítá obsah celé stránky, kvůli čemuž nepůsobí zcela plynule.

Tvůrci aplikace si zakládají na vysoké kvalitě publikovaných receptů, které procházejí pečlivým výběrem a testováním. Obsah prezentovaný na stránkách jednotlivých receptů je díky tomu precizně zpracovaný a rozmanitý, nechybí interaktivní videa ani zajímavé tipy. Přímo z obrazovky detailu lze přejít na následující recept z původního vyhledávání, což je užitečné vylepšení. Dokonce je implementováno rozšíření, které jsme zmiňovali v úvodní kapitole, totiž mapování ingrediencí na odpovídající entity v online supermarketech. Ingredience mohou být ze stránky s detailem receptu po zadání ZIP kódu automaticky přidány do košíku dostupného online supermarketu.

1.3.5 Food.com

Na závěr uveďme aplikaci Food.com, k níž máme k dispozici dataset téměř 200 000 receptů a již víme, že k publikovaným dokumentům poskytuje JSON-LD reprezentaci. Na rozdíl od ostatních aplikací z této sekce nenabízí funkci vyhledávání dle ingrediencí, ale pouze dle názvu receptu. O to více by se hodilo použít recepty z Food.com a obohatit je o vyhledávání dle surovin, stejně jako to dělá zmíněný agregátor receptů Supercook. Naopak zde jako u první aplikace nacházíme rozšiřující informace k ingrediencím. Ty jsou dostupné ze stránky s detailem receptu. Obrazovka detailu ingredience obsahuje různé množství informací v závislosti na důležitosti suroviny. Za povšimnutí stojí podrobné nutriční informace, náhrady jinými ingrediencemi nebo kombinace s ostatními přísadami. Pod textovými informacemi následuje seznam populárních receptů s aktuální ingrediencí, kterým lze listovat s automatickým načítáním nových výsledků. Tento nápad si dovolíme integrovat ve vlastní aplikaci a zobrazit připravené karty receptů z vyhledávací obrazovky také na stránce ingredience. Z detailu přísady na Food.com je dále možné se prokliknout na seznam všech ingrediencí. Odkaz na tuto stránku[11] je v detailu ingredience možná záměrně skrytý, neboť se nezdá, že by k němu vedla alternativní cesta ze společného menu. Dle metadat v HTML dokumentu obsahuje slovník přes 900 ingrediencí.

[11] https://www.food.com/about

2. Architektura řešení

V této kapitole se budeme věnovat návrhu architektury od extrakce a uložení dat přes přípravu vyhledávacích indexů až po prezentaci těchto dat v rámci funkční aplikace. Předzpracování dat bude klíčovou fází řešení spolu s prezentační vrstvou na straně klienta. Serverová část aplikace bude vystupovat pouze jako prostředník mezi frontendem a databází, respektive platformou Solr pro vyhledávací dotazy. Jejím úkolem bude zprostředkování požadovaných dat z perzistovaného úložiště, která téměř beze změny předá klientovi. Tento přístup si vybere daň v podobě obsáhlejší databáze, neboť budeme ukládat i taková data, která bychom dokázali vygenerovat z ostatních informací. Nejdůležitější instancí tohoto opakování dat bude uložení JSON-LD reprezentace a zároveň strukturovaných dat v rámci každého dokumentu receptu. Strukturovaná data by bylo možné při každém dotazu odvodit z JSON-LD nebo naopak. Přesunuli bychom ale komplexitu převodu z jednorázové fáze předzpracování na samotnou aplikaci, ať už na straně serveru či klienta. Příprava dat pro prezentaci by navíc při každém dotazu trvala o něco déle, což by se při větším množství dat mohlo negativně odrazit na svižnosti aplikace a tranzitivně na uživatelském zážitku.

2.1 Příprava dat

V předchozí kapitole jsme si představili řadu alternativ pro získání datasetů s recepty a následně doplňujících informací k ingrediencím. Dle požadavků aplikace potřebujeme minimálně 50 000 receptů z aspoň 2 zdrojů. Také vyžadujeme integraci 2 nebo více znalostních grafů s otevřenými daty. Nemáme spodní limit na počet ingrediencí, které musíme ze znalostních grafů extrahovat. Záleží totiž na úspěšnosti propojení našich ingrediencí s entitami ze znalostních grafů, která se projeví až při praktickém testu.

Přípravu dat můžeme dále rozdělit na 2 základní fáze a to extrakci a čištění dat. Ne všechna data jsou totiž vhodná pro přímou prezentaci uživateli. Jak jsme zmiňovali v minulé kapitole, volně dostupné datasety s recepty často cílí spíše na oblast strojového učení. Extrahovaná data je tedy potřeba manuálně zkontrolovat a navrhnout heuristiku, pomocí které bude většina dat normalizována, případně odstraněna při nesplnění zadaných kritérií. Vzhledem k charakteru problému a omezené časové dotaci nelze na každou datovou sadu aplikovat deterministické řešení, které by eliminovalo všechny anomálie, proto se v některých případech musíme spokojit s heuristikou.

2.1.1 Extrakce dat

Na tomto místě je vhodné rozhodnout, které ze zdrojů dat popsaných v předchozí kapitole nakonec použijeme v rámci našeho řešení. U dat k ingrediencím máme na výběr znalostní grafy DBpedia a Wikidata, případně méně obsáhlý RDF dataset z projektu FoodKG. Pro maximalizaci počtu nalezených výsledků se zaměříme na grafy DBpedia a Wikidata. V kategorii receptů zvolíme kombinaci statických datových sad s recepty a generování vlastních datasetů pomocí procesu web scraping. Během implementace vlastní extrakce dat zapojíme knihovnu Apify

pro Node.js a její koncept tzv. *actorů*, což jsou programy určené primárně pro cloudovou platformu Apify, kde jsou spouštěny uvnitř Docker kontejnerů. Mohou mít za úkol automatizaci libovolných úkonů prováděných ve webovém prohlížeči, od jednoduchého posílání e-mailů až po extrakci dat z komplexních webových stránek. Actory lze pomocí Apify CLI spouštět i lokálně, čehož pro jednodušší konfiguraci využijeme v našem řešení. Volitelně lze aktivovat rotování IP adres, které chrání naši vlastní IP adresu před dočasným či dokonce trvalým zablokováním a zlepšuje poměr úspěšných requestů. Vzhledem k obecně nižší míře blokování ze strany aplikací s recepty by využití proxy nemělo být nutné, je ale doporučeno. Počet současně odesílaných requestů omezíme na doporučenou hranici 50 requestů, čímž bychom měli předejít přetížení zpracovávané webové aplikace.

Food.com

Vzhledem k požadavkům definovaným v předchozí kapitole nám bude vyhovovat dataset Food.com Recipes and Interactions dostupný na platformě Kaggle, z něhož jsme schopni získat přibližně 180 000 identifikátorů receptů a také seznam normalizovaných ingrediencí. Dle provedené analýzy není vhodné použít textová data v prezentační vrstvě vzhledem k jejich lowercase formátu. Navrhneme tedy řešení z oblasti web scrapingu, které na vstupu přijme url adresy s detaily receptů, pošle na každé ze zadaných url GET request a z HTML odpovědi extrahuje JSON-LD data. Program bude mít možnost získat přes CSS selektory libovolná data z načteného HTML, pokud by v JSON-LD reprezentaci nebyla obsažena, nebo byla méně strukturována. Programu tedy přidělíme také zodpovědnost za tvorbu strukturovaných dat, která do vygenerovaného datasetu uloží ke každému receptu spolu s jeho JSON-LD podobou. Strukturovanými daty zde rozumíme čas přípravy, počet porcí, klíčová slova, která jsou v JSON-LD uložena ve společném řetězci namísto pole řetězců, hodnocení receptu s počtem recenzí, nutriční hodnoty s jednotkami měření a ingredience s množstvím (případně i jednotkou) odděleným od ostatního textu.

Extrahované výsledky uložíme do společného JSON souboru, který následně sloučíme s vybranými informacemi z datasetu Food.com Recipes and Interactions. JSON-LD např. neobsahuje kompletní informace o autorovi, ale pouze jeho jméno. Dle samotného jména nejsme schopni autora jednoznačně identifikovat a zjistit odkaz na jeho profil v rámci aplikace Food.com. Url adresa autora je totiž sestavena z jeho unikátního id, které máme k dispozici právě v datasetu z Kaggle. Dále budeme chtít extrahované recepty rozšířit o normalizované ingredience, abychom nemuseli navrhovat vlastní heuristiku a usnadnili si pozdější mapování ingrediencí na entity ze znalostních grafů. Po sloučení všech potřebných dat provedeme finální čištění a následně recepty jako JSON dokumenty uložíme do databáze.

Výše popsané řešení extrakce dat z Food.com má nevýhodu z pohledu škálovatelnosti. Maximální počet receptů, které jsme schopni získat, je roven počtu receptů v datasetu z Kaggle. Celkový počet receptů na stránce Food.com se od doby pořízení datasetu zvětšil více než dvakrát na aktuálních 526 851 receptů. Nicméně i s naším zjednodušeným programem vyžadujícím připravené url adresy detailů receptů jsme schopni získat téměř kompletní data. Zmíněný dataset Recipe1M+ v době psaní této práce obsahuje téměř 510 000 url adres receptů z aplikace Food.com. Při potřebě většího škálování bychom mohli využít tato url,

neměli bychom k nim ovšem normalizované ingredience a byli bychom omezeni striktně akademickým využitím. Pro účely naší práce se spokojíme s horní hranicí 180 000 receptů s normalizovanými ingrediencemi. Tyto recepty jsou dle autorů datasetu Majumdera a kol. podmnožinou receptů z let 2000-2018, které mají aspoň 3 kroky postupu a počet ingrediencí v rozmezí 4 a 20 [9]. Kód souvisejícího projektu pro generování personalizovaných receptů je dostupný jako open-source na platformě GitHub, lze tedy předpokládat, že datovou sadu lze využívat bez omezení.

Allrecipes

Jako další zdroj receptů si vybereme webovou aplikaci Allrecipes. Pro ni sice nemáme k dispozici podrobný dataset jako u stránky Food.com, vystačíme si ale s vlastní extrakcí dat prostřednictvím Apify actora. Mohli bychom využít prakticky stejnou šablonu, jako u programu pro zpracování Food.com. S využitím datasetu Recipe1M+ dokážeme získat 49 000 url adres detailů receptů. Poměrně snadno bychom ale dokázali navrhnout komplexnější řešení extrakce dat, které by dynamicky procházelo celou webovou stránku Allrecipes, našlo detaily všech receptů a z nich extrahovalo aktuální data. Tímto přístupem bychom odstranili závislost na datové sadě Recipe1M+ a získali větší počet výsledků. Pro nalezení všech receptů bychom sice museli zpracovat více požadavků, aplikace Allrecipes ale využívá interní API, přes které lze získat url adresy 48 receptů v rámci 1 requestu. Celkový počet receptů na Allrecipes se aktuálně pohybuje kolem 50 000, což lze zjistit spuštěním vyhledávání bez jakýchkoli nastavených filtrů.

Aplikace Allrecipes nabízí svým uživatelům vyhledávání dle ingrediencí a také možnost přizpůsobit množství ingrediencí dle požadovaného počtu porcí. Tato skutečnost naznačuje, že si aplikace interně spravuje ingredience ve strukturované podobě, přestože v přiloženém JSON-LD je poskytuje jako prostý text včetně množství a jednotky měření. Zaměřme se na konkrétní ingredienci uvnitř HTML dokumentu vybraného receptu. Můžeme si povšimnout, že jsou v atributech příslušného `input` elementu uložena strukturovaná data ingredience v následujícím formátu (ukázka z receptu `92462` pro surovinu kuřecí vývar):

```
<input
    class="checkbox-list-input"
    data-tracking-label="ingredient clicked"
    data-quantity="½"
    data-init-quantity="0.5"
    data-unit="cup"
    data-ingredient="chicken broth"
    data-unit_family="volumetric"
    data-store_location="Soup"
    type="checkbox"
    value="(14.5 ounce) can chicken broth"
    id="recipe-ingredients-label-92462-0-4">
```

Tyto užitečné informace v rámci extraktoru zacílíme pomocí CSS selektorů. Díky tomu získáme výrazně přesnější data, než prostřednictvím normalizovaných ingrediencí z datasetu Food.com Recipes and Interactions.

DBpedia

V první fázi extrakce dat z grafu DBpedia potřebujeme identifikovat entity ingrediencí, které dokážeme namapovat na jména surovin z jednotlivých receptů. K tomu využijeme nástroj Silk Workbench a vytvoříme RDF tvrzení s IRI adresami ingrediencí spojenými vztahem `owl:sameAs`. V rámci úlohy linkování navrhneme tranformaci textu ingrediencí, která dokáže názvy propojit i s mírnými odlišnostmi ve formátu, čísle nebo pádu slov. Pro každou ingredienci vyjádřenou pomocí DBpedia IRI pak extrahujeme vybrané informace včetně názvu, popisu, obrázku, kategorií a místa původu. Z nutričních hodnot se zaměříme na energii v kaloriích nebo kilojoulech, dále na obsah tuku, sacharidů, bílkovin, vlákniny, cholesterolu a cukru. Aktuálně se zabýváme pouze anglickou lokalizací aplikace, všechna textová data tedy omezíme na anglické výsledky. Jedinou povinnou informací bude název (label) ingredience, všechna ostatní data budou nepovinná, neboť se formát i množství dat napříč ingrediencemi výrazně liší.

Teoreticky bychom mohli vytvořit jeden společný SPARQL dotaz pro všechny ingredience a ten odeslat na DBpedia SPARQL endpoint. Dotaz by ale v závislosti na počtu nalezených linků mezi surovinami mohl skončit příliš dlouhý a nenechal by prostor pro škálování. Zvolíme tedy alternativní řešení — dynamicky vytvoříme sadu dotazů stejného formátu, každý s přibližně 20 IRI adresami entit ingrediencí. Tyto dotazy zpracujeme postupně a výsledky uložíme do společného JSON datasetu s detaily ingrediencí. Výsledky si navíc od SPARQL endpointu můžeme vyžádat v řadě různých formátů. Pro naše účely bude nejpraktičtější formát JSON-LD, jehož obsah využijeme v hlavičkách HTML dokumentů ingrediencí. Se SPARQL endpointem lze komunikovat přes grafické rozhraní ve webovém prohlížeči nebo prostřednictvím HTTP GET requestů. Pro snadnější automatizaci procesu extrakce využijeme druhou možnost, kde obsah dotazu předáme na místě query parametru s názvem `query`.

Wikidata

IRI adresy požadovaných entit z Wikidata získáme opět pomocí aplikace Silk Workbench. Také samotný proces extrakce dat bude probíhat analogicky k postupu pro data z DBpedia. I zde využijeme HTTP GET requesty na SPARQL endpoint, kde prostřednictvím query parametrů předáme obsah dotazu a požadovaný formát výsledku. Projekt Wikidata neposkytuje reprezentaci JSON-LD, vystačíme si ale s běžným JSON formátem, který lze vyžádat přes hodnotu query parametru `format` nastavenou na `json`. Z této reprezentace pak sami vytvoříme odpovídající JSON-LD formát, který je vhodný pro strukturovaná data v hlavičce HTML dokumentu.

2.1.2 Čištění dat

V rámci fáze čištění dat potřebujeme extrahovaná data převést do formátu vhodného k prezentaci koncovému uživateli. Jednotlivé kroky procesu čištění mohou být rozloženy do více míst přípravy dat. Již během extrakce dat probíhá odstranění mezer a znaků nového řádku na okrajích řetězců. Dále je potřeba se vypořádat se znaky, které jsou kvůli vnoření v HTML dokumentu kódovány jinými znaky, aby bylo zajištěno jejich korektní zobrazení. Takové znaky se vy-

skytují např. v extrahovaných JSON-LD dokumentech, před jejich uložením do databáze tedy provedeme dekódování. Rekurzivně projdeme obsah každého objektu načteného z JSON-LD dokumentu a všechny řetězce dekódujeme s využitím open-source knihoven pro Node.js. V našem řešení integrujeme knihovny `html-escaper` a `html-entities` dostupné přes správce balíčků npm.

Dále jsme se rozhodli z vyhledávání vyřadit recepty bez fotografie, které lze identifikovat a přeskočit již během fáze extrakce nebo následně při ukládání do databáze, případně až při tvorbě dokumentů pro vyhledávací platformu Solr. Zvolíme poslední způsob, recepty tedy uložíme do vlastní databáze bez ohledu na přítomnost jejich obrázků. Díky tomu budeme mít v budoucnu snadnou cestu k využití zbývajících receptů bez fotografií, ať už pro účely strojového učení nebo i zobrazení uživateli, pokud by větší nabídka receptů výrazně převážila nevýhodu absence ilustračních fotografií.

Také data k ingrediencím budou vyžadovat významné čištění. V datasetu Food.com Recipes and Interactions máme k dispozici přibližně 8 000 unikátních ingrediencí. Co nejvíce z nich bychom chtěli nabídnout uživateli v rámci našeptávače ve vyhledávání dle ingrediencí. Pro tento účel názvy ingrediencí převedeme do estetičtějšího formátu s velkým počátečním písmenem. Po manuální kontrole seznamu ingrediencí ale narazíme na řadu slov, která se mezi suroviny dostala omylem vlivem chybného parsování jmen ingrediencí. Nebudeme zde uvádět kompletní výčet, typicky se ale jedná o názvy jednotek měření nebo obecné fragmenty ingrediencí, které samy o sobě žádnou ingredienci nepředstavují (např. samostatná slova `clove`, `seed`, `extract`, která by byla validní pouze v kontextu typu `garlic clove`, `sesame seed` a `vanilla extract`). Vzhledem k velkému počtu ingrediencí navrhneme heuristiku čištění pomocí regulárních výrazů. Zaměříme se zejména na nejčastěji používané ingredience, které budou zobrazeny v horní části našeptávače. Pro potřeby našeptávače nastavíme limit maximálního počtu slov ingredience a to na hodnotu 3. Pro mapování na entity ze znalostních grafů ale využijeme původní sadu ingrediencí bez omezení počtu slov.

S normalizovanými ingrediencemi z Food.com Recipes and Interactions souvisí další problém — nejsou přiřazeny ke všem receptům z datasetu. Texty surovin sice využijeme z extrahovaného JSON-LD, normalizované ingredience ale potřebujeme k propojení s informacemi z grafů DBpedia a Wikidata. Recepty bez normalizovaných ingrediencí tedy musíme projít a pro každou jejich přísadu zkusit na základě prostého textu nalézt co nejbližší shodu s některou z normalizovaných ingrediencí. Pro zjednodušení budeme akceptovat pouze přesné shody, přestože nám tímto způsobem může část ingrediencí uniknout, neboť mohou být v prostém textu uvedeny v jiném pádě nebo čísle.

Dalším úkolem čisticí fáze bude normalizace JSON-LD reprezentace ingrediencí z grafu DBpedia. Oproti projektu Wikidata zde máme výhodu, jelikož data obdržíme přímo v JSON-LD. Zároveň ale extrahujeme data pro více ingrediencí najednou a každá z nich má vlastní schéma, které se většinou plně neshoduje s ostatními entitami. Při skupinové extrakci dat se ale musí vytvořit univerzální schéma, kterým lze vyjádřit všechny obsažené informace. Naším úkolem bude projít uložené kolekce ingrediencí a pro každou ingredienci vytvořit minimální JSON-LD kontext, kterým ji lze popsat. Jednotlivé ingredience pak do databáze uložíme vždy s vlastním JSON-LD kontextem. V praxi se totiž často stává, že objevíme pod stejnou vlastností různé typy hodnot. Např. region původu ingre-

dience může nést IRI příslušné entity z grafu DBpedia, ale také prostý literál. Skutečný typ musí být řádně definován kontextem JSON-LD dokumentu. Proto není vhodné mít společný kontext pro všechny ingredience, neboť by u některých vlastností existoval duplicitní popis použitých typů.

2.2 Databázový model

Pro uložení dat zvolíme dokumentovou databázi, konkrétně Apache CouchDB.

2.3 Indexy

2.4 Backend

2.5 Frontend

2.6 Aplikační logika

3. Implementace návrhu

3.1 Zpracování vstupních dat

3.2 Databáze Apache CouchDB

3.3 Vyhledávání pomocí Apache Solr

3.4 Rozhraní REST API

3.5 Middleware

3.6 Single-page aplikace

4. Testování

Závěr

Seznam použité literatury

[1] The World Wide Web Consortium. Semantic Web, 2015.

[2] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutiérrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. *Knowledge Graphs*. Number 22 in Synthesis Lectures on Data, Semantics, and Knowledge. Morgan & Claypool, 2021.

[3] Ching-Hua Chen, Deborah L. McGuinness, Nidhi Rastogi, Oshani Seneviratne, Sola Shirai, Ananya Subburathinam, and Mohammed J. Zaki. Foodkg: A semantics-driven knowledge graph for food recommendation.

[4] James David Moody. Categorizing Non-Functional Requirements Using a Hierarchy in UML. mathesis, East Tennessee State University, May 2003.

[5] Garm Lucassen, Fabiano Dalpiaz, Jan Martijn Van der Werf, and Sjaak Brinkkemper. The Use and Effectiveness of User Stories in Practice. pages 205–222, 03 2016.

[6] Javier Marin, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba. Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.

[7] Ed Finkler, Chris Shiflett, and Andreas Birkebæk. Open Recipes.

[8] Shuyang Li. Food.com Recipes and Interactions, 2019.

[9] Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian McAuley. Generating Personalized Recipes from Historical User Preferences. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5976–5982, Hong Kong, China, November 2019. Association for Computational Linguistics.

[10] Boni García, Micael Gallego, Francisco Gortázar, and Mario Organero. A survey of the selenium ecosystem. *Electronics*, 9:1067, 06 2020.

[11] Pierre Laperdrix, Nataliia Bielova, Benoit Baudry, and Gildas Avoine. Browser fingerprinting: A survey. 05 2019.

Seznam obrázků

1.1 Obrazovka vyhledávání receptů pro desktopová zařízení 12
1.2 Obrazovka vyhledávání receptů pro mobilní zařízení 13
1.3 Obrazovka detailu receptu 14

Seznam tabulek

A. Přílohy

A.1 První příloha