Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Návrh a implementace aplikace pro webovou analýzu

Bc. Alžběta Pokorná

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Alžběta Pokorná

Systémové inženýrství a informatika
Informatika

Název práce

Návrh a implementace aplikace pro webovou analýzu

Název anglicky

Design and implementation of application for web analysis

Cíle práce

Cílem práce je analýza, návrh a implementace webové aplikace pro analýzu webových stránek. Aplikace bude sloužit k procházení struktury webů pomocí vytvořeného crawleru, vyhodnocování stránek z hlediska principů SEO a k následné analýze zjištěných dat a návrhů na vylepšení struktury.

Metodika

Práce se sestává z teoretické a praktické části.

Metodika zpracování teoretické části práce spočívá ve studiu odborných informačních zdrojů. Na základě zjištěných poznatků budou formulována teoretická východiska pro zpracování praktické části práce.

Praktická část bude zahrnovat analýzu požadavků, návrh a implementaci webové aplikace s využitím standardních metod a nástrojů softwarového inženýrství. Implementace crawleru a backendové části aplikace bude provedena pomocí programovacího jazyka Python. Dále bude využit vybraný vhodný framework pro implementaci uživatelského rozhranní aplikace.

Výsledná webová aplikace bude otestována na konkrétních webových stránek a budou shrnuty poznatky získané při jejím vývoji a navrženy možnosti budoucího vylepšení.

Doporučený rozsah práce

60-80 stran

Klíčová slova

webová aplikace, webová analýza, Python, SEO, crawler

Doporučené zdroje informací

Beautiful Soup Documentation [online]. Dostupné z: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

BEAZLEY, David M. a Brian K. JONES. Python cookbook. 3rd ed. Sebastopol: O'Reilly, c2013. ISBN 978-1-449-34037-7

Google Search Central documentation [online]. Dostupné z: https://developers.google.com/search/docs

Python documentation [online]. Dostupné z: https://docs.python.org/3/

Předběžný termín obhajoby

2021/22 LS – PEF

Vedoucí práce

Ing. Jiří Brožek, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 1. 11. 2021

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 23. 11. 2021

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 29. 03. 2022

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Návrh a implementace aplikace pro webovou analýzu" jsem vypracovala samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autorka uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 29. 3. 2022 _____

Poděkování

Ráda bych touto cestou poděkovala Ing. Jiřímu Brožkovi, Ph.D. za vstřícnost a ochotu při konzultacích práce. Dále také všem účastníkům uživatelského testování za jejich participaci.

Návrh a implementace aplikace pro webovou analýzu

Abstrakt

Tato diplomová práce se zabývá analýzou, návrhem a implementací webové aplikace určené pro webovou analýzu.

Teoretická část práce se zaměřuje na popis vybraných oblastí problematiky obsahu webů a architektury webových stránek, které jsou dále předmětem analýz prováděných vytvářenou aplikací. Dále jsou v teoretické části popisovány technologie pro vývoj tzv. webových scraperů, skriptů určených pro získávání dat z webů. Rozebírány jsou dále i technologie pro vývoj samotné webové aplikace, především jazyk Python a frameworky Vue.js a Flask.

V praktické části práce je provedena analýza uživatelských požadavků a dostupných aplikací s obdobnou problematikou, návrh samotné aplikace a popis její implementace. V závěru práce je aplikace nasazena, otestována a jsou shrnuty poznatky z jejího vývoje a vytvořeny návrhy pro její další rozšiřování.

Klíčová slova: webová aplikace, webová analýza, Python, SEO, web scraping, crawler, Vue.js, Flask, Beautiful Soup

Design and implementation of application for web analysis

Abstract

This master's thesis deals with the analysis, design and implementation of web application created for web analysis.

The theoretical part contains a description of the areas of website content optimalization and website architecture, which are the subject of analyses performed by the created application. Subsequently, technologies for the development of web scrapers designed to obtain data from websites are described, as well as technologies for the development of the web application itself, especially the Python language and the Vue.js and Flask frameworks.

The practical part of the thesis is an analysis of user requirements, available applications with similar focus, the design of the application itself and a description of its implementation. The application is then deployed, tested, the findings from its development are summarized and suggestions for its further expansion are created.

Keywords: web application, web analysis, Python, SEO, web scraper, crawler, Vue.js, Flask, Beautiful Soup

Obsah

Seznam obrázků

Seznam tabulek

Seznam grafů

Seznam ukázek kódů

Seznam použitých zkratek

SEO	Search Engine Optimization
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
URL	Uniform Resource Locator
HTML	Hypertext Markup Language
SEM	Search Engine Marketing
PPC	Pay Per Click
DoS	Denial of Service
PaaS	Platform as a Service
CDN	Content Delivery Network
WSGI	Web Server Gateway Interface
SMTP	Simple Mail Transfer Protocol
JWT	JSON Web Token

1 Úvod

V dnešní době lidé internet masově využívají jako primární zdroj informací, prostředek pro nakupování, podnikání, a k řadě dalších účelů. Primárním cílem vlastníka webových stránek je většinou především jejich vysoká návštěvnost a dobrá dostupnost ostatním uživatelům internetu. Z tohoto důvodu se musí starat o jejich inzerci, propojenost s ostatními weby a umístění ve webových vyhledávačích. Weby bývají často velmi rozsáhlé a je složité si udržovat přehled o jejich obsahu a celkové funkčnosti, a to především v případě, spravuje-li osoba vyšší počet webových stránek s různým zaměřením.

Tématem této diplomové práce je návrh a vývoj webové aplikace, která majitelům webů umožní analyzovat obsah a architekturu jejich webových stránek. Cílem funkcionalit vyvíjené aplikace je sbírat o webech informace využitelné pro jejich další optimalizaci pro reálné uživatele i webové vyhledávače, detekovat případné nedostatky analyzovaných webů a navrhovat jejich vylepšení a celkově zjednodušit a zpřehlednit majitelům webu správu jejich stránek.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem této diplomové práce je analýza, návrh a implementace webové aplikace sloužící pro analýzu webových stránek. Aplikace bude umožňovat registrovaným uživatelům analyzovat jimi vlastněné weby z hlediska architektury stránek, textového obsahu a plnění principů SEO. V rámci prezentace zjištěných dat z analýz v aplikaci budou uživateli dodávány nápovědy k výstupům a návrhy na možnosti vylepšení.

Aplikace bude nasazena a otestována a budou zhodnoceny poznatky získané během jejího vývoje. Dále budou vytvořeny návrhy na její další vylepšení a rozšiřování.

Dílčími cíli práce je získání teoretických podkladů pro oblasti problematik webových stránek, které budou v rámci aplikace analyzovány. Dále také získání teoretických znalostí pro sestavení samotné aplikace a skriptů určených pro těžbu analyzovaných dat.

2.2 Metodika

Teoretická část práce se bude zabývat popisem principů automatizovaného získávání dat z webů a oblastmi následných analýz prováděných v rámci aplikace, jako je SEO a grafová analýza webů. Dále budou popsány technologie využitelné pro vytváření skriptů pro těžbu dat z webů a vybrané technologie pro tvorbu webových aplikací.

Praktická část práce bude zahrnovat analýzu uživatelských požadavků, analýzu existujících aplikací s obdobnou tématikou, návrh aplikace a samotnou implementaci, která bude popsaná na vybraných částech vytvořené webové aplikace pomocí popsaných ukázek kódů. Výsledná webová aplikace bude nasazena, otestována a budou popsány poznatky získané během jejího vývoje. V praktické části bude využito standardních nástrojů softwarového inženýrství.

Pro tvorbu skriptů pro těžbu dat bude použit programovací jazyk Python a samotná webová aplikace bude vytvořena především s využitím frameworků Vue.js a Flask.

3 Teoretická východiska

3.1 Web scraping

Web scraping je automatizovaný proces shromažďování a zpracovávání rozsáhlého množství dat z webových stránek. Celý tento postup získávání dat se skládá ze dvou hlavních částí. V první se za pomocí algoritmu prochází web, zaznamenává se jeho struktura a získávají se odkazy vedoucí na další webové stránky. Tomuto algoritmu se říká crawler, někdy také pavouk, a funguje na bázi umělé inteligence. Výstupy z crawleru jsou pak předány scraperu, specializovanému softwarovému nástroji, který z nich extrahuje potřebná data, ke kterým se typicky dostává za pomocí HTTP požadavků zasílaných na získané odkazy. Web scraping se využívá například k monitoringu cen a produktů na trhu či získávání e-mailů pro e-mailový marketing. (1)

Webové scrapery mohou být zneužívány k ilegálním aktivitám, jako je porušování práv duševního vlastnictví a ke krádežím cizích obsahů webu. Zejména není povoleno těžit ze stránek citlivá data, jako jsou uživatelská jména a hesla, a obecně používat vytěžená data webů třetích stran k obchodování. Vlastníci webu tak mohou explicitně zakázat scrapování na jejich stránkách v podmínkách užití webu.

Jelikož jsou scrapery automatizovanými nástroji, je možné pomocí nich za velmi krátkou dobu na server odeslat velké množství požadavků. To vede k rychlému získání potřebných dat, ale oproti tomu také velkému zahlcení serveru. Tímto způsobem může dojít k jeho přetížení a shození z provozu, čímž dochází k neúmyslnému DoS útoku. Je tedy potřeba mezi jednotlivými požadavky uměle vytvářet časové prostoje. (2)

3.2 SEO

SEO (Search Engine Optimization neboli optimalizace pro vyhledávače), je možné definovat jako souhrn činností potřebných pro zajištění vysokého počtu zobrazení daného webu na předních příčkách vyhledávacích nástrojů. Hlavními metodami SEO jsou optimalizace obsahu stránek (On-page SEO), a dále zajišťování odkazů na web vedoucích z externích webových stránek. (Off-page SEO). (3)

SEO je podkategorií SEM – Search Engine Marketingu, který zahrnuje marketingové aktivity, které lze provádět prostřednictvím webových vyhledávačů. Optimalizace pro vyhledávače je marketingovou aktivitou, protože jejím primárním cílem

není pouhé umístění stránek na vysokých příčkách ve vyhledávačích, ale především navýšení konverzního poměru. Do SEM mimo SEO spadá dále například reklamní systém PPC (Pay Per Click). (4)

3.2.1 Webové vyhledávače

Webové vyhledávače jsou softwarovou službou a slouží k vyhledávání webových stránek na internetu. Cílem vyhledávačů je vrátit co nejvíce relevantní webové stránky v závislosti na zadaných vyhledávacích frázích. Jednotlivé výstupy jsou pak řazeny dle softwarem vyhodnocené důležitosti nalezených webů. (5)

Vyhledávačů existuje více druhů. Techniky SEO se zaobírají především primárně používanými webovými vyhledávači, které jsou založeny na principu automatického procházení internetu (např. Google, Yahoo!). (3) Automatické procházení internetem a schraňování jednotlivých URL adres webů je zajišťováno pomocí crawlerů. Získaná data se pak ukládají a organizují v databázích.

Mezi další vyhledávací nástroje spadají specializované vyhledávače, které se zaobírají pouze konkrétní problematikou (například akademickými pracemi či hudbou). Do této kategorie lze zařadit například webovou encyklopedii Britannica. Dále existují tzv. Meta vyhledávače, které agregují výsledky z vícero různých jiných vyhledávacích nástrojů (například server dogpile.com). Mezi vyhledávače můžeme řadit také tzv. webové slovníky, které jsou kolekcemi webových odkazů uspořádány neautomatizovaně dedikovanými editory. (např Family friendly sites).

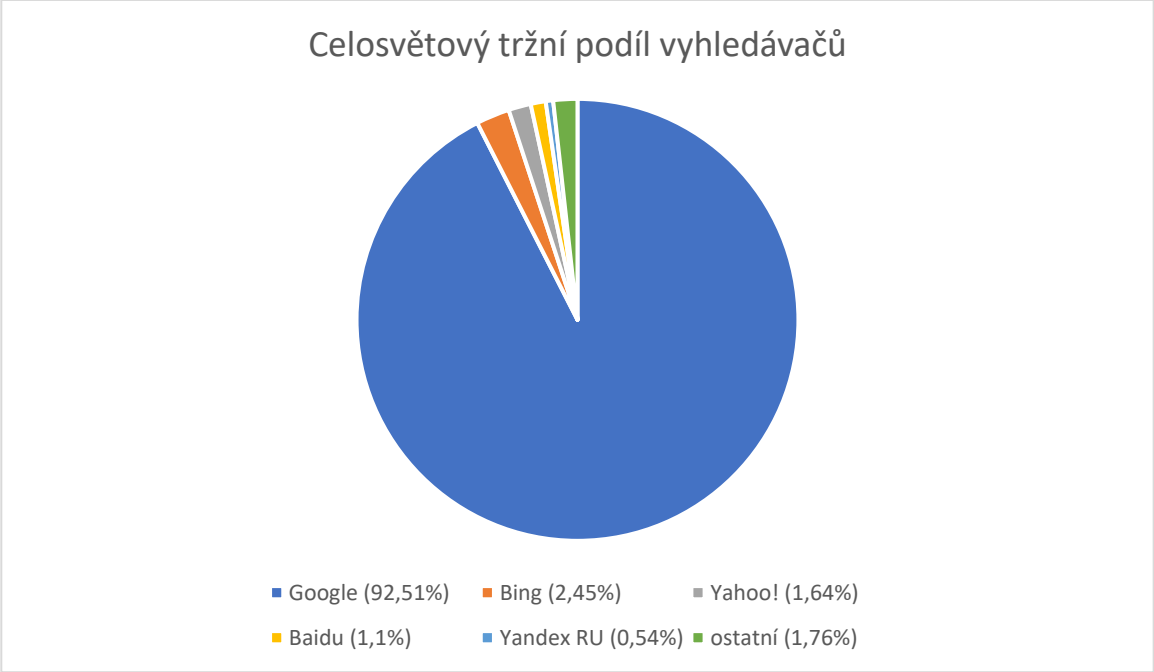
Protože jsou tyto nástroje založeny na různých algoritmech, nelze nikdy dosáhnout optimálního výsledku pro všechny zároveň. Při optimalizaci webových stránek je tedy potřeba zaměřit se pouze na vybrané vyhledávací nástroje. (3)

3.2.1.1 Podíly využitelnosti webových vyhledávačů

Zdaleka nejpoužívanějším webovým vyhledávačem je Google. V roce 2019 byl jeho celosvětový tržní podíl 92,51 %. Na druhém místě se pak nachází vyhledávač Bing a na třetím Yahoo!. Z celosvětového hlediska mají další dva vyhledávače Baidu a Yandex jen nepatrný podíl oproti Google. Baidu je však nejvyužívanějším vyhledávačem v Čínské lidové republice, kde je zakázáno vyhledávač Google používat. Yandex je pak dominantním vyhledávačem v Ruské federaci, kde pouze těsně vítězí nad Googlem. (6)

Graf celosvětového tržního podílu webových vyhledávačů dle dat z roku 2019 je uveden níže. (Graf 1)

Graf 1: Celosvětový tržní podíl vyhledávačů



Zdroj: Vlastní zpracování podle (6)

V České republice jsou nejpoužívanějšími webovými vyhledávači Google a Seznam. Dle statistik z roku 2019 byly dohromady využity v 97 % z celkového počtu vyhledávání. V současné době je již používanějším webovým vyhledávačem Google, a to v poměru 76:24, v roce 2014 byl však podílově nepatrně využívanějším Seznam. (7)

Z důvodu výše uvedených statistik bude tato diplomová práce při diskutování optimalizací pro webové vyhledávače zaměřena primárně na webový vyhledávač Google, protože dominuje jak na světovém, tak i na českém trhu.

3.2.1.2 Google PageRank

Google PageRank je algoritmus navržený spoluzakladateli společnosti Google Larrym Pagem a Segeyem Brinem. Jeho účelem je posuzovat důležitost webových stránek. Důležitost je určována především množstvím a kvalitou hyperlinků vedoucích na danou stránku z ostatních webových stránek. Na základě tohoto algoritmu má být docíleno zobrazování pouze relevantních stránek, jejichž obsah odpovídá vyhledávacím dotazům.

Myšlenka tohoto algoritmus je založena na stejném principu jako algoritmus hodnocení důležitosti akademických prací. Hlavními faktory zohledňovanými při výpočtu je množství a kvalita odkazů vedoucích na stránku, dále počet odkazů ze stránek, které na hodnocený web odkazují, a následně jednotlivé hodnoty Google PageRank těchto odkazujících stránek.

Hodnota Google PageRank je jedním z faktorů, podle kterého se vyhodnocuje, v jakém pořadí se uživateli zobrazí webové stránky ve vyhledávači Google. Přibližné hodnoty Google PageRank pro jednotlivé stránky v současné době společnost Google již veřejně neposkytuje. (8)

3.2.1.3 Práce Googlebotu

Googlebot je obecné označení pro dva typy webových crawlerů používaných společností Google. První typ crawleru má za úkol simulovat chování uživatele webové stránky, který přistupuje na web z desktopového počítače. Druhý typ pak chování uživatele přistupujícího z mobilního zařízení.

Práce Googlebotu probíhá v několika krocích. V prvním se ujistí, zda webová stránka, ke které chce přistoupit nezakazuje přístup robotům (dá se specifikovat v souboru robots.txt, viz kapitola robots.txt). Pakliže webová stránka přístup nezakazuje, je na ni odeslán HTTP požadavek. Z obsahu odpovědi jsou získány veškeré další odkazy uvedené v HTML atributech *href*, které jsou následně zařazeny do tzv. fronty URL. Z té pak dále crawler postupně odebírá URL, a z obsahu k nim příslušných webových stránek získává stejným postupem další odkazy. Obsahy stránek, ke kterým Googlebot takto přistoupil, následně zaindexuje. Tzn.: automaticky zanalyzuje jejich obsah, pokusí se mu porozumět, a veškeré informace pak uloží do databáze zvané Google index. Data z této databáze jsou pak používaná pro nalezení nejvíce relevantních webových stránek jako odpovědí vyhledávacímu výrazu uživatele ve vyhledávači Google. (9)

3.2.2 On-page SEO

První disciplínou optimalizace webů pro vyhledávače je On-page SEO, který se zaobírá optimalizací obsahu webu a jeho architektury. (10) Konkrétní faktory řešené v rámci On-page SEO jsou popsané v následujících podkapitolách.

3.2.2.1 Meta tagy title a description

Mezi nejzásadnější On-page SEO faktory spadá optimalizace HTML meta tagu title a HTML meta tagu description. Jedná se o značky umisťované do hlavičky HTML dokumentu, které stručně popisují téma obsahu daných stránek, a to jak uživatelům, tak i vyhledávačům.

Obsah značky title je zobrazován ve webových prohlížečích, na stránkách výsledků vyhledávání, a případně na některých externích stránkách odkazujících na danou webovou stránku. Optimální velikost obsahu meta tagu title by měla být mezi 60 až 65 znaky, aby byl text správně vykreslen vyhledávači v plné velikosti.

Meta tag description je určen pro delší a konkrétnější popis obsahu stránek. Není viditelný v rámci webové stránky jako takové, ale společně se značkou title bývá zobrazován ve výsledcích vyhledávání. Jeho optimální velikost je 150 až 160 znaků.

Obsahy tagů title a description by měly být pro každou stránku jedinečné. (11)

3.2.2.2 Využití klíčových slov

Klíčová slova jsou slova či fráze, pomocí nichž vyhledávače určují relevantnost webové stránky vzhledem k zadanému vyhledávacímu dotazu. Představují zkratkovitá shrnutí témat, kterými se weby zabývají. Správné využívání klíčových slov v obsahu webu umožňuje vyšší umístění webových stránek ve výsledcích vyhledávačů pro dané vyhledávané fráze, protože algoritmy podle nich lépe dokážou vyhodnotit nakolik webová stránka odpovídá vyhledávacímu dotazu. (12)

U klíčových slov se primárně sleduje jejich hledanost a konkurence. Hledanost klíčového slova udává, kolikrát v průměru za měsíc jej někdo použil jako vyhledávací dotaz u konkrétního vyhledávače. Konkurence klíčového slova pak určuje přepočítanou míru konkurence ostatních webů pro toto slovo. Čím vyšší je konkurence daného klíčového slova, tím těžší je pro weby umístit se pro toto slovo na předních příčkách vyhledávače. Horní hranice hodnoty konkurence není stanovena. (13)

Klíčová slova rozdělujeme na obecnější a na delší a více specifická, tzv. „long tail klíčová slova“. Obecnější klíčová slova mají obvykle výrazně vyšší hledanost než ta více specifická, zároveň je však pro ně mnohem obtížnější optimalizovat obsah stránek, protože tato slova mají obvykle vysokou konkurenci. Long tail klíčová slova mají nižší hledanost, zároveň však ale povětšinou i nižší konkurenci. Díky své specifičnosti navíc spíše vedou

k vyššímu konverznímu poměru (osoba vyhledávající frázi „pánev prodej cena“ má větší předpoklady k zakoupení produktu, než osoba vyhledávající pouze slovo „pánev“). (12)

Při umisťování klíčových slov do textových obsahů stránek je nejdůležitější jejich výskyt především v prvních 150 slovech obsahu. Je tomu tak, protože například vyhledávač Google přidává vyšší váhu obsahu umístěnému výše na stránce. Dále je lepších výsledků dosaženo při přidávání klíčových slov do nadpisů stránek. Využívání značek h1 a značek nadpisů nižší úrovně navíc vyhledávačům pomáhá lépe pochopit strukturu obsahu stránek. Klíčová slova by se měla dále nacházet v meta tazích title a description, v jednotlivých URL jako názvy obrázků a v jejich *alt* popiscích. (14)

Vyšší frekvence využití klíčových slov v textu může přinášet lepší výsledky v pořadí v rámci vyhledávačů. Text stránek by však měl být stále primárně určen spíše lidem. Optimálním vyvážením technik SEO a psaní kvalitních marketingových textu se zabývá SEO copywriting. (15)

3.2.2.3 Další faktory On-page SEO

Mimo využívání klíčových slov lze obsah stránky z hlediska SEO optimalizovat ještě dalšími způsoby. Doporučuje se používat spíše kratší URL, která obsahují pouze malá písmena, a ve kterých jsou slova oddělena pomlčkami. Dále vytvářet unikátní, relevantní a pravidelný obsah, zařazovat aktuální rok do meta tagů title a description a využívat vlastních obrázků namísto obrázků z fotobank. Unikátnímu a aktuálnímu obsahu je totiž přisuzována vyšší důležitost. Další strategií pro On-page SEO je využívání interních linků (odkazů vedoucích na podstránku patřící pod stejný web). Především pak takových, které vedou na stránky jejichž umístění ve vyhledávačích je třeba posílit. (14)

V případě vícejazyčného obsahu se doporučuje využívat odlišných URL pro stránky se stejným obsahem v jiných jazycích než využívat dynamických změn lokalizace, při kterých hrozí, že crawlery neuloží do indexu vyhledávače veškeré jazykové varianty. (9)

3.2.3 Technické SEO

Technické SEO je podkategorií principů optimalizace webových stránek pro vyhledávače, které teoreticky spadá pod On-page SEO. Protože se ale zaměřuje více na technické aspekty obsahu webu, bývá bráno jako samostatná kategorie. (10)

3.2.3.1 Soubor sitemap

Sitemap je soubor ve formátu XML, který poskytuje vyhledávačům informace o stránkách a souborech, které se na webu nacházejí. V tomto souboru, který se hierarchicky ukládá na kořenovou úroveň webu, lze specifikovat důležitost jednotlivých stránek a souborů webu, frekvenci jejich aktualizací a dále zapsat datum jejich poslední změny. Přítomnost souboru sitemap je důležitá hlavně pro rozsáhlejší weby. Crawlery vyhledávačů mají pro weby přidělené pouze určité počty průchodů (tzv. Crawl budget), které nejsou veřejně známé. Kvůli tomuto principu se může se stát, že se crawlery k novým či aktualizovaným obsahům dostanou z důvodu vyčerpání přiřazených průchodu až za značně delší dobu. Soubor sitemap tak může crawlerům přesně určit stránky, které je potřeba nově uložit do indexu. (9) Možný obsah souboru sitemap.xml pro dvě stránky webu s definovaným časem jejich poslední modifikace a s prioritou těchto stránek je ukázán v kódu 1.

Kód 1: Ukázka souboru sitemap.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
<url>
  <loc>https://modrinkytabor.azurewebsites.net/</loc>
  <lastmod>2022-03-16T18:32:43+00:00</lastmod>
  <priority>1.00</priority>
</url>
<url>
  <loc>https://modrinkytabor.azurewebsites.net/Kontakty</loc>
  <lastmod>2022-03-16T18:32:43+00:00</lastmod>
  <priority>0.80</priority>
</url>
</urlset>
```

Zdroj: (Autor)

3.2.3.2 Robots.txt

Robots.txt je dalším souborem umisťovaným do kořenového adresáře webu. Je určen k předávání informací vyhledávačům a umožňuje zakázat průchod a indexaci v něm definovaných stránek. Dále je v rámci tohoto souboru možné specifikovat adresu, kde se

nachází soubor sitemap.xml. Pro rozsáhlejší weby může tento soubor opět dopomoct k optimálnímu využití přiřazených počtů průchodů crawleru. (16)

V souboru robots.txt je zároveň možné specifikovat pomocí Crawl-delay požadovanou dobu rozestupů mezi zasíláním požadavků crawlery z důvodu zabránění přílišného zahlcování serveru. (17) Příklad souboru robots.txt je na ukázce kódu 2.

Kód 2: Ukázka souboru robots.txt

```
User-agent: *
Disallow: /wp-admin/
Allow: /wp-admin/admin-ajax.php
Sitemap: https://www.priklad.cz/sitemap.xml
```

Zdroj: (Autor)

V souboru robots.txt je vhodné zakázat indexaci automaticky překládaných stránek, které z důvodů neoptimálních překladů a tím vygenerovaných nepřirozených větných konstrukcí mohou vyhledávače interpretovat jako spam a penalizovat tak příslušné weby. (9)

Při získávání dat ze stránek pomocí webových scraperů je vždy vhodné se ujistit, které sekce jsou pro vstup robotů povoleny a tyto požadavky majitele webu respektovat, přestože pouhá specifikace nepovolených stránek prakticky jejich těžbě nezabrání. Některé stránky využívají tzv. pastí na crawlery, kdy v rámci svého webu umístí stránku, která při detekování její těžby crawlerem na určitý čas zablokuje jeho IP adresu. Tyto stránky se pak umisťují v robots.txt do sekce zakázaných stránek k průchodu, aby byly případně blokovány pouze ty crawlery, které požadavky ze souboru robots.txt nerespektují. (17)

3.2.3.3 Použití protokolu HTTPS

Stránky, které využívají zabezpečeného protokolu HTTPS, jsou vyhledávači upřednostňovány před těmi, které používají snadno prolomitelný protokol HTTP. Webové prohlížeče navíc při návštěvě webů provozovaných na protokolu HTTP zobrazují varování o nízké bezpečnosti stránek, což může potenciálně uživatele odradit od dalšího průchodu webem. (10)

3.2.3.4 Rychlost webu

Zásadním hodnotícím faktorem webů je pro vyhledávače rychlost načítání obsahu stránek. Obecné doporučení je, že doba načtení stránek by neměla přesahovat tři sekundy. S trvající dobou načítání stránky navíc stoupá pravděpodobnost, že uživatel web opustí ještě před vykreslením veškerého obsahu. Až 40 % uživatelů opouští stránku po třech vteřinách načítání a pro 46 % uživatelů je rychlost webu zásadním faktorem při rozhodování, zda opět web navštívit.

Za pomalejší dobou načítání webu může stát například přílišná velikost souboru stránek, velký počet nekomprimovaných obrázků, nesprávné nastavení cachování, používání zastaralých technologií a přílišné zahlcení serveru, na kterém jsou stránky provozovány. (18)

Problém rychlosti načítání obrázků lze vyřešit využíváním formátů JPEG namísto formátu PNG. Doporučeno je využití formátu progressive JPEG, který umožňuje postupné načtení obrázku nejprve v nižší kvalitě a následném postupném zkvalitňováním výkresu.

V případě globální návštěvnosti webu dopomůže k jeho výraznému zrychlení také síť CDN (Content Delivery Network). Jedná se o síť pro rychlejší doručování obsahu, která funguje na bázi ukládání statického obsahu stránek v úložištích rozmístěných po celém světě, a následně jeho načítání z místa co nejbližšímu místu, kde se nachází uživatel.

K zrychlení webu může pomoci také používání HTTP/2 protokolu, který umožňuje zpracování více požadavků zároveň. (10)

3.2.3.5 JavaScript SEO

JavaScript SEO je součástí technického SEO a jeho cílem je optimalizovat obsah webových stránek, které využívají JavaScript.

Problémem využití JavaScriptu je například jeho možnost vytvářet odkazy i dalšími způsoby, než pomocí tagu „a“ s atributem *href* obsahujícím dané URL. Lze například využít funkci *onclick*. Takovéto odkazy však například Googlebot nedetekuje a příslušné stránky, na které by takový odkaz vedl, nebudou řádně zaindexovány. Přílišné využívání JavaScriptu dále může vést k celkovému zpomalení webu. (19)

Pakliže roboti pro získávání informací ze stránek, které využívají client-side JavaScript, využívají jen prostého odesílání požadavků na serveru bez emulace

internetových prohlížečů, nemusí se k veškerému obsahu webu dostat, a tedy nedetekují prvky, které jsou pro běžného uživatele jinak viditelné. (20)

3.2.4 Off-page SEO

Off-page SEO faktory jsou takové, které se na rozdíl od On-page faktorů a faktorů technického SEO nacházejí mimo řešené webové stránky. Nelze je tedy ovlivňovat tak jednoduše a vyžadují často spolupráci třetí strany. Mezi techniky Off-page SEO spadá především budování zpětných odkazů neboli hypertextových linků mířících na web z ostatních externích stránek. Vyhodnocován je počet zpětných odkazů, ale i kvalita a relevantnost webů, ze kterých jsou odkazy vedeny. Kvalita odkazů se vyhodnocuje na základě obsahové příbuznosti obou webů, stáří odkazů (novější odkazy jsou hodnoceny vyhledávači lépe) a také kde ve struktuře webu je odkaz umístěn. Další součásti Off-page SEO jsou zmínky webu na sociálních sítích. (21)

3.2.5 Zastaralé SEO praktiky

Většina vyhledávačů pravidelně aktualizují své vyhledávací algoritmy. Z tohoto důvodu některé SEO praktiky časem přestávají být funkční. Vyhledávací algoritmy tak v současnosti například dokážou rozpoznat nepřirozené využívání klíčových slov v textu. Pouhé zaplnění obsahu opakováním klíčových slov je tedy spíše kontraproduktivní. Další nefunkční strategií je cílení na přesné vyhledávací dotazy, aniž by souvisely s obsahem stránek, neboť algoritmy tyto praktiky již dokážou rozpoznávat. (22)

Do roku 2009 bylo časté rozsáhlé využívání meta tagu keywords, do kterých bylo umísťováno velké množství často nerelevantních klíčových slov bez ohledu na obsah stránky. Tato praktika nyní nepřináší žádné výsledky, protože po roce 2009 přestal být tomuto tagu přidělován algoritmy význam. (23)

3.2.6 Black Hat SEO

Black Hat SEO jsou neetické a zakázané SEO praktiky jejichž cílem je rychlé zlepšení pozice webových stránek, které ale po odhalení mohou vést naopak k penalizaci či blokování stránek webovými prohlížeči. Mezi takové praktiky spadá například cílená duplikace obsahu stránek, na kterou je pohlíženo jako na plagiátorství. Crawlery procházející weby tak veškeré neunikátní obsahy ignorují. Dalšími nepodporovanými praktikami jsou nákupy zpětných odkazů namísto jejich získávání přirozenou cestou,

záměrné zobrazování uživatelům jiného obsahu než crawlerům, a používání zavádějících nadpisů, které správně nereferují, o čem je příslušný obsah. (24)

3.3 Grafová analýza webu

Na strukturu webových stránek, které jsou mezi sebou propojeny hyperlinky lze, pohlížet jako na orientovaný graf, jehož uzly představují stránky webu a orientované hrany jednotlivé odkazy. Pakliže tedy například stránka A bude mít ve svém obsahu odkaz na stránku B, ale stránka B zpětný odkaz na stránku A v sobě mít nebude, dá se tento vztah znázornit grafem s uzly A a B, kde z uzlu A povede orientovaná hrana do uzlu B. Tímto způsobem pak lze znázornit veškeré stránky webu a vytvořit tak tzv. webový graf. Na přístupu propojení všech webů v systému World Wide Web je založen i výše diskutovaný algoritmus Google PageRank. (25)

3.3.1 Centrality

Centrality jsou koncept vycházející z teorie grafů, který je určen k identifikaci zásadních uzlů v grafu. Tato důležitost uzlů se posuzuje v závislosti na různých specifikacích. Z tohoto důvodu existuje více druhů centralit, kde každá definuje metriky důležitosti různým způsobem.

První takovou je centralita měřená stupněm uzlu (degree centrality). Stupeň uzlu je v případě neorientovaného grafu počet hran, které s uzlem incidují. V případě orientovaného grafu pak rozlišujeme vstupní stupeň vrcholu (počet hran vstupujících do uzlu) a výstupní stupeň vrcholu (počet hran vystupujících z uzlu). Centralita měřena stupněm uzlu pak považuje za důležité ty uzly, které mají vyšší stupeň. (26) V případě grafu reprezentujícího web jsou pak za nejdůležitější pokládané ty stránky do kterých (resp. ze kterých) vede nejvíce unikátních odkazů. Vzorec pro centralitu měřenou vstupním, respektive výstupním stupněm uzlu je zobrazen níže.

$$C_D(v_i) = \frac{deg^{in}(v_i)}{n} \tag{1}$$

$$C_d(v_i) = \frac{deg^{out}(v_i)}{n} \tag{2}$$

kde

deg^{in} (deg^{out}) je výstupní, respektive výstupní stupeň uzlu v_i;

n je celkový počet uzlů v grafu. (27)

Další mírou centrality je blízkost polohy ve středu (closeness centrality). Ta přikládá uzlům důležitost na základě jejich blízkosti ke všem ostatním uzlům. Získává se jako součet geodetických vzdáleností mezi řešeným uzlem a ostatními uzly v grafu. Geodetická vzdálenost dvou uzlů v grafu je definována jako počet hran mezi těmi uzly, které se nacházejí na jejich nejkratší cestě. (26) V grafu webu by měly hodnotu této centrality největší ty stránky, ze kterých vede nejrychlejší cesta po celém webu skrze odkazy. Vzorec pro výpočet je zobrazen níže.

$$C_C(v_i) = \frac{n-1}{\sum_j d_{ij}} \tag{3}$$

kde

d_{ij} je nejkratší cesta mezi uzly v_i;

v_j a n je celkový počet uzlů v grafu. (27)

Poslední zde diskutovanou centralitou je centralita středové mezipolohy (betweenness centrality). Její hodnota je vypočítaná dle toho, kolikrát se řešený uzel vyskytuje na nejkratší cestě mezi všemi páry uzlů v grafu. Takové uzly tvoří tzv. mosty v grafu a jejich odstranění by vedlo k narušení propojenosti či komunikace v analyzované struktuře. Takto se dají detekovat úzká hrdla webů. (26)

$$C_B(v_i) = \frac{\sum \frac{\sigma(v_j, v_k | v_i)}{\sigma(v_j, v_k)}}{(n-1)(n-2)/2} \tag{4}$$

kde

$\sigma(v_j, v_k)$ je počet nejkratších cest z uzlu v_j, do uzlu v_k;

$\sigma(v_j, v_k \mid v_i)$ je počet těch nejkratších cest, které procházejí uzlem v_i;
n je počet uzlů v grafu. (26)

3.4 Vybrané implementační technologie webového scraperu

Nejvíce používaným programovacím jazykem pro vývoj scraperů je v současné době jazyk Python, který obsahuje velké množství knihoven dedikovaných této problematice. (1)

3.4.1 Python

Python je univerzální, vysokoúrovňový, interpretovaný open source programovací jazyk. Je primárně objektově orientovaný, ale podporuje i další programovací paradigma nad objektový rámec, jako je procedurální a funkcionální programování. Python je přenositelný a spustitelný na vícero operačních systémech UNIXového typu, včetně systému Linux a macOS, a dále také na systému Windows. Autorská práva na programovací jazyk Python vlastní nezisková organizace The Python Software Foundation. (28)

Jazyk Python má využití ve vícero aplikačních doménách a nabízí tisíce modulů třetích stran. Jeho možné využití je například při vývoji webových aplikací díky frameworkům Django či Flask, při obecné práci s internetovými protokoly nebo při vědeckých a datových analýzách. (29)

Rozsáhlé využití programovacího jazyka Python je umožněno především díky open source licenci, pod kterou je umožněno vývojářům třetích stran publikovat vlastní moduly a knihovny psané v tomto jazyce, a tyto moduly třetích stran i zároveň využívat. Preferovaným instalačním programem pro instalování takovýchto modulů je program pip. Instalace je možné provádět v rámci tzv. virtuálního prostředí, které umožňuje instalaci balíčků třetích stran jen v rámci konkrétních aplikací, kde je jich využíváno. To pomáhá především k zamezení konfliktům mezi verzemi jednotlivých instalací. Standartním nástrojem pro vytváření virtuálních prostředí je nástroj venv, který je součástí Pythonu od verze 3.3. (28)

3.4.1.1 Knihovna Requests

Knihovna Requests je modulem programovacího jazyka Python vytvořené třetí stranou, kompatibilní s verzí jazyka Python 2.7 a výše. Umožňuje odesílaní HTTP

požadavků a zpracovávání jejich odpovědí, které jsou knihovnou i automaticky dekódovány. Pomocí těchto funkcionalit je tedy například možné získat obsahy webových stránek pro další účely web scrapingu. (30)

Alternativní možností při potřebě odesílání HTTP požadavků ke knihovně Request je knihovna urllib, která je součástí standardní verze Pythonu, a která je využitelná především při základních GET a POST požadavcích. (31)

3.4.1.2 Dekorátory

Dekorátory jsou v pythonu speciální funkce založené na principu návrhového vzoru dekorátor. Jako vstupní parametry přijímají funkce, jejichž chování je třeba určitým způsobem modifikovat. Takto modifikované funkce jsou pak výstupem dekorátorů. Volání dekorátorů probíhá za pomocí jejich umístěním nad dekorovanou funkcí se symbolem zavináče. Specifickými dekorátory jsou *@classmethod* a *@staticmethod*, které z funkce vytvářejí metodu třídy, respektive statické metody tříd.

Využití dekorátorů může být také například umělého přidání zpoždění funkcím zajišťující procházení webových stránek, a tím zamezení hrozby zahlcení dotazovaného serveru. (31)

3.4.1.3 Analýza textového obsahu v jazyce Python

Mnoho základních operací v rámci analýzy textu, například detekce výskytů klíčových slov v textu stránky, je v Pythonu možné vykonávat za pomocí metod třídy String. V případě potřeby normalizace textu či práce s regulárními výrazy je pak možné využít základní vestavěné moduly Pythonu, jako jsou knihovny re a unicodedata. (31)

Externí knihovna langdetect umožňuje s nastavitelnou pravděpodobní detekovat jazyk analyzovaného textu, a tímto způsobem také určit jazyk obsahu webové stránky. Podporována je detekce až 55 jazyků. (32) Pomocí knihovny readtime je pak možné určit přibližnou dobu, jakou uživateli četba analyzovaného textu zabere. (33)

3.4.2 Beautiful Soup

Jednou z knihoven napsaných v jazyce Python, která se zaobírá webovým scrapováním je knihovna Beautiful Soup, která je určena k získávání dat ze souborů ve formátu HTML a XML. Aktuální verze této knihovny je označena verzí 4 a pro její použití je zapotřebí HTML parseru pro syntaktickou analýzu textu. (34)

HTML a XML parsery se používají při potřebě interpretace HTML a XML entit, jako je například entita * *, která reprezentuje nezlomitelnou mezeru. (31) Možné je využití HTML parseru poskytovaného ve standardní knihovně Pythonu, nebo využít některého parseru třetí strany. Pakliže analyzovaný dokument není validní, jednotlivé parsery ho mohou interpretovat různými způsoby. Mezi knihovnou Beautiful Soup podporované parsery patří html.parser, lxml a html5lib. Jednotlivé parsery se od sebe pak odlišují rychlostí zpracování dokumentů, závislostí na dalších knihovnách a striktností při interpretaci výstupů.

Parsovaný analyzovaný dokument je dále reprezentován objektem vytvořeným BeautifulSoup konstruktorem. Vytvoření tohoto objektu a následná práce s ním je prezentována na ukázce kódu číslo 3.

Kód 3: Ukázka použití třídy BeautifulSoup

```
from bs4 import BeautifulSoup

soup = BeautifulSoup("<html>Web page content</html>", "html.parser")

links = soup.findAll('a', href=True)
```

Zdroj: (Autor)

Při další práci s dokumentem se pak operuje se čtyřmi základními objekty: Tag, NavigableString, již zmiňovaný objekt BeautifulSoup a Comment.

Objekt Tag je ekvivalentní k HTML a XML tagům z původních dokumentů. Pomocí metody *findAll* je možné získat veškeré tagy dokumentu příslušného typu. Tímto způsobem se tak lze dostat ke všem odkazům webové stránky pro potřeby crawleru. Každý tag má svůj slovník atributů, ke kterým lze přistupovat klasickým způsobem programovacího jazyka Python.

NavigableString označuje text uvnitř tagů a Comment komentáře v HTML a XML dokumentech. S objektem NavigableString je možné pracovat jako s běžným objektem typu String v jazyce Python.

Pomocí knihovny Beautiful Soup lze dále analyzovat kaskádové styly definované v tagu *style* či JavaScriptové skripty vložené v tagu *script*.

Knihovna Beautiful Soup výrazně zjednodušuje automatizovanou práci s HTML soubory, při kritériu rychlosti zpracovávání dokumentů je však lepší variantou přímá práce s výše popisovanými parsery, které knihovna využívá. (34)

3.4.3 Selenium

Selenium je název souboru open-source projektů, které vznikly pro umožnění automatizovaného používání a testování webových prohlížečů. Jednou z odnoží Selenia je i knihovna v jazyce Python, která je taktéž pojmenována Selenium.

Selenium pro spouštění a ovládání prohlížečů využívá protokolu WebDrive. Práce s prohlížeči může být provozována lokálně i vzdáleně. Podporována je práce například s prohlížečem Google Chrome, Firefox či Safari.

Přestože je projekt primárně zamýšlen k automatizovaném testování, má své využití i při webovém scrapingu, a to především díky umožnění spouštění JavaScriptového kódu v emulovaném prohlížeči. Dále je možné skrze takovýto prohlížeč například pořizovat screenshoty obrazovek či automatizovaně ovládat tlačítka na stránce.

Pro práci se Seleniem je potřeba operovat na serveru, kde jsou scrapovací skripty provozovány, s binárními soubory používaných driverů prohlížečů a také přímo se souborem obsahující aplikaci samotného prohlížeče. Po předání Seleniu umístění těchto souborů je možné z kódu automatizovaně otevřít příslušný webový prohlížeč a v něm webovou stránku dle zadaného URL. S obsahem této stránky je pak možné dále pracovat a přistupovat k jeho jednotlivým prvkům. Pro zamezení reálného otevírání oken prohlížeče během těžby dat z webu je možné je spustit v tzv. „headless“ módu. (35)

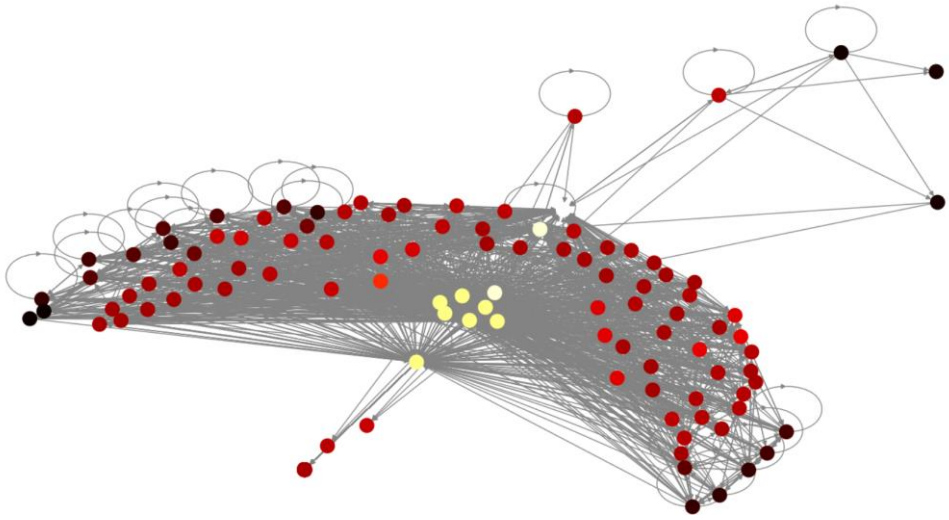
Oproti knihovně Beautiful Soup knihovna Selenium umožňuje práci i s dynamicky se měnícími webovými stránkami využívající JavaScript. (34)

3.4.4 NetworkX

NetworkX je open source modul jazyka Python sloužící k tvorbě a analýze sítí a grafů. Mezi podporované typy grafů patří orientovaný a neorientovaný graf a multigraf. NetworkX umožňuje i základní vizualizace vytvářených grafů. Jeho primárním účelem je však především jejich analýza a při potřebě grafických vyjádření podporuje spíše využívání jiných, dedikovaných nástrojů k tomuto účelu, jako je například software Graphviz.

Grafy je možné vytvářet automatizovaně skrze zadané uzly a hrany ve více podporovaných formátech, a je tedy i možným nástrojem pro vytváření webových grafů. Součástí knihovny jsou zároveň i funkce pro výpočty několika typů centralit, včetně dříve diskutované centrality měřené stupněm uzlu, centrality středové mezipolohy a centrality blízkosti polohy ve středu. Hodnoty všech vypočítaných centralit jsou vždy normalizované vzhledem k maximálním stupňům grafů a lze je zobrazit pouze jako prosté číselné výstupy, či je zároveň graficky zohlednit ve vykresleném grafu, například pomocí tónů barev jednotlivých uzlů. Ukázka takového výstupu webového grafu se zohledněním hodnot centrality blízkosti polohy ve středu je níže na obrázku 1.

Obrázek 1: Webový graf se zohledněním hodnot centrality blízkosti polohy ve středu



Zdroj: (Autor)

Součástí knihovny je i funkce pro výpočet PageRank jednotlivých uzlů grafu a funkce pro určení uzlů, se kterými daný uzel přímo sousedí, nebo naopak nesousedí, a výpočet nejkratších cest grafem. (36)

3.5 Vybrané implementační technologie pro tvorbu webové aplikace

3.5.1 Flask

Flask je framework napsaný v jazyce Python, který je určený pro tvorbu webových aplikací. Jedná se o tzv. mikroframework, který se snaží o udržení jádra aplikace na jednoduché a škálovatelné úrovni. Neobsahuje tedy například žádný integrovaný způsob

mapování dat do databáze, ale umožňuje připojování dalších modulů, které tyto a další funkcionality umožňují.

Flask využívá rozhraní WSGI (Web Server Gateway Interface), (37) což je protokol, který udává způsob komunikace mezi webovým serverem a webovou aplikací a který byl vyvinut pro využívání v rámci aplikací psaných v jazyce Python. (38) Pro práci s WSGI a jednotlivými serverovými požadavky a odpověďmi Flask využívá sadu nástrojů Werkzeug. (37) Werkzeug dále napomáhá i k zajišťování bezpečnosti aplikace, například poskytuje nástroje k hashování hesel ukládaných v databázi a zabezpečování dalších citlivých údajů, se kterými se v rámci aplikace může pracovat. (39)

Flask je možné používat díky integrovanému šablonového systému jinja i k vývoji Frontendu. (37)

3.5.1.1 Struktura Flask aplikací

Základním pilířem každé aplikace psané pomocí frameworku Flask je tzv. instance aplikace. Jedná se o objekt, který má na starosti zpracování veškerých požadavků obdržených ze strany klienta. Během tohoto zpracování se řídí výše zmíněným protokolem WSGI.

Pro určení toho, jaký kód aplikace se má spustit pro konkrétní URL, na které byl odeslán požadavek, slouží systém trasování (routes). Jednotlivé funkce v kódu, je tedy třeba označit příslušnými Python dekorátory se specifikací URL adresy, ke které jsou přiřazeny. Na ukázce kódu číslo 4 je příklad, kdy při vstupu na hlavní úroveň stránky je pomocí funkce index vrácen příslušný HTML obsah. Instance aplikace je zde pojmenována jako „aplikace“.

Kód 4: Ukázka kódu ve Flask

```
aplikace = Flask(__name__)

@aplikace.route('/')
def index():
    return "<h1>Nadpis</h1>"
```

Zdroj: (Autor)

V rámci odpovědí serveru pak lze zároveň odesílat i stavové kódy HTTP a tím například server upozornit na nevalidní vstupy. Uvnitř dekorátoru lze také specifikovat, jaký typ HTTP žádosti funkce zpracovává. Defaultně je nastavena metoda GET.

Routy mohou být specifikované i v rámci tzv. blueprintů. Jedná se o objekty, které schraňují sobě příbuzné routy a umožňují jejich hromadnou modifikaci a zároveň pomáhají k přehlednější struktuře aplikace. (39)

3.5.1.2 Flask-SQLAlchemy

SQLAlchemy je databázový framework, který slouží jako ORM a umožňuje v rámci aplikace správu relačních databází. Podporuje práci s řadou databázových systémů, včetně například MySQL či PostgreSQL. Flask-SQLAlchemy je pak knihovna, která usnadňuje použití frameworku SQLAlchemy v rámci kódu aplikace psané za pomocí frameworku Flask. Připojení k databázi skrze Flask-SQLAlchemy probíhá pomocí databázových URL obsahující typ databázového serveru, název databáze a jméno a heslo pro přístup k databázi.

Veškeré operace nad databází se v kódu provádějí pomocí instance třídy SQLALchemy. Jednotlivé relační tabulky databáze jsou reprezentovány jako třídy a označují se jako modely. Obsahy těchto tříd pak odpovídají jednotlivým atributům tabulek databáze. V rámci specifikace atributů je pak možné definovat konkrétní omezení nad vkládaným obsahem, primární a cizí klíče tabulek a konkrétní datové typy. (39)

3.5.1.3 Flask-Mail

Pro automatizované e-mailové notifikace uživatelů, obstarávání funkcionality e-mailového ověřování nových uživatelů či řešení zapomenutého hesla je v rámci aplikace Flask možné využít modulu Flask-Mail. Knihovna funguje na bázi připojování k SMTP serveru, kterému předává e-maily k doručení. Odesílání lze provádět bez autentizace skrze port 25 a lokální server, nebo je možné po nastavení konfiguračních souborů odesílat zprávy skrze externí SMTP server, například pomocí Google Gmail účtu. (39)

3.5.2 Vue.js

Vue.js je JavaScriptový framework pro vytváření uživatelských rozhraní. Svou funkcionalitou a syntaxí je podobný frameworkům Angular a React. Řadí se mezi tzv. progresivní frameworky, které umožňují vyvíjet webovou aplikaci po částech, které jsou

mezi sebou vzájemně nezávislé. Vue.js je možné použít i v již existujících aplikacích bez nutnosti přepisování jejich kódu. Jeho primární využití je pro tvorbu single-page aplikací. (40)

Vue.js je reaktivní framework a při každé změně dat provede automatické přerenderování šablony. Obsah stránky se tedy takto může měnit bez nutnosti jejího znovunačtení uživatelem v prohlížeči. (41)

Paradigma Vue.js je rozložení webové stránky do samostatně fungujících, nezávislých komponent, které mají vlastní statickou HTML šablonu a JavaScriptové metody, a jsou schopny přijímat a zobrazovat data. Jednotlivé komponenty jsou pak znovupoužitelné v dalších místech aplikace. Části aplikace mají v sobě zapouzdřena data, která jsou definována ve formátu JSON a ke kterým je pak možné přistupovat v rámci statické šablony komponenty. Na základě hodnoty dat je dále možné určit, které části statické HTML šablony mají být skutečně uživateli zobrazeny, či přímo v šabloně procházet a vypisovat datové struktury za pomocí cyklů. Data si pak mezi sebou jednotlivé komponenty předávají za pomocí proměnných. (41)

Výhodou využití frameworku Vue.js k tvorbě webu z hlediska SEO je umožnění vykreslování obsahu na straně serveru, a tím vyhnutí se client-side JavaScriptu, které některými crawlery nemusí být správně detekováno. (40)

3.5.2.1 Axios

Při využití rozdílných technologií pro implementaci backendové a frontendové části aplikace (tedy například právě Flasku jako backend a Vue.js jako frontend) je potřeba zajistit jejich vzájemnou komunikaci a předávání dat z backendu na frontend. K tomuto účelu lze ve Vue.js využít knihovny Axios, která funguje jako HTTP klient. Pomocí knihovny Axios lze vytvářet asynchronní HTTP požadavky s použitím vybraných HTTP metod, a data získaná z odpovědí pak zpracovávat a prezentovat uživateli. (42)

3.5.3 JSON Web Tokens

JSON Web Token (JWT) je objekt ve formátu JSON, který umožňuje bezpečnou výměnu dat mezi dvěma stranami a ověření autenticity těchto dat. Účelem JWT tedy není uchránit data před zobrazením, ale zabránit jejich modifikaci. Jeho využití je například pro udržení informace o přihlášeném uživateli mezi backendovou a frontendovou stranou aplikace.

JWT se skládá ze tří částí. První z nich je hlavička, kde je specifikován typ algoritmu, kterým se bude kódovat podpisová část JWT. V druhé části objektu jsou pak samotná předávaná data, jako je například e-mailová adresa přihlášeného uživatele a dále je zde určena expirační doba tokenu, po které je nutné opětovné přihlášení uživatele a tím i vytvoření nového tokenu. Poslední částí JWT je podpis, kterým se ověřuje autenticita dat. (43) Knihovna umožňující vytváření JSON Web Tokenů v rámci programů psaných v jazyce Python nese název PyJWT. (44)

3.5.4 Docker

Docker je softwarový nástroj umožňující provozuschopnost aplikací na různých platformách za pomocí jejich izolace v kontejnerech.

Docker kontejnery se skládají ze sady virtuálních prostředků, které vyvolávají iluzi, že aplikace má veškeré zdroje počítače, na kterém je provozována pouze pro sebe. Veškeré aplikace izolované v Docker kontejnerech běžící na totožném stroji sdílí operační systém tohoto počítače i jeho paměť a procesor. Právě sdílení těchto prostředků počítače zajišťuje jejich vyšší rychlost a menší zdrojovou náročnost oproti virtuálním strojům, které obsahují i vlastní operační systém.

Pro spuštění procesu izolace aplikace do Docker kontejneru se používá skriptový soubor Dockerfile. Jeho výstupem je vytvoření Docker Image, který představuje zachycení stavu souborového systému, který může obsahovat například zdrojové kódy aplikace a který je pak spouštěn v rámci kontejneru.

Pracovat s Dockerem je možné skrze příkazy v příkazové řádce či přes dedikované grafické rozhraní. (45) (46)

3.5.5 Heroku

Heroku je cloudová služba typu PaaS (Platform as a Service), která umožňuje hosting aplikací. Podporuje celou řadu programovacích jazyků, včetně jazyka Python. Nasazování aplikací v rámci Heroku je prováděno za pomocí systému Git. Kód aplikace je nahrán na Git server vlastněný Heroku a poté je automaticky spuštěna instalace, konfigurace a samotné nasazení aplikace v rámci hostitelského serveru. Běh aplikací v rámci Heroku je zajišťován v Linuxových kontejnerech zvaných dynos. Podle počtu těchto kontejnerů se také odvíjí cena za hostování aplikace. Tzv. webové dynos reprezentují webový server a v případě potřeby vyššího přístupu do aplikace je možné

počet těchto dynos navyšovat. Pro plnění podpůrných úkolů důležitý pro běh aplikace slouží tzv.: worker dyno. Správu aplikací je možné provádět skrze webové rozhraní či za pomocí programu Heroku v rámci příkazové řádky. V příkazové řádce je také možné sledovat logy hostovaných aplikací. (39)

V rámci Heroku je možné s aplikací integrovat vice služeb třetích stran. Umožňuje například podporu databáze PostgreSQL či serveru Redis.

Heroku umožňuje i hosting aplikací zdarma, pokud se jedná o nekomerční soukromé projekty. (47)

3.5.5.1 Nasazení aplikace s pomocí Dockeru

Heroku umožňuje mimo nasazování pomocí jejich Git serveru i nasazování aplikace za pomocí Dockeru. Této možnosti je využíváno při potřebě hostování aplikace psané v Heroku nepodporovaném programovacím jazyce či při použití kombinace více jazyků, například při rozdílných technologií implementující frontend a backend aplikace. (48) Lze nasazovat již vytvořené Docker image či využít souboru Heroku.yml, který umožňuje jejich vytvoření přímo pro potřeby běhu v Heroku s možností definovat potřebné přidružené program provozované na Heroku, které aplikace k běhu využívá. (47)

3.5.5.2 Omezení Heroku

Jakýkoliv HTTP požadavek na aplikaci hostované na Heroku má limit 30 sekund, ve kterém jej server musí vyřídit a vrátit příslušnou odpověď. V případě neukončení požadavku v této době je automaticky přerušen a je odeslána odpověď s chybovým číselným stavem. K překročení limitu dochází například při nechtěném zacyklení kódu, nebo při zpracovávání déle trvající úkonů, jako je například těžba dat během webového scrapování. Tento limit není v rámci Heroku nijak konfigurovatelný a je tedy vyžadována asynchronní práce scrapovacích skriptů spuštěných v rámci HTTP požadavků.

Heroku má i další řadu omezení, ty se však dále týkají především kapacity a počtu hostovaných aplikací, které v případě úkolu implementace jedné aplikace nižšího rozsahu nejsou limitující. (47)

4 Vlastní práce

4.1 Analýza uživatelských požadavků

Na základě výstupů z teoretické práce byly sestaveny požadavky na vyvíjenou webovou aplikaci. Vytvořené funkční a nefunkční požadavky se opírají především o diskutované vlastnosti webových crawlerů a scraperů (například nutné zajištění legality těžby dat a nepřetěžování serverů), dále pak o popisované možnosti analýzy webů z teoretické části, jako jsou principy SEO a centrality. Zahrnuty jsou i požadavky vyplývající z možností popisovaných implementačních technologií.

4.1.1 Funkční požadavky

4.1.1.1 F1 – Uživatelské účty

V rámci aplikace bude každý uživatel mít možnost vlastního uživatelského účtu. Jednotlivé účty mají rovné přístupy k systému – tj. není definován žádný uživatel s vyššími právy. Uživatel si může informace o svém účtu v rámci aplikace zobrazovat.

4.1.1.2 F2 – Přihlašování

Uživatelé s existujícím uživatelským účtem budou mít možnost se v rámci aplikace ke svému účtu přihlásit za pomocí e-mailu a jimi definovaného hesla. Přihlášení uživatelé mají přístup ke všem částem aplikace.

4.1.1.3 F3 – Odhlašování

Přihlášení uživatelé budou mít možnost se ručně z aplikace odhlásit a dále aplikaci využívat jako nepřihlášení uživatelé.

4.1.1.4 F4 – Registrace

Jakýkoliv nepřihlášený uživatel nevlastnící uživatelský učet v aplikaci bude mít možnost si tento účet založit. Pro registraci je nutné zadat jméno, příjmení, unikátní e-mail a zvolené heslo. Po ověření skrze odeslání kontrolní zprávy na vyplněný e-mail bude uživatelský účet vytvořen.

4.1.1.5 F5 – Zapomenuté heslo

V případě zapomenutí hesla bude mít nepřihlášený uživatel možnost zažádat si o heslo nové. Tato obnova hesla bude provedena skrze zaslání dočasného odkazu na zadaný e-mail (bude-li tento e-mail přiřazen k existujícímu uživatelskému účtu).

4.1.1.6 F6 – Weby uživatelů

Přihlášený uživatel bude mít možnost přidat ke svému účtu informaci o vlastněném webu. Weby budou identifikovány svým názvem a URL adresou. Uživatel bude moct seznam svých webů zobrazovat a spravovat.

4.1.1.7 F7 – Verifikace webů uživatelů

Přihlášený uživatel bude mít možnost verifikovat svůj web tak, že prokáže možnost úpravy kódu příslušného webu. Tato verifikace proběhne skrze vložení vygenerovaného HTML tagu a jeho umístění uživatelem do hlavičky hlavní stránky kódu webu.

4.1.1.8 F8 – Těžba dat z webů

Přihlášený uživatel bude mít možnost zahájit těžbu dat v rámci některého z jeho verifikovaných webů. Těžba dat bude probíhat za pomocí tzv. crawlerů, které budou trojího typu.

4.1.1.9 F9 – Basic crawler

Přihlášení uživatelé budou mít možnost na svých verifikovaných webech pustit tzv. Basic crawler, jehož výstupem budou základní informace o webu:

- Informace o existenci souboru sitemap.xml na webu
- Informace o existenci souboru robots.txt na webu
- Informace o použití či nepoužití protokolu HTTPS
- Informace o jazykovém rozvržení webu
- Informace o počtu externích a interních odkazů
- Informace o dobách odezvy webu po odeslání HTTP požadavku
- Informace o časech potřebných na přečtení obsahů jednotlivých stránek
- Informace o obsahu meta tagu title a description a hlavních nadpisů stránek

- Informace o obrázcích umístěných na webu, jejich formátech a popiscích

4.1.1.10 F10 – Keywords crawler

Přihlášený uživatel bude mít možnost na svých verifikovaných webech pustit tzv. Keywords crawler, jehož výstupem budou informace o přítomnosti uživatelem definovaných klíčových slov ve zkoumaných částech webu. Konkrétně budou detekována klíčová slova v:

- Meta značkách title a description
- Volně v textu webu
- Názvech souborů obrázků na webu
- V popiscích obrázků webu

4.1.1.11 F11 – Centrality crawler

Přihlášený uživatel bude mít možnost na svých verifikovaných webech pustit tzv. Centrality crawler, jehož výstupem bude sestavení webového grafu zkoumaného webu a vypočítání jednotlivých hodnot centralit pro webové stránky.

4.1.1.12 F12 – Prezentace získaných dat z webů

Přihlášený uživatel bude mít možnost u svých webů zobrazit historii záznamů o provedených těžbách dat s detailními informacemi o jejich výsledcích, které budou prezentovány pomocí tabulek a grafů. Některé výstupy crawlerů budou dále opatřeny vysvětlivkami a doporučeními.

4.1.1.13 F13 – Klíčová slova webů

Přihlášený uživatel bude mít možnost v rámci svých verifikovaných webů definovat klíčová slova, jejichž přítomnost v obsahu webu chce sledovat.

4.1.1.14 F14 – Dokumentace k webu

Webová aplikace bude obsahovat podstránku přístupnou všem uživatelům, která bude vysvětlovat práci s dostupnými nástroji aplikace a popisovat její výstupy.

4.1.1.15 F15 – Informace k webu

Webová aplikace bude obsahovat základní informace s výčtem jejích základních funkcí a kontakt na autorku aplikace.

4.1.2 Nefunkční požadavky

4.1.2.1 N1 – Cena

Za používání aplikace nebude od uživatelů vyžadována žádná platba, a zároveň webová aplikace bude provozována na platformě Heroku v režimu provozu zdarma.

4.1.2.2 N2 – Legalita

Pro zabránění zneužívání aplikace k těžbě dat z nepovolených webů bude zajištěna nutnost prokázání vlastnění analyzovaných webů uživatelem. Verifikace bude prováděna skrze vložení vygenerovaného unikátního HTML tagu do hlavičky kódu stránky. Takto verifikované weby pak budou procházeny bez závislosti na obsahu souboru robots.txt, protože průchod bude prováděn pomocí robotů spuštěných vlastníkem dané webové stránky. Ze stránek nebudou těženy žádné citlivé informace, které by nebyly volně dostupné.

4.1.2.3 N3 – Implementační technologie

Pro implementaci aplikace budou použity technologie popsané v teoretické části aplikace. Primárně tedy framework Flask a Vue.js s použitím Python knihoven Beautiful Soup a Selenium

4.1.2.4 N4 – Doba odezvy HTTP požadavků

Z důvodu omezení hostingové služby Heroku bude zajištěno, aby doba odezvy HTTP požadavků nepřesahovala 30 sekund.

4.1.2.5 N5 – Doba těžby dat

Doba jedné těžby dat z webu bude omezena časovým intervalem čtyř hodin. Po této době bude těžba automaticky ukončena a data budou zobrazena pouze pro stránky, které byly v rámci webu vytěženy v tomto časovém intervalu.

4.1.2.6 N6 – Rozšiřitelnost

Aplikace bude vytvořena tak, aby byla možná její další rozšiřitelnost o další možnosti těžby dat aj.

4.1.2.7 N7 – Prevence proti zatěžování webů

Veškeré skripty zajišťující scrapování webů budou obsahovat i umělé prodlevy mezi jednotlivými požadavky v délce jedné sekundy, aby bylo zamezeno přílišnému zatěžování analyzovaných webů.

4.1.2.8 N8 – Prevence proti zatěžování aplikace

Uživatel bude mít možnost spustit v jeden okamžik vždy jen jeden scrapovací skript v rámci jednoho verifikovaného webu. Každý uživatel bude moct v aplikaci evidovat pouze deset webů a pro každý web budou ukládány informace pouze pro padesát historických záznamů dat.

4.1.2.9 N9 – Jazyk webové aplikace

Uživatelské rozhraní vytvářené webové aplikace bude v anglickém jazyce pro možnost jeho užití větším spektrem uživatelů.

4.2 Analýza dostupných řešení

Následující kapitola se zabývá analýzou již existujících aplikací sloužících k rozboru architektury a obsahu webů. Popsány jsou vybrané aplikace zabývající se podobnou problematikou jako webová aplikace vytvářená v rámci této diplomové práce. Vynechány jsou služby typu Google Analytics, které zkoumají spíše chování uživatelů na analyzovaných webech než jejich obsah.

4.2.1 Marketing Miner

Marketing Miner je nástroj pro získávání marketingových dat. Umožňuje uživatelům například vytvářet analýzy klíčových slov s přehledem jejich hledanosti, analýzy konkurence, CPC, umístění zadaného webu ve vyhledávačích pro daný vyhledávací výraz a dalších ukazatelů. Možností je i analýza slov pro český trh. Aplikace

dále obsahuje našeptávač klíčových slov, který pomůže uživateli získat návrhy na další hledaná klíčová slova.

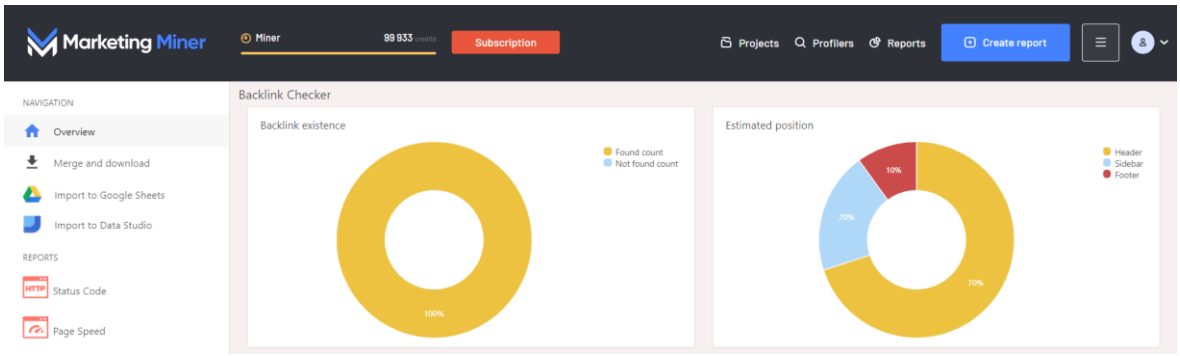
Nástroj lze využít i k analýze zadaných URL. U těch pak ověřuje například přítomnost nefunkčních odkazů či možnosti indexace analyzovaných webových stránek. Jednotlivá URL je nutné zadat samostatně, nástroj si je sám nezjišťuje v rámci příslušnosti k analyzovanému webu. Analyzovat lze i samostatnou doménu. O té nástroj zjistí například dostupné kontakty a DNS informace.

Uživatelé si mohou vybrat z předpřipravených platebních plánů lišících se přiřazenými měsíčními kredity, které jsou čerpány za prováděné analýzy. Je možné si definovat i vlastní platební plán. Nástroj je možné používat i v bezplatném režimu, jeho funkce jsou ovšem velmi omezeny. V nástroji není nijak ověřováno, zda je uživatel skutečně majitelem webu, analyzovat lze web jakýkoliv. (13)

V rámci analýzy webu Marketing Miner bylo provedeno testovací analyzování webové stránky vlastněné autorkou práce, která má ve svém robots.txt zakázanou těžbu kterékoliv podstránky webu roboty. Marketing Miner tuto stránku zanalyzoval a v závěru vyhodnotil, že je tato stránka blokována pro těžbu roboty, vytěžené informace z webu ale dostupné byly.

Ukázkový výstup z nástroje Marketing Miner je na obrázku číslo 2.

Obrázek 2: Ukázka výstupu aplikace Marketing Miner



Zdroj: (13)

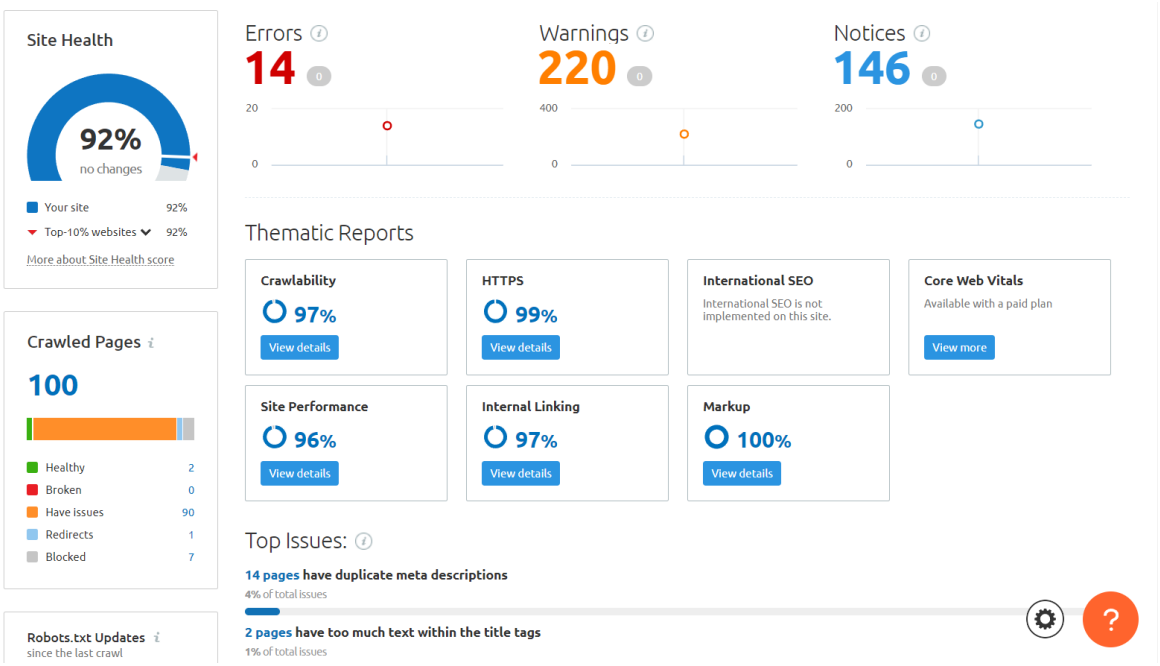
4.2.2 SEMRush

SEMRush je platforma pro zajišťování viditelnosti online obsahu a správu online marketingu. Často bývá využívána především k analýzám klíčových slov. Jedná se primárně o placený nástroj, který lze využívat i v omezeném bezplatném režimu. Mezi

dostupné analýzy patří například možnosti porovnávání webů s konkurencí, SEO audity, analýza obsahu webů a analýza trhů. (49)

Nástroj SEMRush byl otestován v režimu bezplatného využití. Při pokusu zanalyzovat webovou stránku, která obsahovala soubor robots.txt, byla tato skutečnost nástrojem detekována a blokována. Pro další analyzovanou stránku třetí strany, která průchod roboty neblokovala, pak nástroj například detekoval nedostatečně definované obsahy meta značek title a jiné informace týkající se SEO principů.

Obrázek 3: Ukázka výstupu aplikace SEMRush



Zdroj: (49)

4.2.3 Závěr analýzy

Existuje více různých aplikací zabývající se obdobnou problematikou jako webová aplikace, jež je předmětem této závěrečné práce. Hlavní odlišností vytvářené aplikace je především její dostupnost v plném rozsahu zdarma a dále nižší zaměřenost na analýzu klíčových slov, které tyto aplikace mají často jako primární doménu. Dále se mezi dostupnými řešeními nepodařilo najít žádné takové, které by se zaměřovalo na problematiku určování hodnot centralit v rámci vytvářených webových grafů

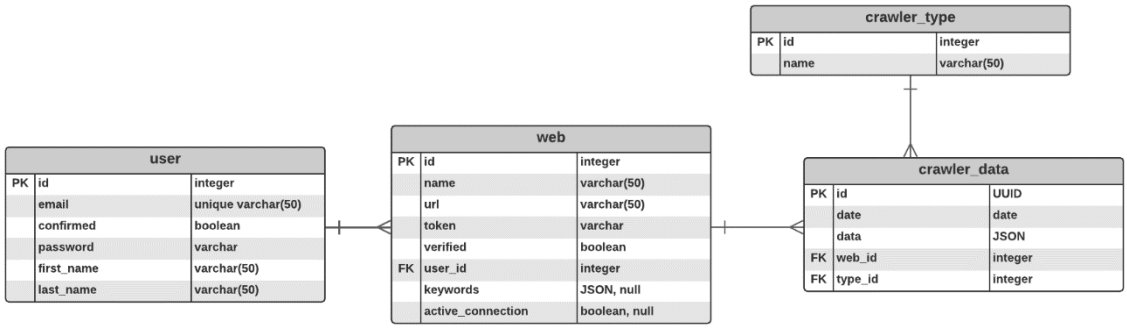
analyzovaných webů. Odlišností je také přístup k těžbě cizích webových stránek – analyzované aplikace nevyžadovali verifikaci toho, zda uživateli analyzovaný web náleží.

Analyzovaná dostupná řešení obecně pohlíží na analýzu webů i v rámci celkového prostředí internetu (například porovnávání webů s konkurencí jako funkce nástroje SEMRush), zatímco vytvářená aplikace sleduje vždy jeden konkrétní web izolovaně bez pohlížení na hledanost klíčových slov, zpětné odkazy, konkurenci aj.

4.3 Návrh databáze

V rámci webové aplikace bude potřeba uchovávat informace o těchto třech hlavních oblastech webu: uživatelích, provedených těžbách dat a analyzovaných webech. Návrh relační databáze je uveden na následujícím obrázku (obrázek 4).

Obrázek 4: Návrh databáze



Zdroj: (Autor)

4.3.1 User

Tabulka „user“ je určena pro uchovávání dat o registrovaných uživatelích. Zaznamenáváno je jméno, příjmení a unikátní e-mail uživatele. Dále je zde uloženo heslo k uživatelskému účtu, které bude z bezpečnostních důvodů ukládáno v šifrované formě. U každého uživatele je sledováno, zda je jeho e-mail ověřený a má tak právo se ke svému účtu přihlásit.

4.3.2 Web

Jednotlivé analyzované weby jsou zaznamenávány v tabulce „web“. Každý web má své jméno a URL a také vygenerovaný token skrze který se ověřuje, zda je příslušný uživatel skutečným majitelem webu. Uživatel může vlastnit žádný či více webů. Každý

web má právě jednoho uživatele, ke kterému je přiřazen. Tabulka „web“ dále obsahuje nepovinné atributy keywords a active_connection. Atribut keywords je typu JSON a obsahuje výčet klíčových slov, která jsou v rámci webu sledována. Active_connection je pomocný atribut k určení, zda v rámci webové aplikace právě probíhá analýza příslušného webu či ne.

4.3.3 Crawler_data

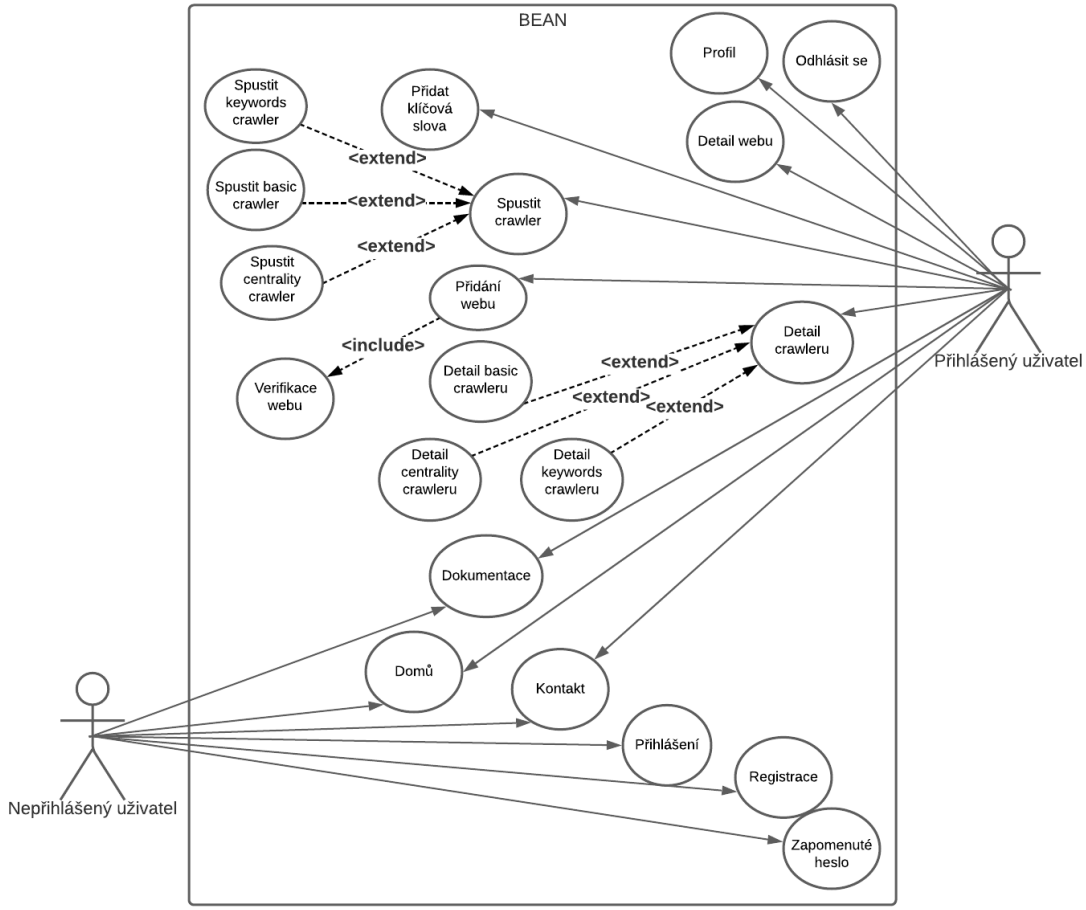
Výsledky z analýzy webů jsou uloženy v tabulce „crawler_data“. Záznam „crawler_data“ je vždy přidružen k právě jednomu webu a web může mít nula či více dat z crawlerů. O každém výstupu z crawlerů je zaznamenáno datum, kdy byla těžba dokončena, a dále jakého typu daný crawler byl (jednotlivé typy jsou uchovávány v tabulce „crawler_type“). Samotná data získaná z webů jsou ukládána ve formátu JSON.

4.4 Use Case diagram

Na základě analýzy požadavků byl sestaven seznam use casů a aktérů.

Se systémem nazvaným BEAN pracují konkrétněji dva typy aktéru: přihlášený uživatel a nepřihlášený uživatel. Use Case diagram, který ukazuje přístup aktérů k jednotlivým definovaným use casům, je uveden na obrázku číslo 5.

Obrázek 5: Use Case diagram



Zdroj: (Autor)

4.5 Pokrytí funkčních požadavků

Následující tabulka zobrazuje pokrytí jednotlivých funkčních požadavků z kapitoly 4.1.1 definovanými use casy. Funkční požadavky jsou v tabulce identifikovaný ve sloupcích se svými přiřazenými čísly. Jejich pokrytí příslušným případem užití je zobrazeno velkým písmenem x ve žlutém poli.

Tabulka 1: Pokrytí funkčních požadavků

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Kontakt															X
Dokumentace															
Přihlášení		X												X	
Registrace				X											
Zapomenuté heslo					X										
Odhlásit se			X												
Profil	X														
Detail webu						X									
Detail crawleru												X			
Spustit crawler								X	X	X	X				
Přidání webu						X									
Verifikace webu							X								
Domů															X
Přidat klíčová slova													X		

Zdroj: (Autor)

Z tabulky lze vyčíst, že veškeré definované funkční požadavky jsou pokryty.

4.6 UI specifikace

V této kapitole jsou uvedeny vybrané části z UI specifikace, která je obsahem přílohy práce. Na těchto vybraných částech je pak v kapitole „Implementace“ zároveň demonstrován výsledek webové aplikace.

4.6.1 Detail webu

4.6.1.1 Use Case

Uživatel očekává

- Zobrazení základních informací o webu;
- Možnost návratu zpět na profil uživatele;
- Možnost dostat se k editaci klíčových slov u verifikovaných webů v případě neprobíhající těžby dat;

- Možnost smazat daný web v případě neprobíhající těžby dat;
- Možnost spustit jednotlivé typy crawlerů pro verifikované weby v případě neprobíhající těžby dat;
- Seznam historických těžeb dat s odkazy k bližším informacím;
- Možnost smazání historického záznamu o těžbě dat;
- Informaci o případné probíhající těžbě dat;
- Možnost opětovné verifikace webu v případě neverifikovaných webů;
- Informace o případných chybových hlášení.

4.6.1.2 Scénář

Systém zobrazí

- Název a URL dané webové stránky;
- Štítek zobrazující informaci o tom, zda byl web verifikován;
- Tlačítko „Back to profile“
- Tlačítko pro spuštění funkce Basic crawler v případě verifikovaného webu a neprobíhající těžby dat;
- Tlačítko pro spuštění funkce Keywords crawler v případě verifikovaného webu a neprobíhající těžby dat;
- Tlačítko pro spuštění funkce Centrality crawler v případě verifikovaného webu a neprobíhající těžby dat;
- Tabulku s výčtem historických těžeb dat s jejich typem, datem ukončení těžby a odkazem na detailní informace o provedené těžbě a checkboxy pro jejich označení;
- Tlačítko pro smazání záznamů historických těžeb dat;
- Hlášku o tom, že web není verifikován – v případě neverifikovaného webu;
- Odkaz na stránku pro editaci klíčových slov webu v případě verifikovaného webu a neprobíhající těžby dat;
- Hlášku o přesáhnutí povoleného počtu padesáti záznamů dat, pokud tak nastane. Systém pak neumožní spuštění těžebních skriptů;
- Tlačítko pro smazání webu v případě neprobíhající těžby dat.

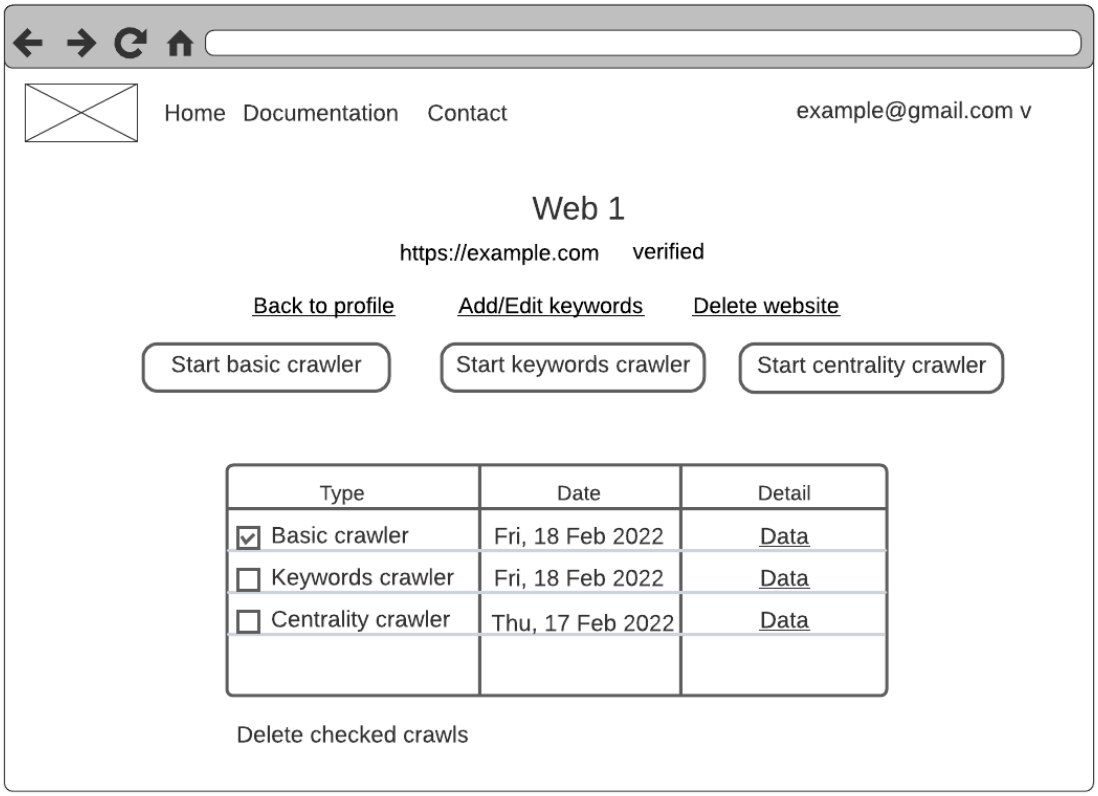
Po kliknutí na:

- Tlačítko „Back to profile“ systém zobrazí stránku profilu uživatele;

- Odkaz na stránku pro editaci klíčových slov systém zobrazí stránku pro editaci klíčových slov;
- Tlačítko smazání webu systém zobrazí dialog s otázkou, zda si je uživatel jistý svým rozhodnutím a po potvrzení smaže záznamy o daném webu;
- Tlačítko smazání historických těžeb dat systém odstraní označené záznamy;
- Odkaz na detail výstupu z crawleru systém zobrazí příslušný detailní výpis;
- Tlačítko pro spuštění konkrétních crawlerů systém spustí příslušný skript a zobrazí informaci o probíhající těžbě. V případě že v začátku těžby bude zaznamenáno, že na stránce chybí verifikační značka, těžba bude ukončena a uživatel bude na tento fakt upozorněn. Web bude označen za neverifikovaný. V případě spuštění funkce Keywords crawleru bez přiřazených klíčových slov k webu těžba neproběhne a uživatel bude na tento fakt upozorněn.

4.6.1.3 Logický design

Obrázek 6: Web detail – logický design



Zdroj: (Autor)

4.6.2 Detail „Basic crawler“

4.6.2.1 Use Case

Uživatel očekává

- Základní informace o provedené těžbě dat;
- Možnost návratu zpět na detail webu;
- Možnost zvolit si typ vypisovaných dat;
- Možnost zobrazení nápověd pro některé výstupy;
- Možnost zobrazení všech výstupů z provedené těžby dat.

4.6.2.2 Scénář

Systém zobrazí

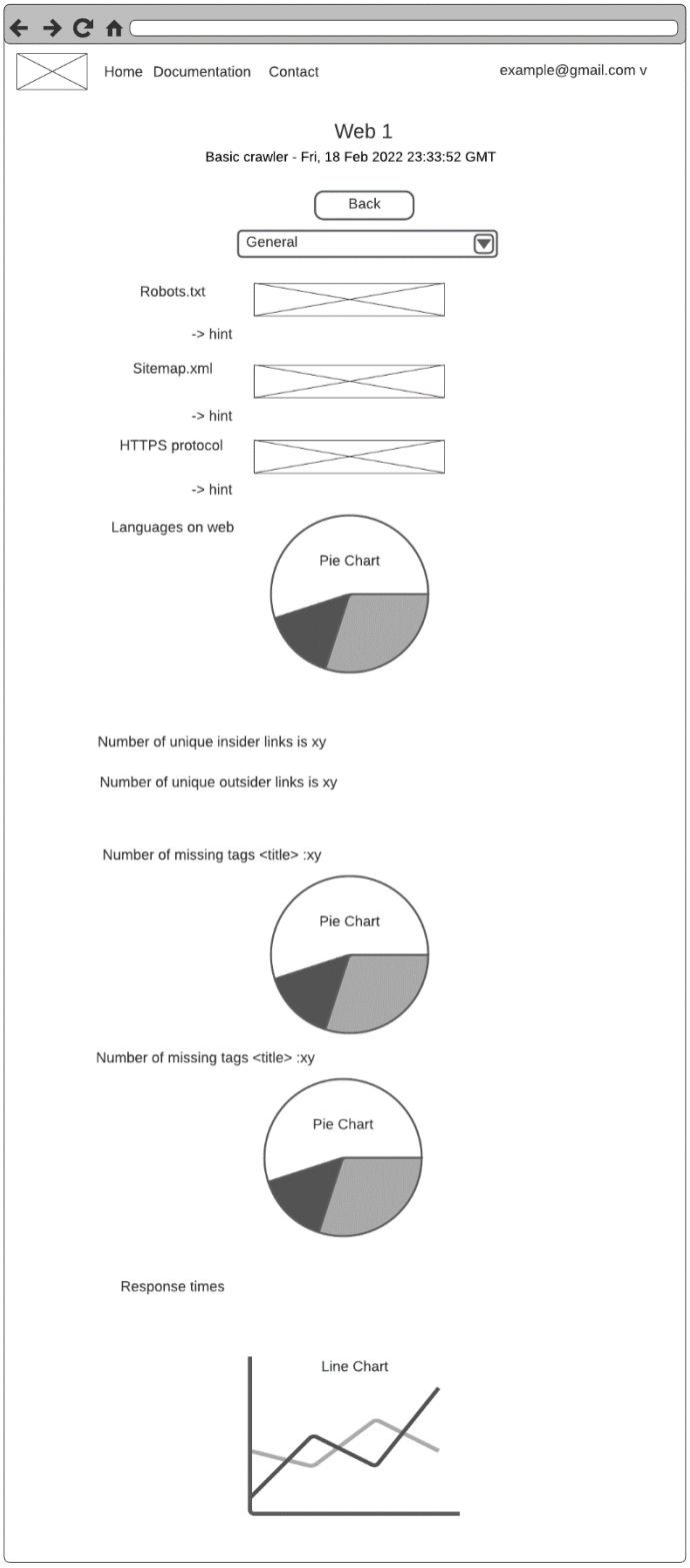
- Tlačítko pro návrat zpět na detail webu;
- Výběrový box pro zvolení typu vypisovaných dat;
- Výpis informací získaných z těžby dat.

Po kliknutí na:

- Tlačítko pro návrat zpět na detail webu systém zobrazí stránku detailu webu;
- Výběrový box systém zobrazí výběr typu vypisovaných dat – obecná data či data pro vybrané stránky webu. Po výběru změní vypisovaná data vzhledem k volbě uživatele.

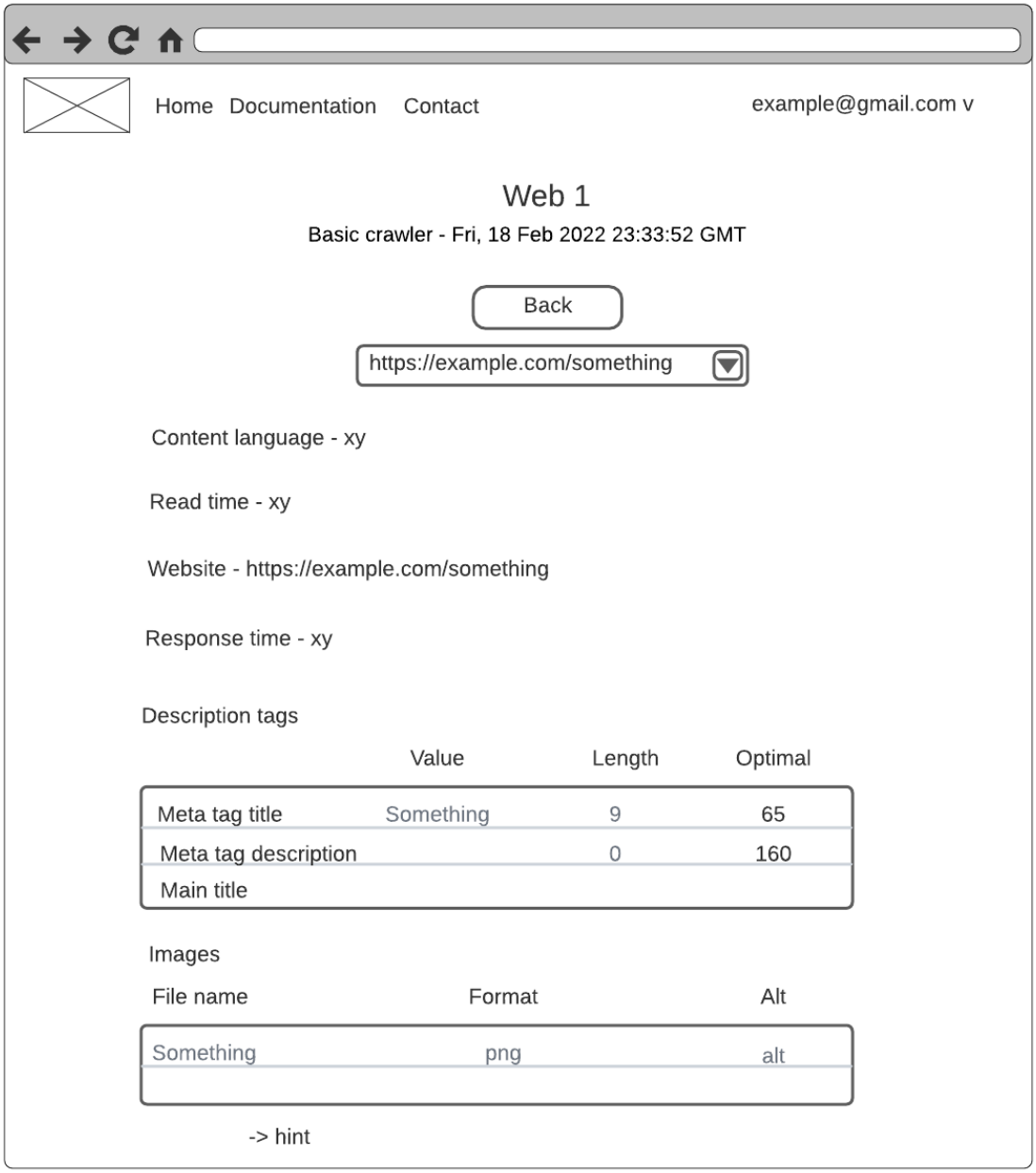
4.6.2.3 Logický design

Obrázek 7: Basic crawler detail 1 – logický design



Zdroj: (Autor)

Obrázek 8: Basic crawler detail 2 – logický design



Zdroj: (Autor)

4.6.3 Detail „Keywords crawler“

4.6.3.1 Use Case

Uživatel očekává

- Základní informace o provedené těžbě dat;
- Možnost návratu zpět na detail webu;

- Možnost zvolit si typ vypisovaných dat;
- Možnost zobrazení všech výstupů z provedené těžby dat.

4.6.3.2 Scénář

Systém zobrazí

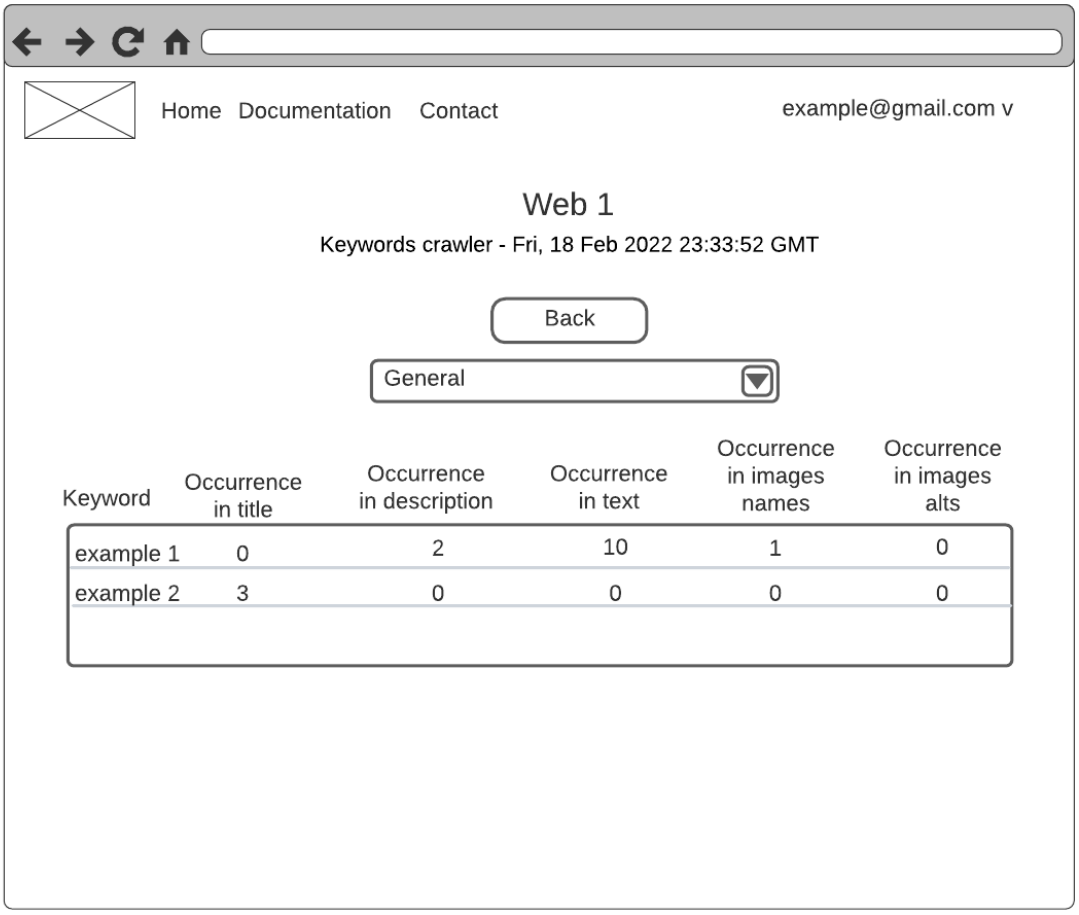
- Tlačítko pro návrat zpět na detail webu;
- Výběrový box pro zvolení typu vypisovaných dat;
- Výpis informací získaných z těžby dat ve formě tabulek s informacemi o nalezených uživatelem definovaných klíčových slovech v konkrétních částech webu.

Po kliknutí na:

- Tlačítko pro návrat zpět na detail webu systém zobrazí stránku detailu webu;
- Výběrový box systém zobrazí výběr typu vypisovaných dat – obecná data či data pro vybrané konkrétní vybrané klíčové slovo. Po výběru změní vypisovaná data vzhledem k volbě uživatele.

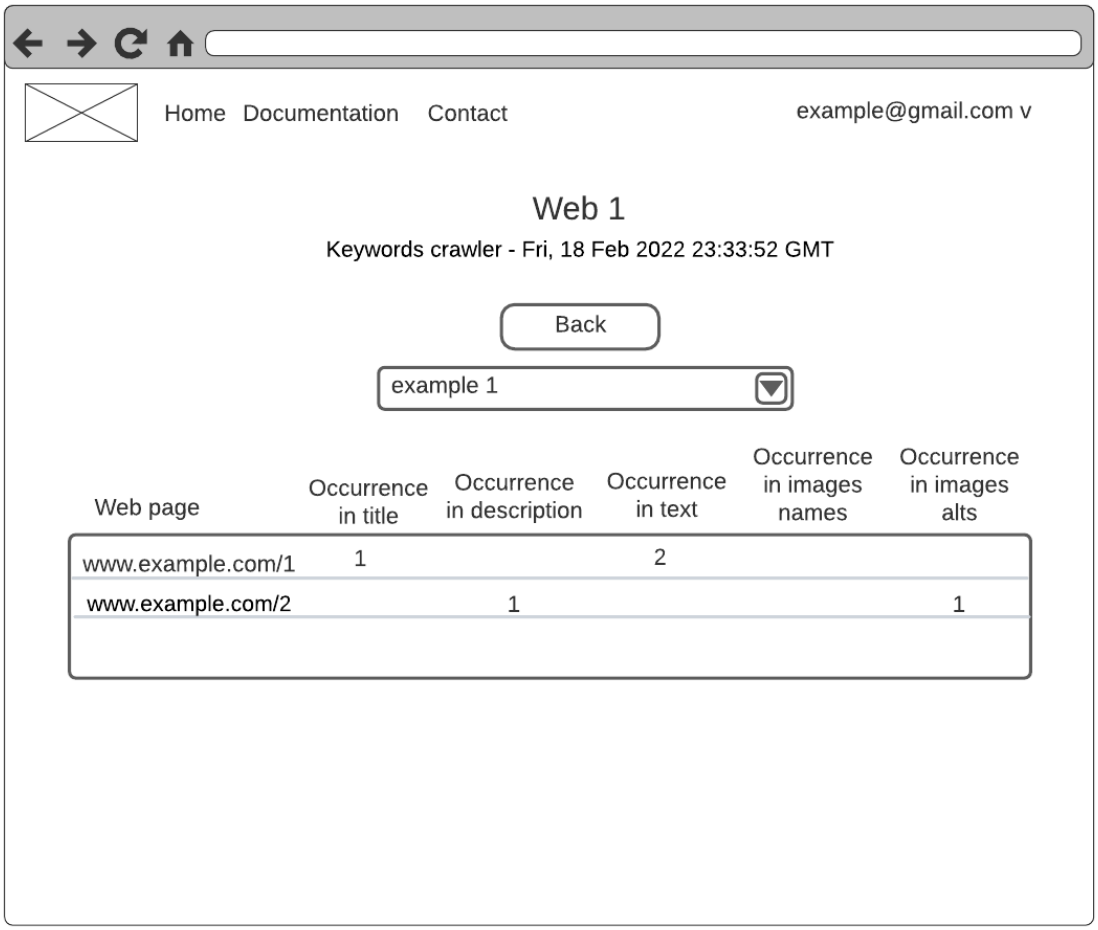
4.6.3.3 Logický design

Obrázek 9: Keywords crawler detail 1 – logický design



Zdroj: (Autor)

Obrázek 10: Keywords crawler detail 2 – logický design



Zdroj: (Autor)

4.6.4 Detail „Centrality crawler“

4.6.4.1 Use Case

Uživatel očekává

- Základní informace o provedené těžbě dat;
- Možnost návratu zpět na detail webu;
- Možnost zvolit si typ vypisovaných dat;
- Možnost zobrazení nápověd pro některé výstupy;
- Možnost zobrazení všech výstupů z provedené těžby dat;
- Možnost zobrazení nápověd pro některé výstupy;
- Možnost změny vypsaného webového grafu v závislosti na zvolených webových stránkách, které mají být v grafu zahrnuty.

4.6.4.2 Scénář

Systém zobrazí

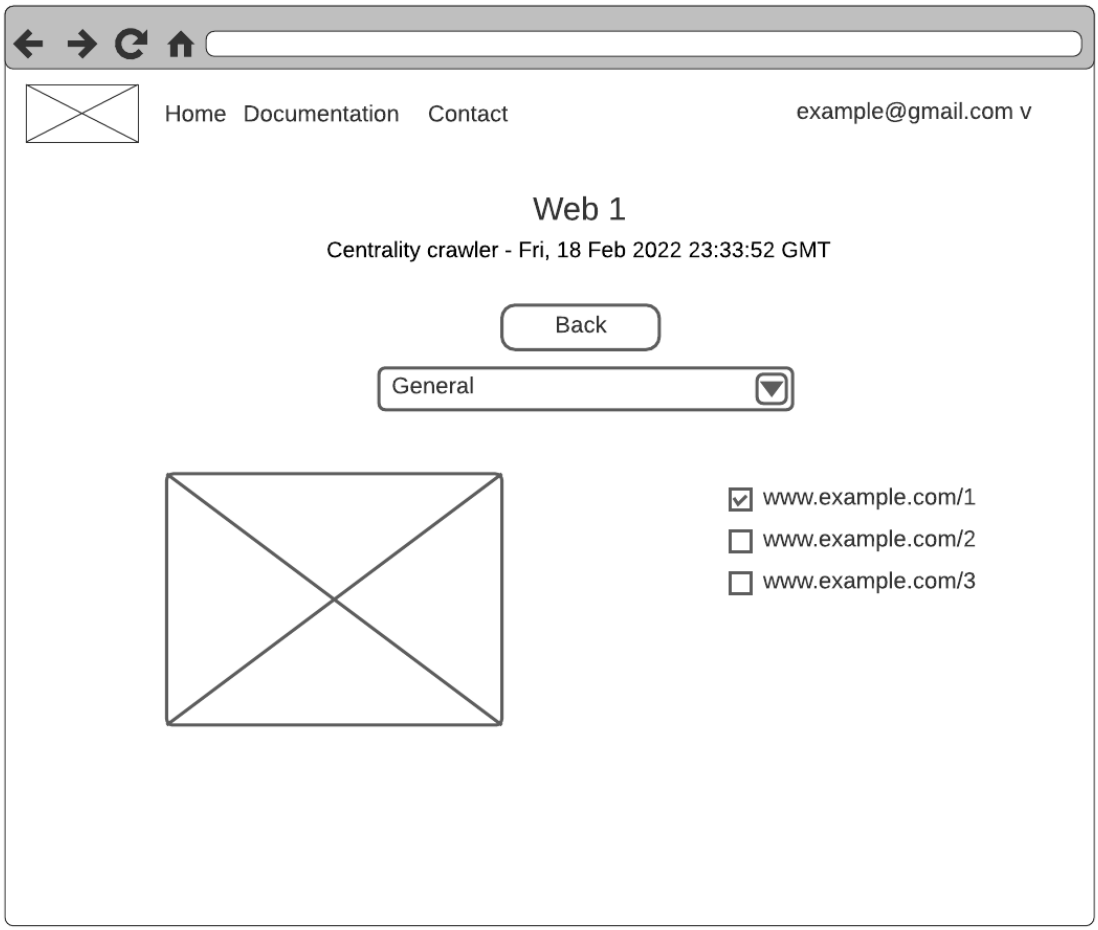
- Tlačítko pro návrat zpět na detail webu;
- Výběrový box pro zvolení typu vypisovaných dat;
- Výpis informací získaných z těžby dat v podobě tabulek s vypočítanými hodnotami centralit pro veškeré stránky webu a upravitelného webového grafu.

Po kliknutí na:

- Tlačítko pro návrat zpět na detail webu systém zobrazí stránku detailu webu;
- Výběrový box systém zobrazí výběr typu vypisovaných dat – obecná data či data pro konkrétní míry centralit. V případě výběru obecných dat systém zobrazí webový graf s označitelnými webovými stránkami v seznamu. V případě jejich označení zahrne dané stránky do webového grafu.

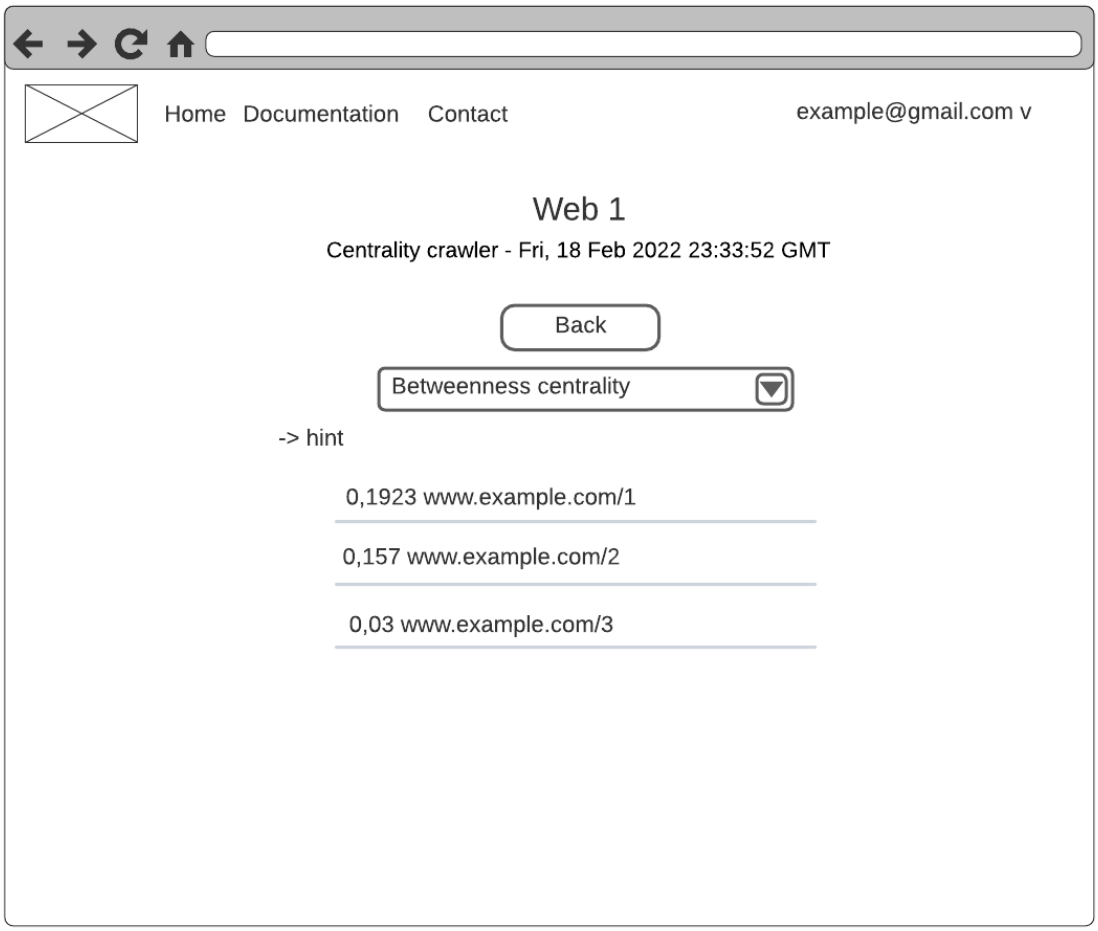
4.6.4.3 Logický design

Obrázek 11: Centrality crawler detail 1 – logický design



Zdroj: (Autor)

Obrázek 12: Centrality crawler detail 2 – logický design



Zdroj: (Autor)

4.7 Implementace

Pro implementaci webové aplikace byly vybrány implementační prostředky popsané v teoretické části práce. Backendová část aplikace byla vytvořena za pomocí frameworku Flask. Skripty obsluhující těžbu dat z webů byly napsány s použitím jazyka Python, a to především za pomoci knihoven Beautiful Soup a Selenium. Komunikace s PostgreSQL databází probíhá s využitím SQLAlchemy.

Frontendová část aplikace byla vytvořena pomocí frameworku Vue.js. Komunikace mezi frontendem a backendem je obsluhována knihovnou Axios. Udržení informace o přihlášených uživatelích je zajištěno pomocí JSON Web Tokens.

Vytvořená aplikace bude v následujících podkapitolách demonstrována na několika jejich částech pomocí ukázek kódu a konečných výstupů. Konkrétní vybrané části jsou

stránka s názvem „detail webu“ a dále všechny tři implementované nástroje pro těžbu dat: Basic crawler, Keywords crawler a Centrality crawler. Veškerý kód aplikace je pak součástí přílohy práce.

4.7.1 Detail webu

Stránka „detail webu“ je určena k zobrazení základních informací o analyzované webové stránce. Uživatel zde má možnost ověřit si, zda je tento web verifikován (tj. bylo prokázáno jeho vlastnictví uživatelem), a popřípadě zajistit jeho verifikaci. V případě verifikovaného webu má uživatel možnost zahájit analýzu webu. Dále může spravovat historické záznamy o těžbách dat a prokliknout se k jejich výstupům. Stránka dále obsahuje odkaz k formuláři pro editaci klíčových slov a možnost smazání veškerých záznamů o webu v aplikaci.

4.7.1.1 Backendová část

Pro zajištění funkcionalit a získání dat pro stránku „detail webu“ je v rámci backendové Flaskové aplikace vyčleněno několik funkcí, které jsou spuštěny při odeslání požadavku na URL specifikované pomocí dekorátoru *@web.routes*. Stránka detail webu je dostupná pouze přihlášeným uživatelům. Proto je před spuštěním každé z těchto funkcí nejprve z JSON Web Tokenu zjištěno, zda je uživatel přihlášen. Tento proces zpracovává funkce *token_required,* díky které jsou dekorované funkce nedostupné nepřihlášeným uživatelům. V případě zjištění neaktivního přihlášení všechny tyto funkce automaticky vrátí stavový kód 401.

Pro ukázku implementace backendové části pro stránky „detail webu“ bylo vybráno pět funkcí. První z nich je funkce zjišťující, zda v rámci daného webu již neprobíhá těžba dat (kód 5). V takovém případě jsou pak některé funkcionality aplikace blokovány. V databázi je v tabulce s názvem „web“ pro tyto účely vyhrazen atribut „active_connection“. Stejně nazvaná funkce nejprve z URL adresy, na kterou byl odeslán požadavek, získá informaci, o jaký web se jedná (proměnná web_id). Pomocí tohoto identifikátoru pak z databáze získá příslušný záznam o webu a zjistí, zda je atribut „active_connection“ nastaven na *true* či *false*. Příslušnou informaci pak vrátí jako svůj výstup společně se stavovým kódem 200.

Kód 5: Metoda active_connection

```
@webs.route('/api/<web_id>/active_connection', methods= ['GET'])
@token_required
def active_connection(current_user,web_id):
    web = Web.query.filter_by(id=web_id).first()
    if web.active_connection == True:
        return 'true', 200
    else:
        return 'false', 200
```

Zdroj: (Autor)

Funkce *web_detail* (kód 6) je určena k získání základních informací o webu z databáze. Tyto informace jsou nejprve serializované pomocí objektu *web_schema* a následně předány jako výstup z funkce.

Kód 6: Metoda web_detail

```
@webs.route('/api/<web_id>/detail', methods= ['GET'])
@token_required
def web_detail(current_user,web_id):

    web = Web.query.filter_by(id =web_id).first()
    web_data = web_schema.dump(web)

    return web_data, 200
```

Zdroj: (Autor)

Funkce *delete_crawl*s (kód 7) je funkce zajišťující smazání vybraných historických záznamů o těžbách dat. Seznam těchto záznamů (crawlů) je obdržen jako součást příslušného HTTP požadavku ve formátu JSON. Následně jsou záznamy za pomoci jejich identifikátorů nalezeny v databázi a postupně mazány. Po skončení *for* cyklu, který tuto službu zastává, je pak provedena metoda *commit* nad objektem spravující připojení k databázi, a tím jsou změny uloženy. Protože se jedná o zpracovávání požadavku typu POST, je navrácen stav 201.

Kód 7: Metoda delete_crawls

```
@webs.route('/api/<web_id>/delete_crawls', methods= ['POST'])
@token_required
def delete_crawls(current_user,web_id):
    data = request.json
    for crawl in data['data']:
        crawl_to_delete = Crawler_data.\
                          query.filter_by(id=crawl['id'])\
                          .first()
        db.session.delete(crawl_to_delete)
    db.session.commit()
    return '', 201
```

Zdroj: (Autor)

V rámci stránky „detail webu“ je zajišťováno i spouštění jednotlivých analýz (crawlů). Na ukázku je zde předvedena funkce zpracovávající požadavek na spuštění základní analýzy webu (basic crawler, kód 8). Před spuštěním samotné těžby dat je nejprve zjištěno, zda již v databázi není pro daný web uloženo padesát záznamů historických těžeb dat. V takovém případě další těžbu nelze spustit do doby, než uživatel některý z předchozích záznamů vymaže.

V případě možnosti spuštění těžby je na příslušném webu nejprve nastaven atribut *active_connection* na hodnotu *true*, čímž je uložena informace o tom, že v rámci webu byla spuštěna analýza. Následně je vytvořeno nové vlákno, ve kterém se pustí samotná funkce obsluhující spuštění příslušného skriptu, a je předána odpověď na HTTP požadavek, který funkci spustil. Spuštění nového vlákna je prováděno z důvodu zajištění nepřesažení délky požadavku 30 sekund, což je jedním z omezení služby Heroku, jak bylo probíráno v teoretické části (viz nefunkční požadavek číslo 4 z kapitoly 4.1.2.4).

Kód 8: Metoda crawl_basic

```
@app.route('/crawl/api/basic_crawl/<web_id>', methods = ['GET'])
@token_required
def crawl_basic(current_user, web_id):
    web = Web.query.filter_by(id =web_id).first()
    crawls = Crawler_data.query.filter_by(web_id = web_id).all()
    if len(crawls) >= 50:
        return 'too many crawls', 200
    web.active_connection = True
    db.session.commit()
    threading.Thread(target=handleCrawl, args=(web, web_id, 1)).start()
    return '', 200
```

Zdroj: (Autor)

Funkce *handleCrawl* (kód 9) z obdrženého parametru type rozezná, o jaký typ analýzy se jedná a spustí příslušný skript. Scrapovací skripty jsou blíže popsány v kapitole 4.2.2. Funkce se zároveň stará i o uložení získaných dat do databáze. Protože v rámci těžebních skriptů je vždy opět kontrolováno, jestli je daný web stále verifikovaný (tedy zda je v kódu jeho stránky umístěný příslušný vygenerovaný HTML tag), je nejprve před uložením dat kontrolováno, zda byl web skutečně analyzován, nebo byl-li skript v počátku ukončen z důvodu nenalezení tagu.

Kód 9: Metoda handleCrawl

```
def handleCrawl(web,web_id, type, words = None):
    with app.app_context():
        if type == 1:
            data = basic_crawler(web.url, web.token)

        if type == 2:
            data = keywords_crawler(web.url, words, web.token)

        if type == 3:
            data = centrality_crawler(web.url, web.token)
            if data != 'not_verified':
                data = handleCrawlData(data)
        web = Web.query.filter_by(id=web_id).first()

        web.active_connection = False
        if data != 'not_verified':
            crawl = Crawler_data(
                data=data,
                date=datetime.now(),
                web_id=web_id,
                type_id=type)

            db.session.add(crawl)
        else:
            web.verified = False
        db.session.commit()
```

Zdroj: (Autor)

4.7.1.2 Frontendová část

Frontendová část stránky „detail webu“ je obsluhována Vue.js komponentou nazvanou WebDetail. Opět byly vybrány části kódu pro demonstraci některých obsluhovaných funkcionalit, a to především ty, které korespondují s popisovanými funkcemi z předchozí kapitoly.

Na ukázce kódu 10 je nejprve za pomocí knihovny Axios odeslán požadavek na backendovou část aplikace pro zjištění, zda v rámci webu neprobíhá aktivní těžba dat. Získaná informace je následně uložena do proměnné *crawl_in_progress*. Tento požadavek je pak vždy odesílán každých 60 000 milisekund znovu spuštěním funkce *crawlInProgressCheck*. Takto je zajištěno pravidelné ověření, zda je možné poskytnout uživateli funkci spuštění analýz. V hlavičce požadavku je zároveň vždy odesílán příslušný JSON Web Token k ověření přihlášení uživatele.

Odesláním dalšího požadavku jsou získána základní data o řešeném webu, která jsou následně uloženy do proměnné web. Z těchto dat je zároveň získána informace, zda je web již verifikovaný.

Kód 10: Ukázka komponenty WebDetail

```
const check = await axios.get(

`${this.API_URL}webs/api/${this.$route.params.webID}/active_connection`,
    { headers: { Authorization: `Bearer: ${jwt}` ,
        "Access-Control-Allow-Origin" : "*"} }, )

this.crawl_in_progress = check.data
this.interval = window.setInterval(() => this.crawlInProgressCheck(),
60000);

const response = await
    axios.get(`${this.API_URL}webs/api/${this.$route.params.webID}/detail`,
    { headers: { Authorization: `Bearer: ${jwt}` ,
        "Access-Control-Allow-Origin" : "*"} }, )

this.web = response.data

if(this.web.verified){
  this.verification_banner = true
}
```

Zdroj: (Autor)

Asynchronní funkce *deleteItems* (kód 11) zajišťuje vymazání vybraných záznamů o historických těžbách dat. Každý záznam vypsaný v uživatelském rozhraní je opatřen checkboxem. Po jeho zaškrtnutí se záznam uloží do pole *checked_items*. Právě to se odesílá jako obsah POST požadavku odesílaným na backend aplikace. Po odeslání požadavku je zavoláním *this.$router.go(0)* znovu načtena celá stránka, a to bez již vymazaných záznamů.

Kód 11: Metoda deleteItems

```
async deleteItems(){
  this.loader_check = true
  const jwt = localStorage.getItem('token')

  await axios

.post(`${this.API_URL}webs/api/${this.$route.params.webID}/delete_crawls`,
      {'data': this.checked_items},
      { headers: { Authorization: `Bearer: ${jwt}` ,
          "Access-Control-Allow-Origin" : "*"} }, )

  this.$router.go(0)
}
```

Zdroj: (Autor)

Výstupní obrazovka je vykreslována pomocí Vue.js šablony. Kód 12 ukazuje její část – konkrétně tabulku záznamů o historických těžeb dat. V její hlavičce se nachází tlačítko, po jehož stisknutí se spustí funkce popsaná v kódu 11 výše. Veškeré informace o záznamech jsou uloženy v proměnné *crawl_data*. Ty jsou z ní vypisovány do řádků tabulky pomocí *foreach* cyklu. Vypsán je vždy název typu analýzy, datum jejího ukončení a dále je uveden odkaz vedoucí na výpis informací získaných z dané analýzy. Každý záznam je také opatřen checkboxem. Ten je vždy defaultně nastaven jako nezaškrtnutý. Pokud dojde k zaškrtnutí checkboxu, respektive odškrtnutí, je tento fakt detekován pomocí události *onchange*. Následně se spustí funkce *checked*, která daný záznam přidá, respektive odebere z pole označených záznamů.

Kód 12: Ukázka tabulky historických záznamů těžeb dat

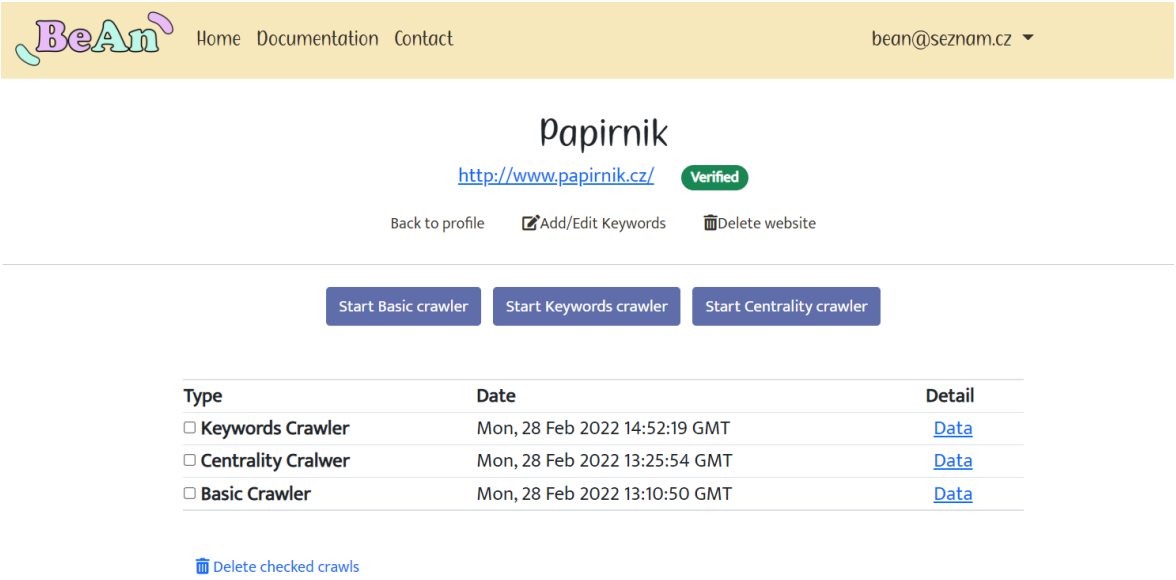
```
<table class="table table-bordered  center"  style="max-width: 70%" >

  <caption>
    <br>
    <button  class="btn btn-link" style="text-decoration: none"
@click="deleteItems">
      <i class="fas fa-trash-alt"></i> Delete checked crawls
    </button>
  </caption>
  <tr>
    <th scope="col">Type</th>
    <th scope="col">Date</th>
    <th scope="col">Detail</th>
  </tr>
  <tr v-for="item in this.crawl_data" :key="item.id">
    <td style=" font-weight: bolder!important;">
      <input placeholder="this.unchecked" type="checkbox"
             v-on:change="checked(item)"
             aria-label="Checkbox for following text input" >
      {{item.type}}
    </td>
    <td>{{item.date}}</td>
    <td><router-link :to="{ name: item.link, params: { crawlID: item.id}}">
      Data
    </router-link></td>
  </tr>
</table>
```

Zdroj: (Autor)

4.7.1.3 Výstupní obrazovka

Na obrázku 13 je ukázána výstupní obrazovka stránky „Detail webu“. Web, pro který byla tato stránka vykreslena, se jmenuje Papírník a je verifikovaný. Zároveň není zobrazena informace o probíhající těžbě dat, a jsou tedy spustitelné všechny tři typy možných analýz.

Obrázek 13: Ukázka výstupní obrazovky Web detail



Zdroj: (Autor)

4.7.2 Skripty pro těžbu dat

V této kapitole jsou popsány základní principy implementovaných skriptů pro samotné sbírání dat analyzovaných webů a jejich vyhodnocování. Skripty jsou trojího typu: skript pro získávání základních informací (v aplikaci nazýván jako Basic crawler), skript pro detekci klíčových slov (Keywords crawler) a skript pro tvorbu webového grafu a výpočet centralit (Centrality crawler).

4.7.2.1 Základní logika

Základní struktura je pro všechny tři skripty stejná. Celá logika se nachází uvnitř *while* cyklu, který je ukončen po prošetření všech podstránek webu. Pro ty jsou vždy vybrány veškeré odkazy, které se na dané stránce nacházejí. Zkontroluje se jejich příslušnost k analyzovanému webu, a pakliže je vyhodnoceno, že vedou na některou z podstránek webu, která ještě nebyla prošetřena, jsou tyto odkazy přidány do pole, ze kterého odebírá položky zmíněný *while* cyklus.

Na kódu 13 je ukázána část tohoto cyklu. Na začátku je pomocí knihovny *requests* odeslán požadavek na analyzovanou podstránku a je nastaveno správné kódování obsahu. Nejedná-li se o HTML typ obsahu, je analýza stránky ukončena. Kód 14 ukazuje otevření stránky na pozadí v simulovaném prohlížeči pomocí knihovny Selenium a Google Chrome driveru. Získaný obsah webu je pak načten do objektu BeautifulSoup ze stejnojmenné

knihovny a jsou v něm nalezeny veškeré odkazy (*findAll('a')*). Tyto odkazy jsou pak funkcí *analyze_link* analyzovány, a je vyhodnoceno, zda budou přidány do pole nazvané *frontier*. Z tohoto pole jsou pak webové stránky postupně odebírány pro celkovou analýzu obsahu.

Kód 13: Ukázka ze skriptu pro těžbu dat

```
source = requests.get(page, verify=False)
source.encoding = source.apparent_encoding
print(source.headers["content-type"])
if "text/html" not in source.headers["content-type"]:
    continue
selenium_source = requests.post(HOST, json={'page': page})

soup = BeautifulSoup(selenium_source.text, 'html.parser')

links = soup.findAll('a', href=True)
for link in links:
    link_type = analyze_link(link['href'], seed)
    if link_type == "outsider" and link['href'] not in outiders:
        outiders.append(link['href'])
    elif (link_type == "full_url" or link_type == "inner") \
            and link['href'] not in inner:
        inner.append(link['href'])
    if link_type == 'full_url':
        frontier.append(link['href'])
    elif link_type == 'inner':
        if link['href'] == '/':
            continue
        if link['href'] != '' and link['href'][0] == '/':
            link['href'] = link['href'][1:]
        frontier.append(domain(seed) + link['href'])
    else:
        continue
```

Zdroj: (Autor)

Kód 14: Metoda selenium

```
@app.route('/selenium', methods = ['POST'])
def selenium():
    page = request.json['page']
    op = webdriver.ChromeOptions()
    op.binary_location = os.environ.get("GOOGLE_CHROME_BIN")
    op.add_argument("--headless")
    op.add_argument("--no-sandbox")
    op.add_argument("--disable-dev-sh-usage")
    driver = webdriver.Chrome(
        executable_path=os.environ.get("CHROMEDRIVER_PATH"),
chrome_options=op
    )
    driver.get(page)
    source = driver.page_source
    driver.close()
    return source
```

Zdroj: (Autor)

Po každém průchodu *while* cyklu je počkáno jednu sekundu před odesláním požadavku na další stránku. Tato funkcionalita je přidána pro splnění nefunkčního požadavku číslo 7. Pro účely nefunkčního požadavku číslo 5 je pak kontrolováno, zda těžba netrvá více než čtyři hodiny.

4.7.2.2 Získávání základních informací

Skript pro získávání základních informací pracuje především s nalézáním příslušných HTML značek ze získaného obsahu stránky. V kódu 15 je ukázka nalezení hlavního nadpisu ve struktuře stránky. Zároveň je pomocí metody *len* zjištěna případná délka jeho textu. Nalezené informace jsou přidány do JSON objektu *page_data* pomocí Python knihovny deepmerge a její metody *merge*.

Kód 15: Ukázka získávání nadpisu ze stránek

```
try:
    main_title = soup.h1.text
except:
    main_title = None

if main_title == None:
    main_title_len = 0
else:
    main_title_len = len(main_title)
page_data = always_merger.merge(page_data, {'h1':
                                            {'value': main_title,
'lenght': main_title_len}
                                        })
```

Zdroj: (Autor)

4.7.2.3 Detekce klíčových slov

Klíčová slova příslušných webů, která jsou definována uživatelem, jsou hledána v obsahu webu v rámci meta tagů *title* a *description* a obecně v textu, názvech obrázků a jejich *alt* popiscích. V kódu 16 je ukázka detekce přítomnosti klíčového slova v rámci obrázků na webových stránkách. Programem jsou procházena jednotlivá klíčová slova uložená v poli s názvem *keywords*, a pro veškeré obrázky nalezené na stránce je zjištěno, zda a kolikrát je slovo obsaženo v jejich názvu a popisku. Samotný název a popisek jsou z objektů představující obrázky získávány pomocí metody *get* jako atributy objektu. Název obrázku je získán konkrétně z atributu source.

Kód 16: Ukázka detekce klíčových slov v obrázcích

```
for k in keywords:
    src_occures = 0
    alt_occures = 0
    for img in images:
        try:
            alt = img.get('alt', '')
            if alt == '':
                alt = img.get('data-alt', '')
            src = img.get('src', '')
            if src == '':
                img.get('data-src', '')
            src_name = src.split('/')[-1]
            src_name = src_name.split('.')[0]
            if src_name != None and k in src_name:
                occures = src_name.count(k)
                src_occures = src_occures + occures
                for w in keywords_data:
                    if k == list(w.keys())[0]:
                        w[k]['img_name'] = w[k]['img_name'] + occures
                        break
            if alt != None and k in alt:
                occures = alt.count(k)
                alt_occures = alt_occures + occures
                for w in keywords_data:
                    if k == list(w.keys())[0]:
                        w[k]['alt'] = w[k]['alt'] + occures
                        break
```

Zdroj: (Autor)

4.7.2.4 Tvorba webového grafu a výpočet centralit

Jednotlivé uzly vytvářeného grafu tvoří odkazy získané dle dříve popsané základní logiky skriptů. Během tohoto úkonu je zároveň v rámci scriptu Centrality crawler také ukládána informace o propojení mezi jednotlivými linky. V kódu 17 je ukázána funkce *graph_cent*, která za pomocí obdržených hran a uzlů sestaví graf funkcí DiGraph z v teoretické části popisované knihovny NetworkX. Následně tato funkce vrací vypočítané hodnoty měr centralit. Konkrétně to jsou hodnoty centrality měřené stupněm uzlu (In-Degree centrality), centrality blízkosti polohy ve středu (Closeness centrality) a centrality středové mezipolohy (Betweenness centrality).

Kód 17: Metoda graph_cent

```
def graph_cent(links, nodes):
    G = nx.DiGraph()
    for n in nodes:
        G.add_node(n['name'])
    for l in links:
        _from = ''
        to = ''
        for x in nodes:
            if l['sid'] == x['id']:
                _from = x['name']
            if l['tid'] == x['id']:
                to = x['name']
            if _from != '' and to != '':
                break
        G.add_edge(_from, to)

        return in_degree_centrality(G), betweenness_centrality(G),
closeness_centrality(G)
```

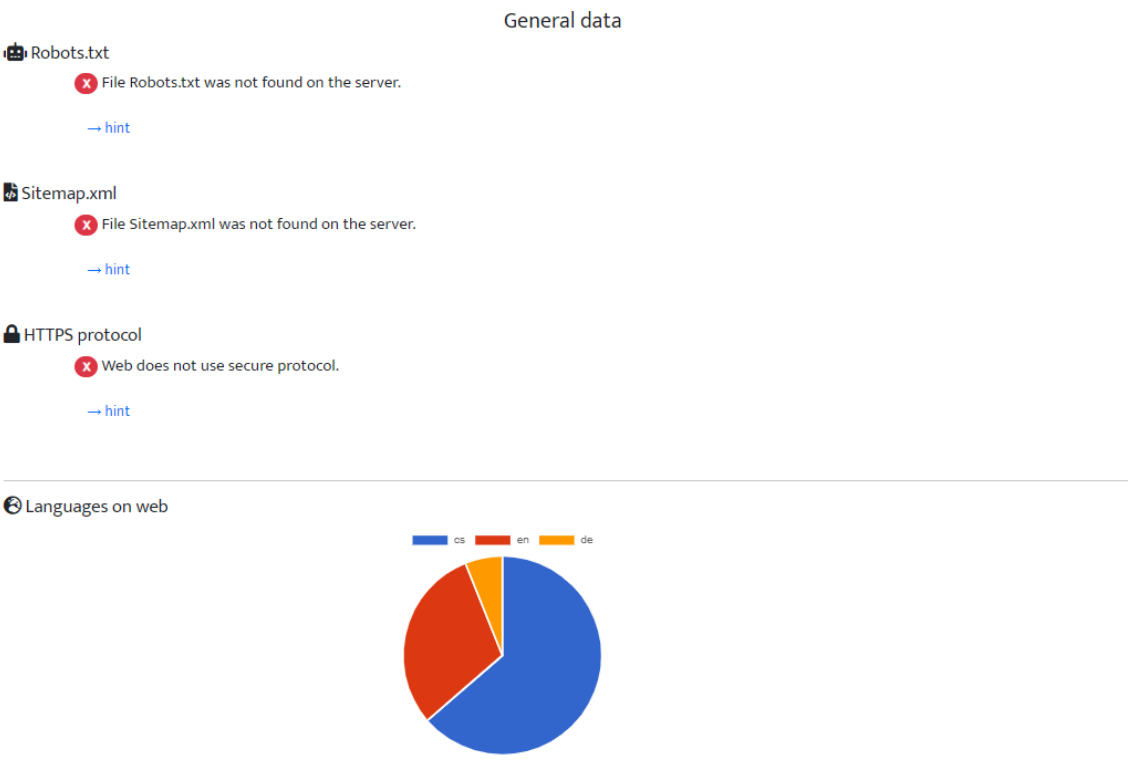
Zdroj: (Autor)

4.7.3 Ukázka výstupních dat webových analýz

Uživatel má možnost si veškerá výstupní data z těžeb jeho webů zobrazit skrze odkazy v tabulkách historických těžeb dat ze stránky „detail webu“. V této kapitole jsou ukázky prezentací dat uživateli.

Obrázek 14 ukazuje část výstupu funkce Basic crawler při výběru zobrazení dat typu „general“. Ve výstupu je patrné, že na analyzovaném webu nebyl nalezen soubor robots.txt ani soubor sitemap.xml, a že web nepoužívá protokol HTTPS. Uživatel si může zobrazit nápovědu týkající se těchto problémů. Zobrazen je i graf rozložení jazyků jednotlivých stránek. Po najetí myší na část grafu se zobrazí konkrétní počet stránek pro daný jazyk. Další nezachycená data v rámci výpisu typu „general“ jsou grafy zobrazující poměr podstránek s uvedenými meta tagy title a description oproti podstránkám, které tyto tagy nemají. Dále je uveden počet vnitřních a externích linků a graf délky odezvy veškerých podstránek.

Obrázek 14: Ukázka výstupní obrazovky Basic crawler 1

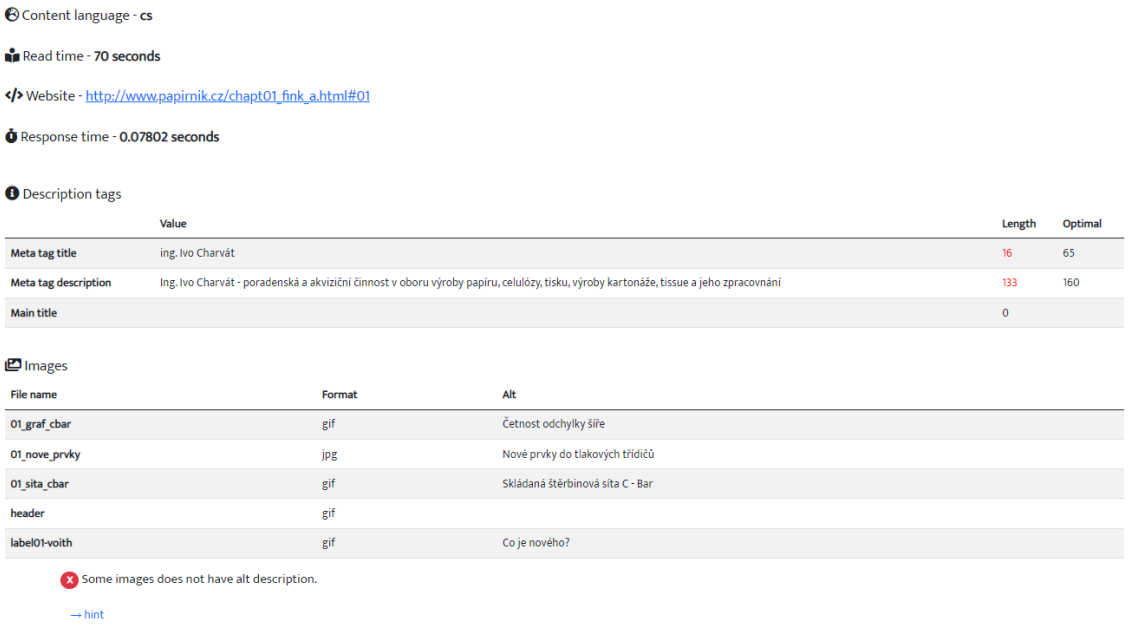


Zdroj: (Autor)

Mimo obecný výstup Basic crawleru je možné zobrazit výstupy pro jednotlivé stránky webu. Na obrázku 15 je takový výstup zobrazen. Je zde uveden jazyk příslušné stránky, doba potřebná na přečtení veškerého obsahu, odkaz na danou stránku, doba odezvy stránky, a dále tabulka ukazující obsah meta tagů *title* a *description* a hlavního nadpisu (jsou-li přítomné). Ve výstupu je uvedená i délka jejich obsahu a pro *title* a *description* je toto číslo zbarvené podle toho, zda odpovídá doporučené délce z hlediska SEO. Doporučená délka je uvedena v posledním sloupci tabulky.

Pokud se na stránce nacházejí obrázky, jsou uvedeny v tabulce s vypsaným názvem obrázku, jeho formátem a případným *alt* popiskem.

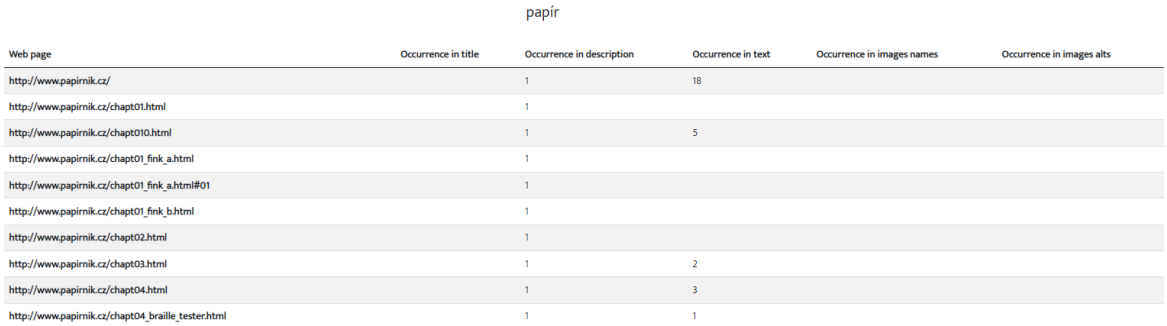
Obrázek 15: Ukázka výstupní obrazovky Basic crawler 2



Zdroj: (Autor)

Ukázka výstupu Keywords crawleru (skriptu pro detekci klíčových slov v obsahu stránek) je na obrázku 16. Zde je vybrán výstup pro klíčové slovo papír, které uživatel definoval před spuštěním příslušného skriptu. V tabulce jsou vypsány veškeré podstránky, na kterých bylo slovo nalezeno, a je uveden i počet jeho obsažení v jednotlivých částech stránky. Tento výstup je možné zobrazit pro veškerá definovaná slova, která měla na webu alespoň jeden nález. Dále je také možné zobrazit obecná data s tabulkou všech definovaných klíčových slov a počtem jejich nálezů.

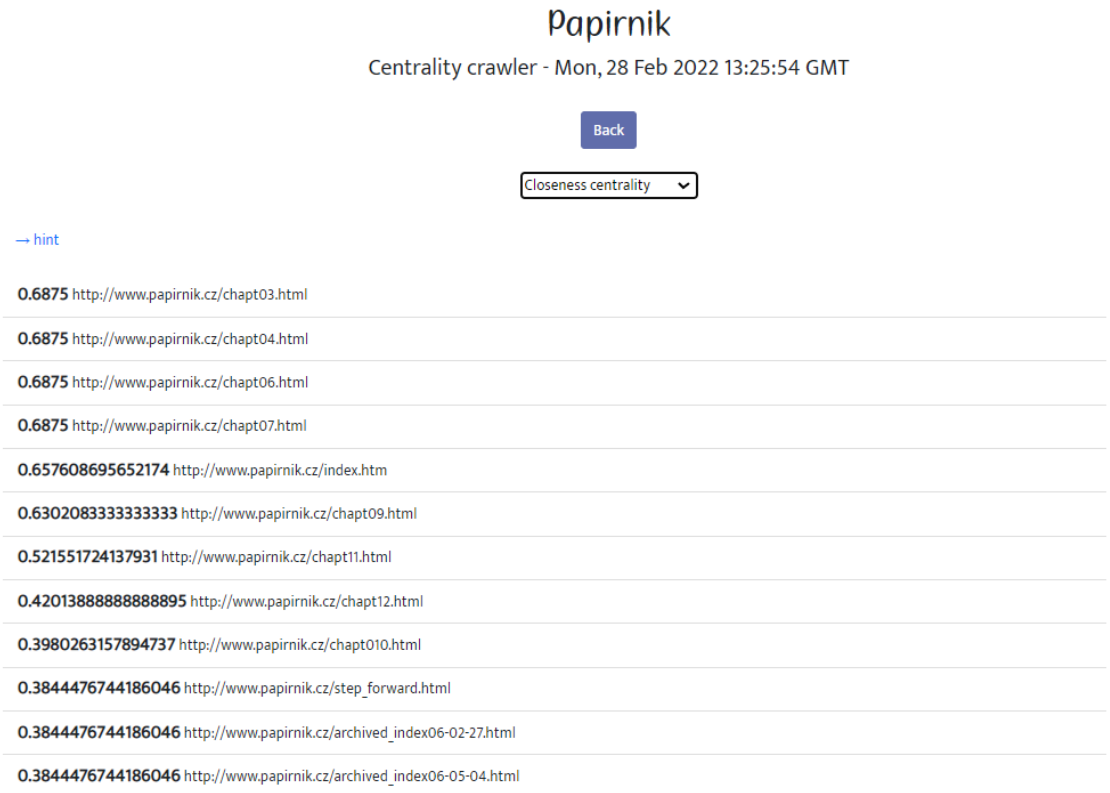
Obrázek 16: Ukázka výstupní obrazovky Keywords crawler



Zdroj: (Autor)

Na obrázcích 17 a 18 jsou výstupy Centrality crawleru (skriptu pro vytvoření webového grafu a výpočet jednotlivých měr centralit. Na obrázku 17 je konkrétně výstup výpočtu měr centralit blízkosti polohy ve středu (Closeness centrality). Na výběr k zobrazení jsou hodnoty všech tří dříve zmiňovaných hodnot centralit. Jednotlivé stránky webu jsou seřazeny dle hodnot centrality od nejvyšší po nejnižší. Možností je také zobrazení nápovědy k daným výstupům.

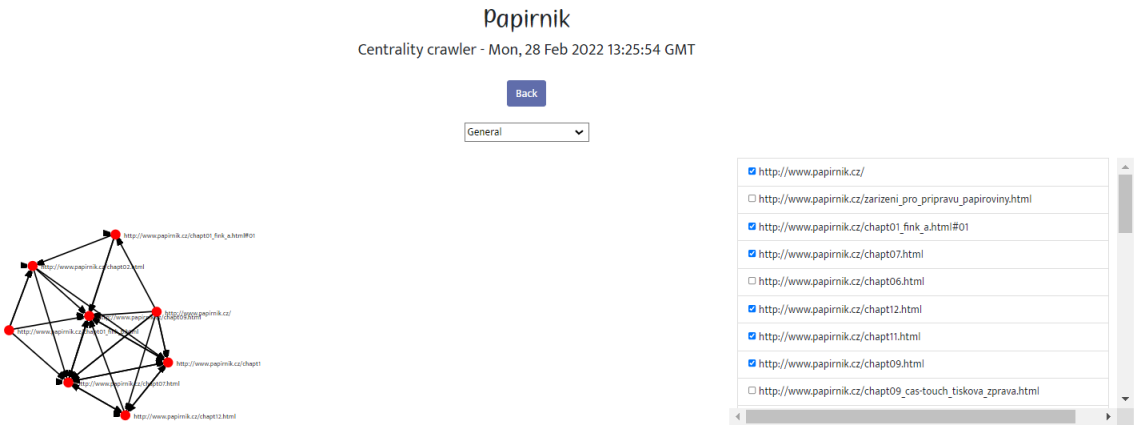
Obrázek 17: Ukázka výstupní obrazovky Centrality crawler 1



Zdroj: (Autor)

Obrázek 18 ukazuje vykreslený webový graf. Tento graf je dynamicky měněn podle označení stránek, které si uživatel přeje do grafu zahrnout. Pro větší přehlednost grafu se dá s jednotlivými uzly posouvat myší a graf tak upravovat do přehlednějšího tvaru. Webový graf se nachází pod výběrem výpisu typu „general“.

Obrázek 18: Ukázka výstupní obrazovky Centrality crawler 2



Zdroj: (Autor)

5 Výsledky a diskuse

5.1 Testování webové aplikace

Webová aplikace byla nasazena pomocí PaaS Heroku a je dostupná na URL adrese https://be-an.herokuapp.com/.

Po jejím zveřejněním bylo provedeno uživatelské testování. Toho se zúčastnili čtyři uživatelé, kteří měli za úkol postupně provést operace dle následujícího scénáře:

- Uživatel se v rámci aplikace zaregistruje a následně přihlásí
- Uživatel přidá ke svému profilu webovou stránku a zajistí její ověření v rámci aplikace
- Uživatel zjistí, kolikrát a v kterých místech se na jeho webové stránce vyskytuje jím vybrané klíčové slovo
- Uživatel ověří základní informace o webové stránce a pokusí se určit, co lze v rámci obsahu webu vylepšit
- Uživatel zobrazí webový graf analyzované webové stránky
- Uživatel zobrazí hodnoty „betweenness centralit“ jednotlivých stránek webu a pokusí se vysvětlit, co dané hodnoty znamenají.

V rámci testování bylo odhaleno, že tlačítka zobrazující nápovědu jsou málo viditelná. Ve výstupech základní analýzy webu je účastníci využili, ve výstupy hodnot centralit webu si tohoto tlačítka však nevšiml ani jeden participant. Zásadní problém tedy činil především poslední úkol ze scénáře, protože účastníkům nebylo zřejmé, že je informace, co konkrétní hodnoty centralit znamenají, vůbec dostupná v rámci aplikace. Zároveň nikoho z participantů nenapadlo využít dokumentace, která je taktéž součástí webové aplikace.

Jako dalším ne příliš intuitivním prvkem uživatelského rozhraní se ukázalo být menu pro výběr typu vypisovaných dat. Toho, že se dají zvolit i další druhy výpisů crawlerů si účastníci testování všimli až při úkolu najít hodnoty měr centralit. U předchozích dvou analýz tohoto prvku vůbec nevyužili.

Ostatní části aplikace se prokázaly jako relativně intuitivní a účastníci s jejich použitím neměli potíže.

5.2 Zhodnocení ze strany autora

SEO a analýza webů jsou v současné době velmi diskutované. Existuje velké množství materiálů zabývající se touto problematikou. Z toho ale zároveň i vyplývá silná konkurence na trhu s aplikacemi, které se webovou analýzou zabývají. Konkurenční výhodou vyvinuté aplikace může být především její dostupnost zdarma a výpočty měr centralit, které nebyly nalezeny jako součástí analýzy u žádného z analyzovaných existujících řešení.

Veškeré využité implementační technologie autorka hodnotí jako vhodné pro účely vyvinuté webové aplikace. Pro všechny technologie byly zároveň dostupné intuitivní dokumentace a návody.

5.2.1 Návrhy na vylepšení

Z uživatelského testování vyplynul problém malé viditelnosti jak prvků zobrazující nápovědy k výstupům z analýz webů, tak i vlastní dokumentace webové aplikace a rozklikávacího menu pro zobrazení dalších výpisů dat z analýzy.

Pro lepší viditelnost prvků nápovědy bude použit znak otazníku, který uživatelům bude více evokovat dostupnost nápovědy k výstupům. U výstupů ukazujících hodnoty měr centralit bude navíc tento symbol umístěn přímo vedle menu pro výběr typu vypisovaných dat. Pro prvky bude také použit větší font písma. V nápovědách budou dále uvedeny odkazy na dokumentaci webu, která bude více propagována i na hlavní stránce webové aplikace.

Menu pro výběr vypisovaných stránek bude celkově zvětšeno a opatřeno popiskem, který poukáže na účel tohoto prvku.

Jak bylo uvedeno v teoretické práci, algoritmy pro hodnocení webových stránek, které využívají webové prohlížeče se stále vyvíjí. Pro budoucí vývoj aplikace je proto důležité udržet aktuálnost analyzovaných oblastí webů.

6 Závěr

Hlavním cílem této práce byla analýza, návrh a implementace webové aplikace pro analýzu webových stránek. Dílčím cílem bylo získání potřebných teoretických poznatků pro samotnou implementaci webové aplikace a popsání oblastí webových stránek, které jsou v rámci aplikace předmětem prováděných analýz.

V teoretické části práce byly splněny dílčí cíle popisem samotné problematiky automatizovaného získávání dat z webů, principů SEO a grafové analýzy webových stránek a dále vymezením vybraných implementačních technologií pro tvorbu webových scraperů, crawlerů a samotných webových aplikací.

V praktické části byl splněn hlavní cíl práce provedením analýzy uživatelských požadavků a existujících řešení a následným navržením aplikace včetně její samotné implementace, která byla demonstrována pomocí popisu ukázek vytvořených kódů. Aplikace byla následně nasazena a uživatelsky otestována a byly vytvořeny návrhy na další úpravy aplikace, týkající se především zvýšení viditelnosti některých důležitých prvků uživatelského rozhraní.

Během testování se aplikace ukázala jako funkční a splňující všechny definované požadavky. Implementované funkcionality umožňují automatizovanou analýzu webových stránek a prezentaci získaných dat uživateli. Zároveň je uživatel upozorněn na případné nedostatky webových stránek z analyzovaných hledisek a výstupy jsou opatřeny nápovědami.

7 Seznam použitých zdrojů

1. What is Web Scraping and How to Use It?. *GeeksForGeeks* [online]. 2021 [cit. 2022-01-20]. Dostupné z: https://www.geeksforgeeks.org/what-is-web-scraping-and-how-to-use-it/
2. Web Scraping 101: 10 Myths that Everyone Should Know. *Octoparse* [online]. 16 August 2021 [cit. 2022-01-20]. Dostupné z: https://www.octoparse.com/blog/10-myths-about-web-scraping
3. JERKOVIC, John I. *SEO warrior*. Sebastopol: O'Reilly, c2010. ISBN 978-0-596-15707-4.
4. SEM. *Shoptet* [online]. [cit. 2022-01-20]. Dostupné z: https://www.shoptet.cz/slovnik-pojmu/sem/
5. Webový vyhledávač. *Google Arts & Culture* [online]. [cit. 2022-01-22]. Dostupné z: https://artsandculture.google.com/entity/search-engine/m02wvbrv
6. THEKKETHIL, Dileep. 52 Search Engines: Best Google Alternatives for 2022. *Stan Ventures* [online]. [cit. 2022-01-22]. Dostupné z: https://www.stanventures.com/blog/top-search-engines-list/
7. LUKÁŠ, Václavík. Vyhledávač Googlu je v Česku třikrát používanější než Seznam. *Cnews* [online]. 17. 2. 2020 [cit. 2022-01-22]. Dostupné z: https://www.cnews.cz/google-seznam-vyhledavac-2014-2019-podil
8. Google PageRank - proč na něm pořád záleží. *SEO prakticky* [online]. [cit. 2022-01-22]. Dostupné z: https://www.seoprakticky.cz/co-je/google-pagerank/
9. Google Search Central documentation [online]. Dostupné z: https://developers.google.com/search/docs
10. VELIČKA, Matěj, Richard KLAČKO a David BRENNER. *Ochutnejte technické SEO* [online]. [cit. 2022-01-24]. Dostupné z: https://taste.cz/ke-stazeni/taste-ochutnejte-technicke-seo.pdf
11. GAVRIL, Alexandra. A beginner’s guide to writing title tags and meta descriptions that get clicks. *123Reg* [online]. 6 February 2017 [cit. 2022-01-24]. Dostupné z: https://www.123-reg.co.uk/blog/seo-2/a-beginners-guide-to-writing-title-tags-and-meta-descriptions-that-get-clicks/

12. HLADIŠ, Karel. Https://www.collabim.cz/akademie/knihovna/klicova-slova-seo/. *Collabim* [online]. 4. 10. 2017 [cit. 2022-01-24]. Dostupné z: https://www.collabim.cz/akademie/knihovna/klicova-slova-seo/
13. *Marketing Miner* [online]. [cit. 2022-01-26]. Dostupné z: https://www.marketingminer.com/
14. ON-PAGE SEO: The Definitive Guide. *Backlinko* [online]. [cit. 2022-01-26]. Dostupné z: https://backlinko.com/on-page-seo
15. JARTYM, Pavel. Rozdíl mezi SEO copywritingem a klasickým copywritingem. *Aira bloguje* [online]. [cit. 2022-01-26]. Dostupné z: https://blog.aira.cz/rozdil-mezi-seo-copywritingem-klasickym-copywritingem
16. HILL, Martin. Co je soubor robots.txt a k čemu slouží v SEO?. *Bridge - ecommerce magazine* [online]. [cit. 2022-01-27]. Dostupné z: https://www.ecommercebridge.cz/co-je-soubor-robots-txt-a-k-cemu-slouzi-v-seo/
17. LAWSON, Richard. *Web scraping with Python: scrape data from any website with the power of Python*. Birmingham: Packt Publishing, 2015. Community experience distilled. ISBN 978-1-78216-436-4.
18. ŠPOLJARIČ, Damir. Co může zpomalovat načítání webových stránek a e-shopů?. *VShosting* [online]. [cit. 2022-01-27]. Dostupné z: https://vshosting.cz/blog/co-muze-zpomalovat-nacitani-webovych-stranek-a-e-shopu
19. STOX, Patrick. JavaScript SEO: What You Need to Know. *Ahrefs* [online]. 29 January 2021 [cit. 2022-01-27]. Dostupné z: https://ahrefs.com/blog/javascript-seo/
20. JavaScript. *Collabim* [online]. [cit. 2022-01-27]. Dostupné z: https://www.collabim.cz/akademie/seofaktory/javascript/
21. Co jsou off-page faktory. *Mioweb* [online]. [cit. 2022-01-27]. Dostupné z: https://www.mioweb.cz/slovnicek/offpage-faktory/
22. HOLLINGSWORTH, Sam. 12 Completely Outdated SEO Practices You Should Avoid. *Search Engine Journal* [online]. 6 October 2021 [cit. 2022-01-29]. Dostupné z: https://www.searchenginejournal.com/outdated-seo-practices/276343/
23. GRIFFTH, Tommy. Meta Keywords: Why You Shouldn't Use Them in 2022 (& Beyond). *ClickMinded* [online]. [cit. 2022-01-29]. Dostupné z: https://www.clickminded.com/meta-keywords/

24. JOHNSON, Desiree. Black Hat SEO Techniques That Can Destroy Your Search Ranking. *Bluehost* [online]. [cit. 2022-01-29]. Dostupné z: https://www.bluehost.com/resources/black-hat-seo-techniques-that-can-destroy-your-search-ranking/
25. NEMIROVSKY, Danil. *Web Graph and PageRank algorithm*. St Petersburg, 2006. St. Petersburg State University.
26. BHASIN, Jatin. Graph Analytics — Introduction and Concepts of Centrality. *Towards Data Science* [online]. [cit. 2022-02-02]. Dostupné z: https://towardsdatascience.com/graph-analytics-introduction-and-concepts-of-centrality-8f5543b55de3
27. JAGANATHAN, B a Kalyani DESIKAN. *Enhanced Web Page Ranking Method Using Laplacian Centrality*. 2018, **7**(4.10), 566-569. ISSN 2227-524X. Dostupné z: doi:10.14419/ijet.v7i4.10.21282
28. Python documentation [online]. Dostupné z: https://docs.python.org/3/
29. Applications for Python. *Python* [online]. [cit. 2022-02-05]. Dostupné z: https://www.python.org/about/apps/
30. *Requests: HTTP for Humans™* [online]. [cit. 2022-02-05]. Dostupné z: https://docs.python-requests.org/
31. BEAZLEY, David M. a Brian K. JONES. *Python cookbook :*. 3nd ed. Beijing: O´Reilly, 2013. ISBN 9781449340377.
32. Langdetect. *PyPI* [online]. [cit. 2022-02-07]. Dostupné z: https://pypi.org/project/langdetect/
33. Readtime. *PyPI* [online]. [cit. 2022-02-07]. Dostupné z: https://pypi.org/project/readtime/
34. Beautiful Soup Documentation [online]. Dostupné z: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
35. SAHIN, Kevin. Web Scraping using Selenium and Python. *ScrapingBee* [online]. 8 July 2021 [cit. 2022-02-07]. Dostupné z: https://www.scrapingbee.com/blog/selenium-python/
36. *NetworkX: Newtork Analysis in Python* [online]. [cit. 2022-02-09]. Dostupné z: https://networkx.org/

37. What is Flask Python. *Python Basics* [online]. [cit. 2022-02-10]. Dostupné z: https://pythonbasics.org/what-is-flask-python/
38. ŠTRAUCH, Adam. Python a Apache: hosting bezpečně přes WSGI. *Root.cz* [online]. 15. 4. 2009 [cit. 2022-02-10]. Dostupné z: https://www.root.cz/clanky/python-a-apache-hosting-bezpecne-pres-wsgi/
39. GRINBERG, Miguel. *Flask Web Development: Developing web applications with Python*. Sebastopol: O'Reilly, 2014. ISBN 9781449372620.
40. KOĎOUSKOVÁ, Barbora. Vue JS: Výhody, Nevýhody a možnosti využití. *Rascasone* [online]. 15. 7. 2021 [cit. 2022-02-10]. Dostupné z: https://www.rascasone.com/cs/blog/co-je-framework-vuejs
41. BARÁŠEK, Jan. Vue.js český manuál a návody. *Jan Barášek* [online]. [cit. 2022-02-10]. Dostupné z: https://vue.baraja.cz/
42. RUSSELL, Richard. #100DaysOfCode Day 33: Connecting Front-End to Back-End using Axios, An Inquiry on Ports. *Cold Brew Code* [online]. [cit. 2022-02-10]. Dostupné z: https://medium.com/cold-brew-code/100daysofcode-day-33-connecting-front-end-to-back-end-using-axios-an-inquiry-on-ports-4712947fa46c
43. JSON Web Tokens. *Zooom* [online]. [cit. 2022-02-10]. Dostupné z: https://zooom.cz/json-web-tokens-jwt/
44. PyJWT. *Read the Docs* [online]. [cit. 2022-02-21]. Dostupné z: https://pyjwt.readthedocs.io/
45. První seznámení s Dockerem. *B2A Software develompent* [online]. [cit. 2022-02-13]. Dostupné z: https://b2a.cz/2021/02/18/seznameni-s-docker/
46. STONEMAN, Elton. *Learn Docker in a month of lunches*. Shelter Island: Manning, [2020]. ISBN 978-1-61729-705-2.
47. *Heroku Dev Center* [online]. [cit. 2022-03-03]. Dostupné z: https://devcenter.heroku.com/
48. TUCKER, John. Why Docker on Heroku?. *Codeburst* [online]. 2 February 2019 [cit. 2022-03-03]. Dostupné z: https://codeburst.io/why-docker-on-heroku-92a0ec392ce5
49. *SEMRush* [online]. [cit. 2022-03-10]. Dostupné z: https://www.semrush.com/

8 Přílohy

Příloha A CD se zdrojovým kódem aplikace

Příloha B Kompletní UI specifikace

Příloha A – CD se zdrojovým kódem aplikace

Analyzer-app.zip

Příloha B – Kompletní UI specifikace

UI specifikace

Motivace a cíle

Tato UI specifikace vznikla pro účely vývoje webové aplikace sloužící pro analýzu webových stránek, nazvanou jako BEAN. Cílem je navrhnout co nejvíce uživatelsky přívětivé prostředí, které bude umožňovat snadný přístup ke všem definovaným funkcionalitám webu.

Persony

Primární persona

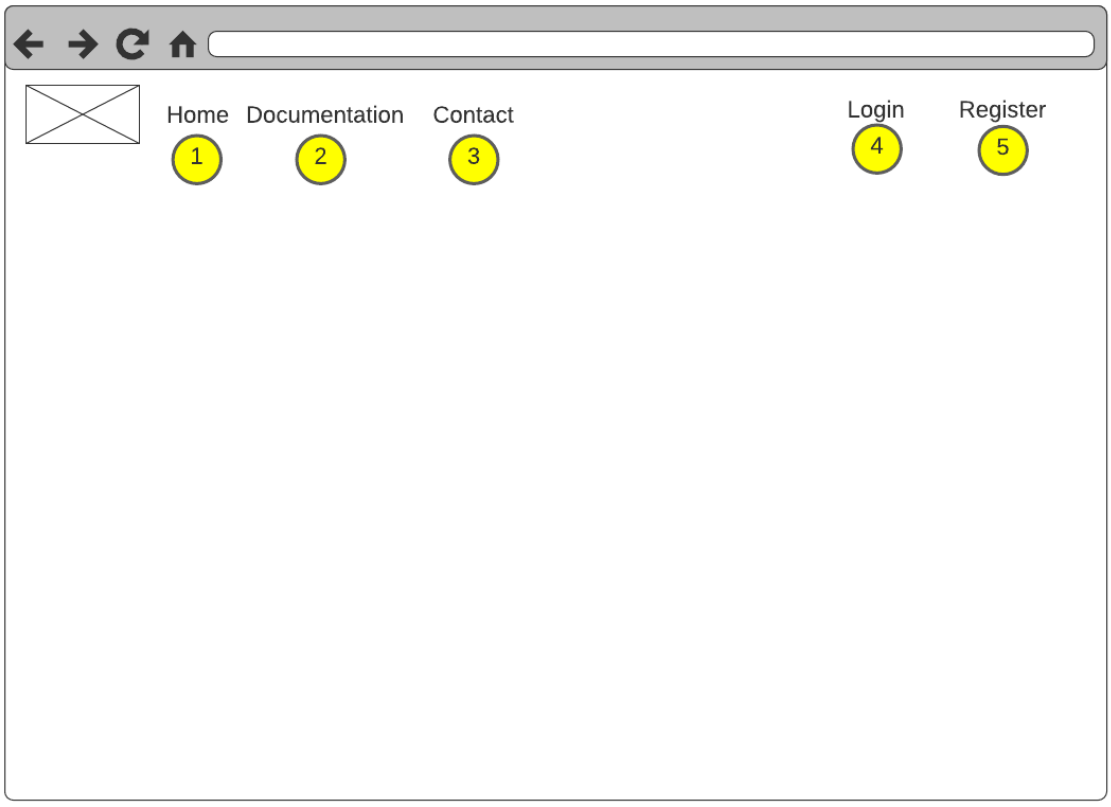
- Jméno: Zuzana Marketingová
- Pohlaví: Žena
- Věk: 33 let
- Koníčky: Hudba, práce
- Historie:
    - Zuzana se narodila v Pelhřimově kde studovala základní i střední školu. Po úspěšné maturitě se přestěhovala do Prahy, kde začala na vysoké škole studovat obor zaměřený na online marketing. Po studiích se vdala a začala podnikat ve sféře vytváření e-shopů na zakázku
- Typický den
    - Zuzana vstane v 8 ráno, nasnídá se a přesune se do pracovny ve svém bytě. Zde pak většinu dne tráví vyřizováním pracovních e-mailů a spravováním obsahu e-shopů klientů. Večer po práci často vyráží s přáteli na koncerty
- Cíl
    - Zuzana by ráda využívala nástroj, který by ji zdarma umožnil automaticky analyzovat obsahy ji spravovaných e-shopů. Ráda by snadno detekovala případné problémy týkající se SEO webu či sledovala umístění klíčových slov na stránkách.

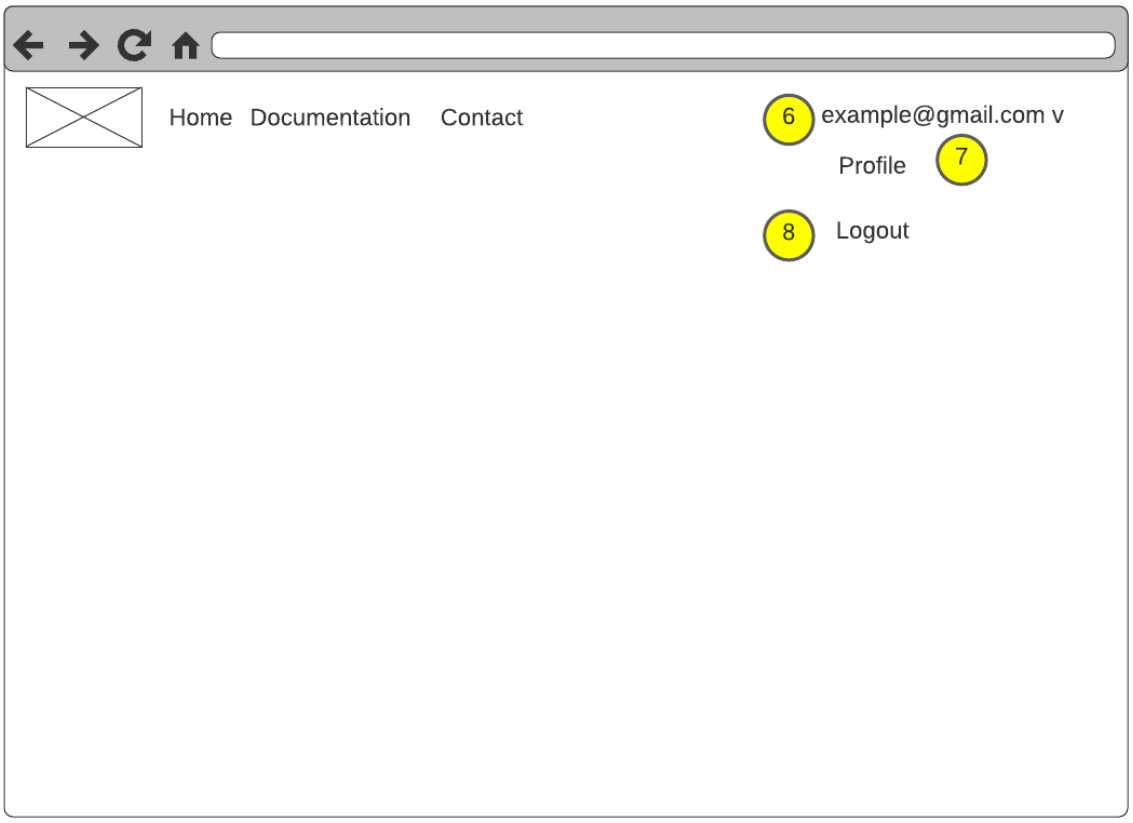
Sekundární persona

- Jméno: Jakub Dřevěný
- Pohlaví: Muž
- Věk: 55 let

- Koníčky: vyřezávání, sledování televize
- Historie
    - Jakub pochází z Tábora, kde v dětství navštěvoval skautský oddíl. Zde jej vedoucí naučili vyřezávat hračky ze dřeva, a Jakub se do této činnosti zamiloval. Vyučil se řezbářem a vydělává si prodejem vlastnoručně vyráběných dřevěných hraček. Původně je prodával převážně v kamenných obchodech a na trzích, před 5 lety si ale k tomu i založil e-shop
- Typický den
    - Jakub tráví většinu času ve svém rodinném domku u Tábora, kde se věnuje vyřezávání hraček, jejich prodeji a péčí o zahradu. Večer často s přáteli sledují v nedaleké restauraci fotbalová utkání
- Cíl
    - Jakub vlastní e-shop a má spíše nižší povědomí o tom, jak zajistit jeho vyšší návštěvnost. Ocenil by tedy přehledný web, který by mu ukázal případné nedostatky jeho e-shopu a pomohl mu je napravit.

Obecná pravidla





1. Home – záložka v menu vedoucí na úvodní informace webové aplikace
2. Documentation – záložka v menu vedoucí na dokumentaci aplikace
3. Contact – záložka v menu vedoucí ke kontaktům na autorku
4. Login – záložka v menu vedoucí k přihlašovacímu formuláři
5. Register – záložka v menu vedoucí na registrační formulář
6. Email přihlášeného uživatele – signalizace přihlášení uživatele, slouží zároveň jako rozklikávací menu
7. Profile – záložka v menu vedoucí na profil uživatele
8. Logout – Odkaz způsobující odhlášení uživatele

Úvod

Use case

Uživatel očekává

- Nalezení základních informací o webové aplikaci.

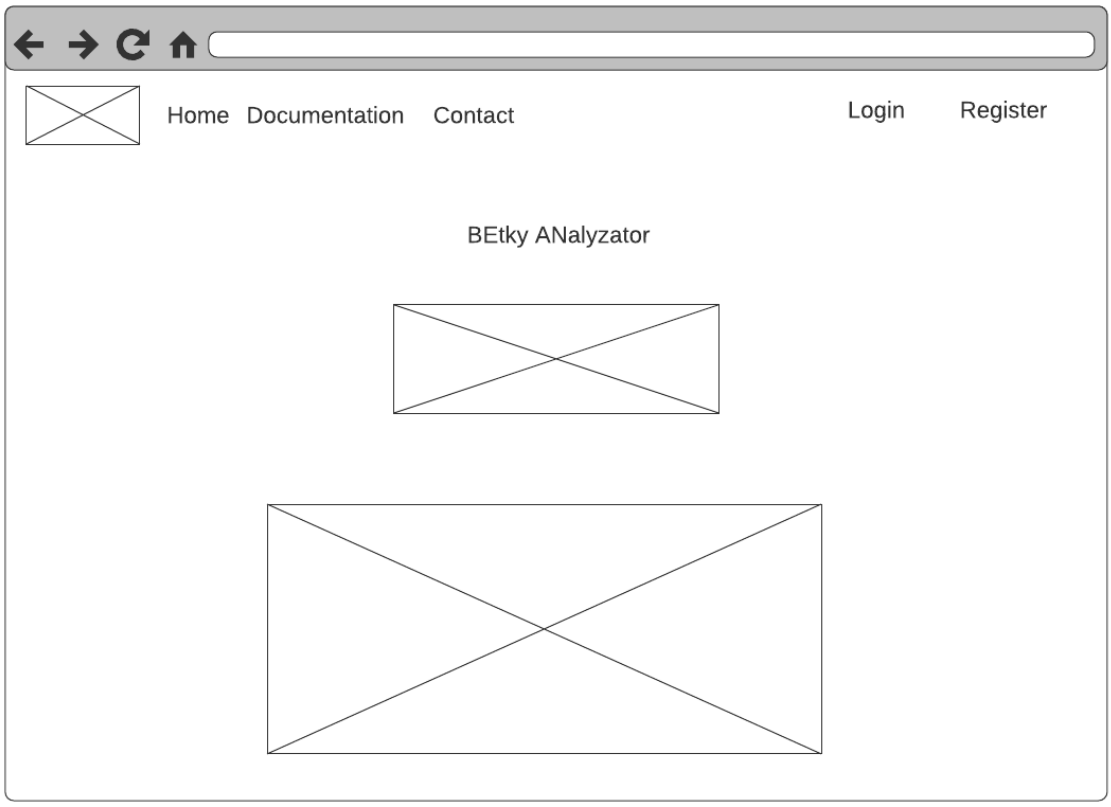
Scénář

Systém zobrazí

- Název aplikace;

- Výčet základních funkcionalit aplikace.

Logický design



Dokumentace

Use case

Uživatel očekává

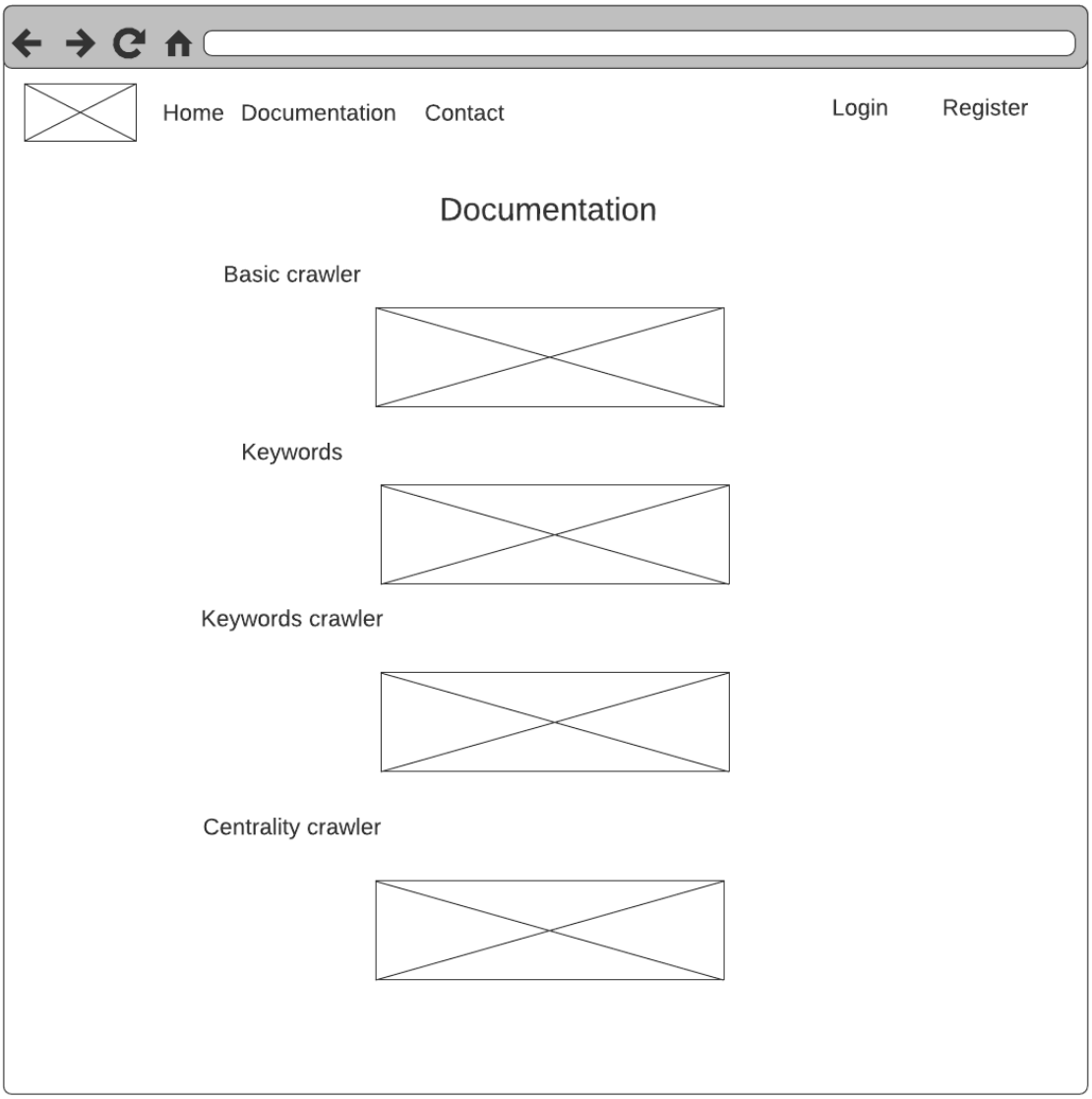
- Uživatel očekává návod k jednotlivým částem webu.

Scénář

Systém zobrazí

- Informace k výstupům funkce Basic crawler;
- Informace k výstupům funkce Centrality crawler;
- Informace k výstupům funkce Keywords crawler;
- Návod k práci s funkcí přidávání a odebírání klíčových slov.

Logický design



Kontakt

Use case

Uživatel očekává

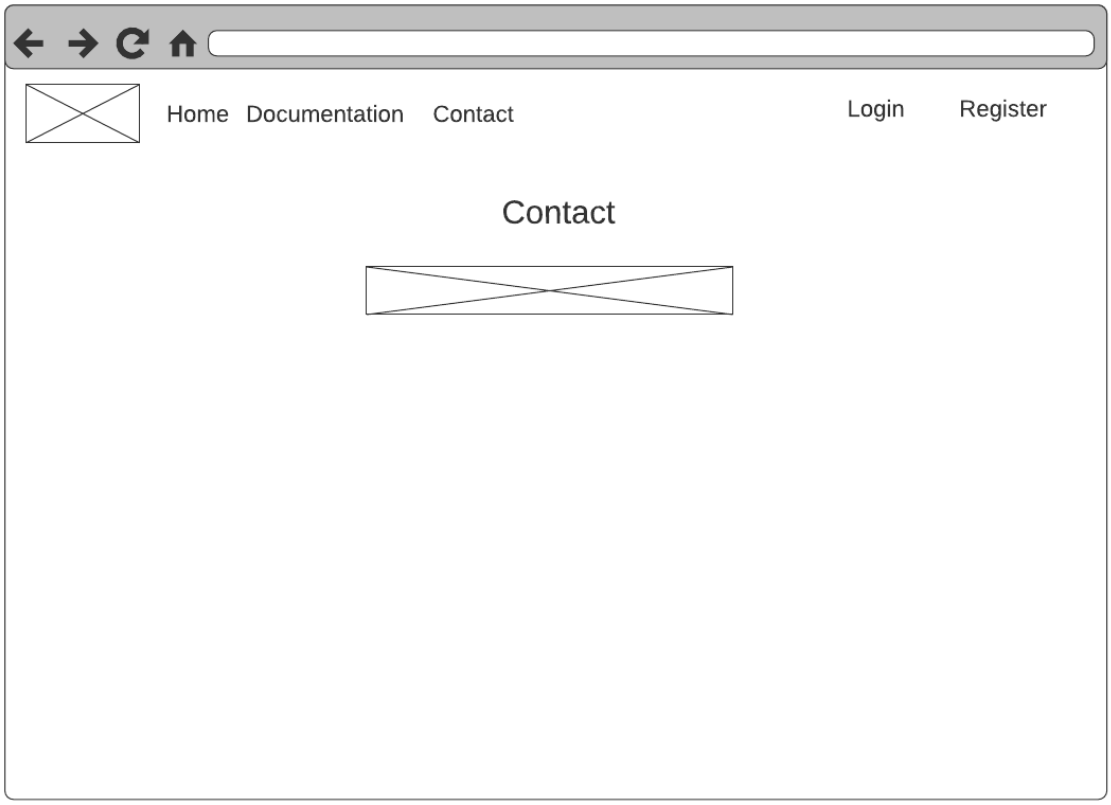
- Nalezení kontaktu na autorku aplikace.

Scénář

Systém zobrazí

- Kontakt na autorku práce.

Logický design



Přihlášení

Use case

Uživatel očekává

- Možnost zadat e-mail pro přihlášení;
- Možnost zadat heslo ke svému uživatelskému účtu;
- Možnost odeslat zadané přihlašovací údaje a přihlásit se tak ke svému účtu;
- Možnost přejít k registraci v případě nevlastnění uživatelského účtu;
- Možnost obnovy zapomenutého hesla;
- Chybové hlášky v případě špatně zadaných přihlašovacích údajů.

Scénář

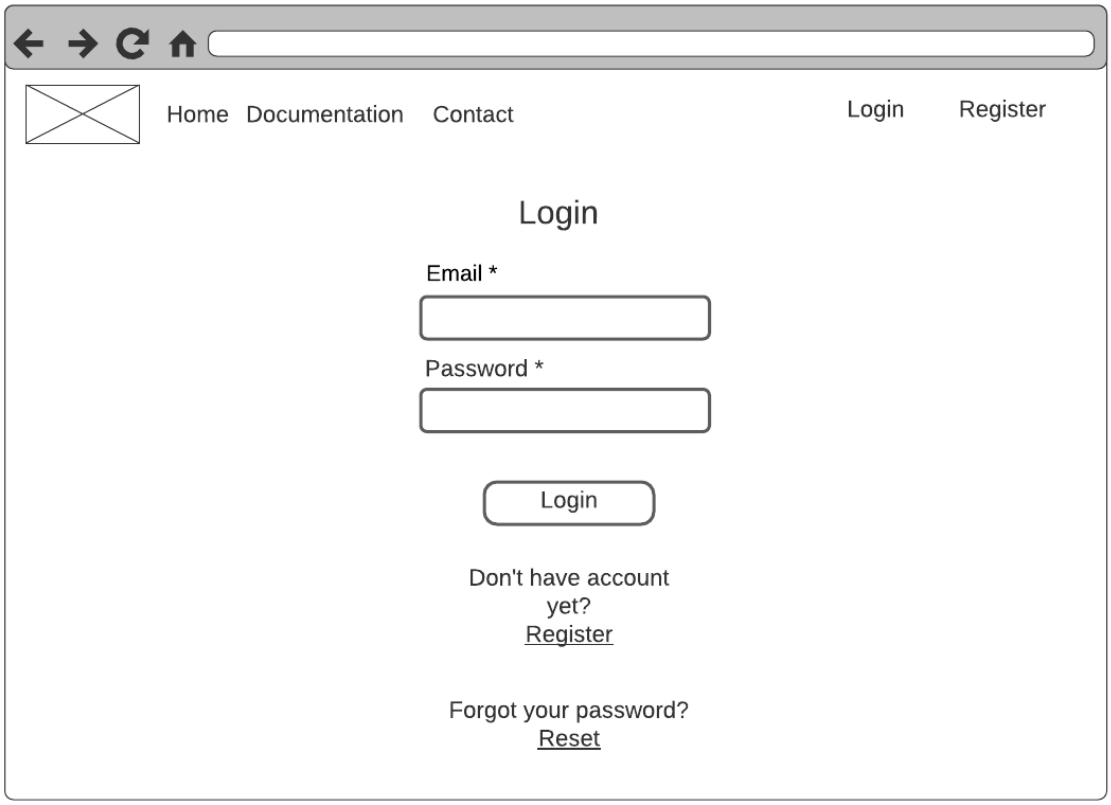
Systém zobrazí

- Pole pro zadání e-mailu;
- Pole pro zadání hesla;
- Tlačítko „Login“ pro provedení přihlášení;
- Odkaz na registrační formulář;
- Odkaz na formulář pro resetování hesla.

Po kliknutí na

- Tlačítko „Login“ v případě správně zadaných údajů systém provede přihlášení, v případě chybně zadaných údajů zobrazí chybovou hlášku;
- Odkaz k registraci systém převede uživatele na registrační formulář;
- Odkaz k resetování hesla systém převede na formulář k vyřízení resetování hesla

Logický design



Registrace

Use case

Uživatel očekává

- Možnost zadat e-mail;
- Možnost zadat křestní jméno;
- Možnost zadat příjmení;
- Možnost zadat heslo;
- Možnost potvrzení hesla;
- Možnost odeslat zadané údaje

- Upozornění na případné chyby ve vyplněném formuláři;
- Možnost rychlého přesměrování k přihlášení v případě existujícího uživatelského účtu

Scénář

Systém zobrazí

- Pole pro zadání e-mailu;
- Pole pro zadání křestního jména;
- Pole pro zadání příjmení;
- Pole pro zadání hesla;
- Pole pro zadání kontroly hesla;
- Tlačítko pro odeslání;
- Odkaz k přesměrování k registraci v případě vlastnění uživatelského účtu

Po kliknutí na

- Tlačítko „odeslat“ systém vytvoří uživatelský účet a upozorní uživatele na nutnost ověření e-mailové adresy skrze zaslaný odkaz a zároveň umožní znovu odeslání tohoto e-mailu. V případě chybných údajů systém zobrazí chybové hlášky a umožní opětovné zadaní.

Logický design

Profil uživatele

Use case

Uživatel očekává

- Zobrazení základních údajů o svém účtu;
- Seznam přiřazených webů k účtu;
- Možnost přidat nový web;
- Možnost zobrazit detailní informace o webech.

Scénář

Systém zobrazí

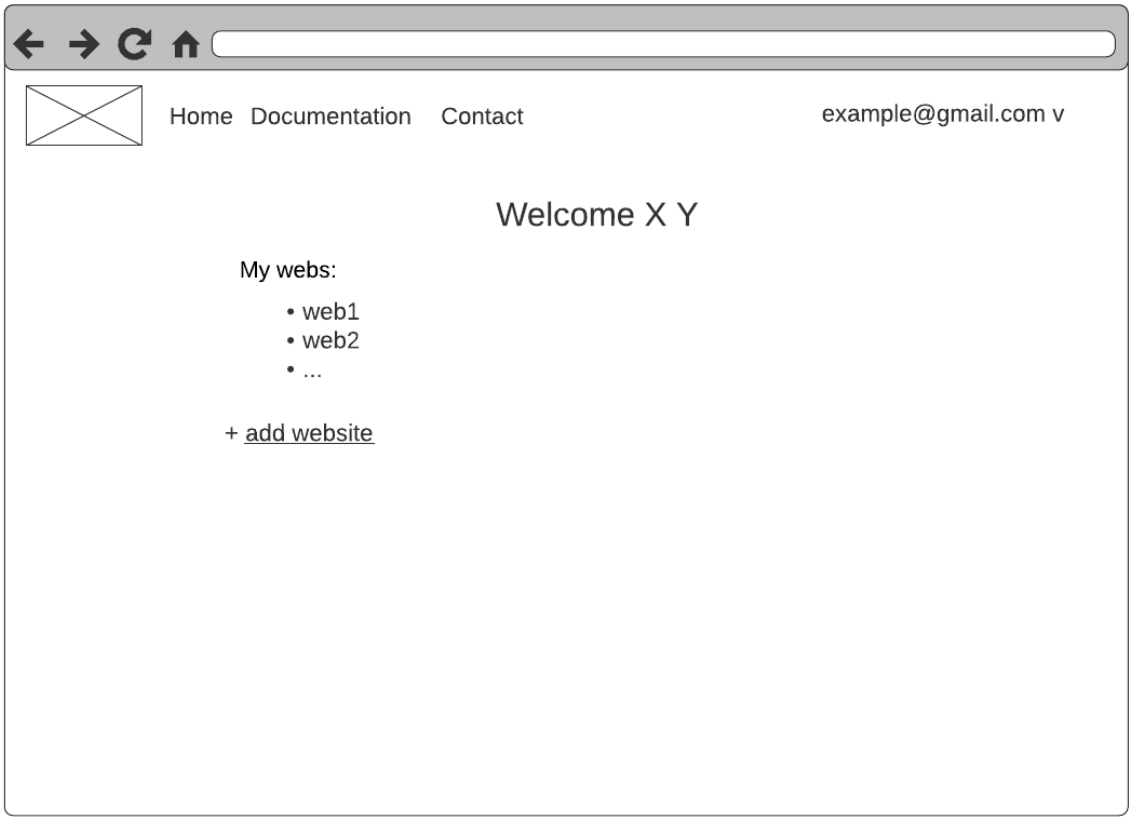
- Jméno a příjmení uživatele;
- Seznam odkazů na detaily webů označené příslušnými jmény webů;

- Tlačítko k přidaní nového webu, pokud již uživatel nemá pod svým profilem vedeno deset webů. V opačném případě zobrazí informaci o této skutečnosti

Po kliknutí na tlačítko

- Přidání webu systém přesměruje uživatele na formulář přidání nového webu.

Logický design



Přidání webu

Use case

Uživatel očekává

- Možnost zadat název nového webu;
- Možnost zadat URL daného nového webu;
- Získání informaci o případném chybném vyplnění formuláře;
- Možnost odeslání formuláře;
- Možnost návratu zpět na profil.

Scénář

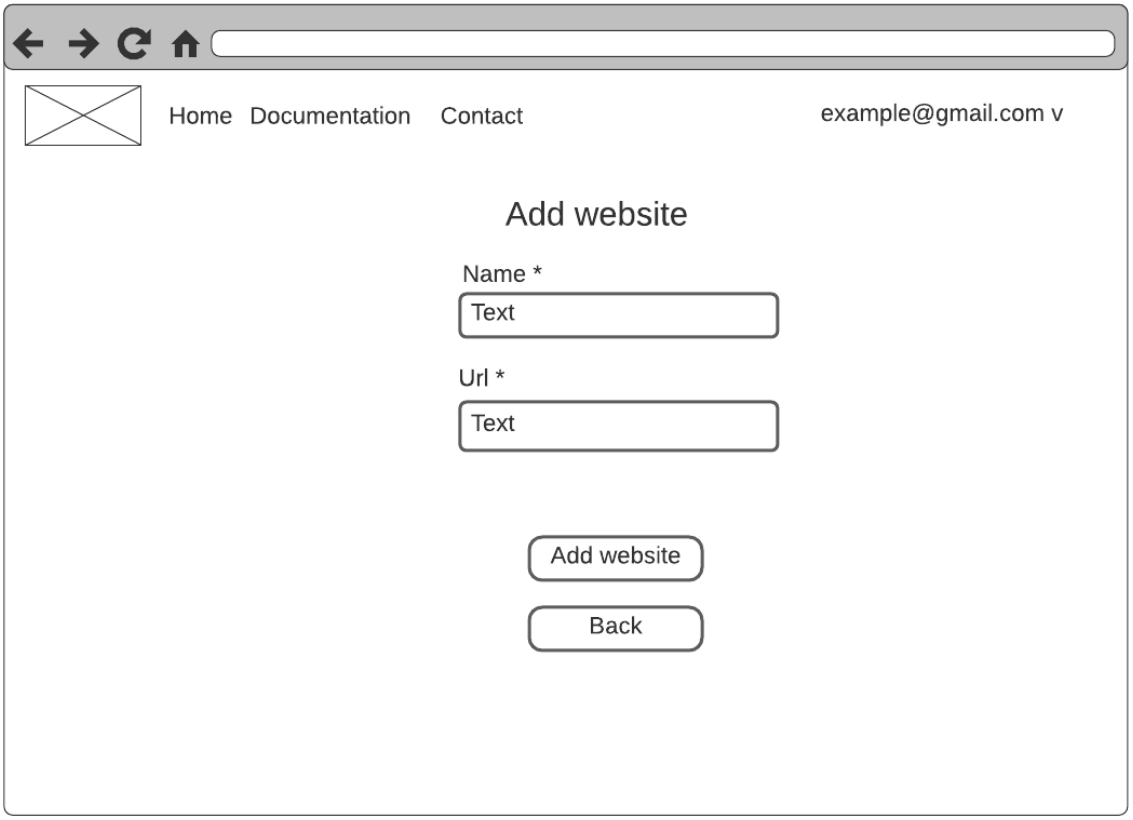
Systém zobrazí

- Textové pole pro zadání názvu stránky;
- Textové pole pro zadání URL stránky;
- Tlačítko pro uložení změn;
- Tlačítko pro návrat zpět na profil uživatele.

Po kliknutí na

- Tlačítko „přidat“ v případě správně zadaných údajů systém uloží údaje a přesune uživatele na stránku „verifikace webu“. V případě chybně zadaných údajů vypíše chybovou hlášku.
- Tlačítko pro návrat na profil systém zobrazí profil uživatele.

Logický design



Verifikace webu

Use case

Uživatel očekává

- Možnost získat HTML značku pro vložení do kódu stránky;
- Možnost tuto značku rychle zkopírovat;
- Možnost provést verifikaci;
- Možnost vrátit se zpět na profil;
- Informaci o případné neúspěšné verifikaci.

Scénář

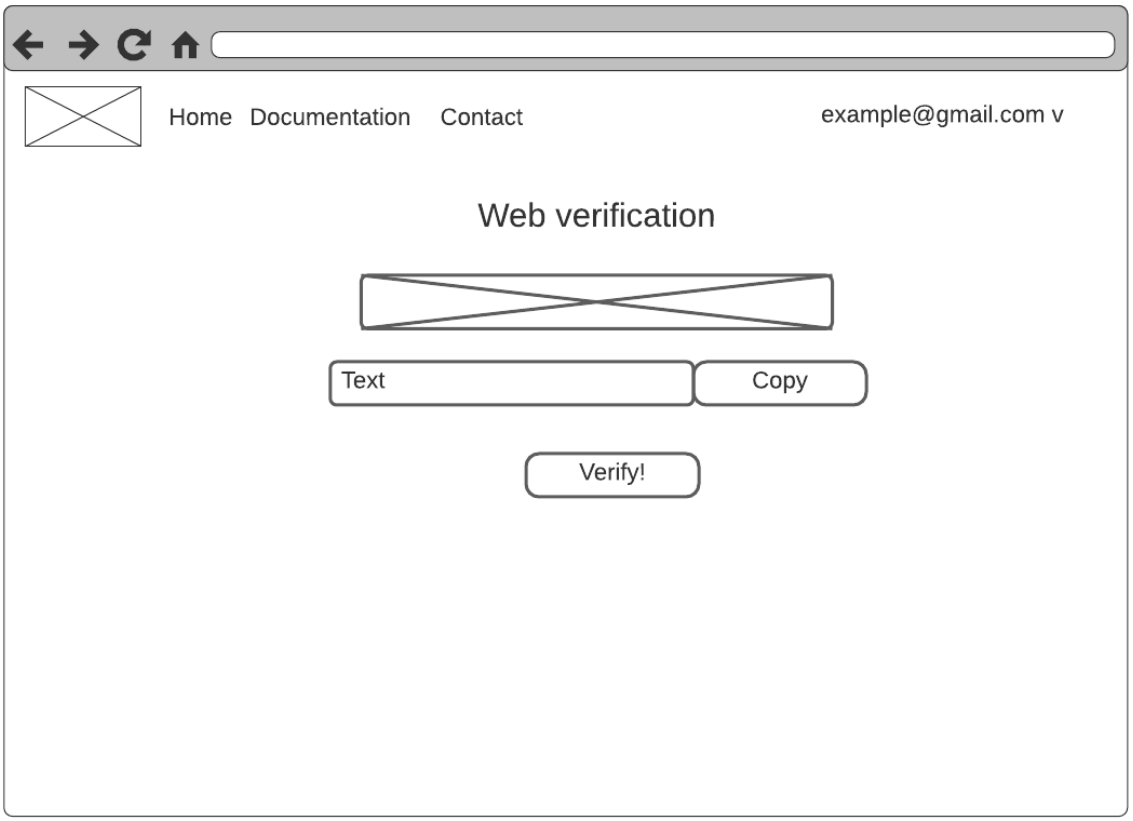
Systém zobrazí

- Pole s vygenerovanou HTML značkou;
- Tlačítko pro zkopírování značky;
- Tlačítko pro provedení verifikace;
- Tlačítko „zpět“.

Po kliknutí na

- Tlačítko zpět se zobrazí profil uživatele;
- Tlačítko verifikovat proběhne verifikace, pokud web nebude verifikován, zobrazí se chybová hláška. V případě úspěšné verifikace se zobrazí detail nového webu.

Logický design



Detail webu

Use case

Uživatel očekává

- Zobrazení základních informací o webu;
- Možnost návratu zpět na profil uživatele;
- Možnost dostat se k editaci klíčových slov u verifikovaných webů v případě neprobíhající těžby dat;
- Možnost smazat daný web v případě neprobíhající těžby dat;
- Možnost spustit jednotlivé typy crawlerů pro verifikované weby, v případě neprobíhající těžby dat;
- Seznam historických crawlerů s odkazy k bližším informacím
- Možnost smazání historického crawleru
- Informaci o případné probíhající těžbě dat;
- Možnost opětovné verifikace webu v případě neverifikovaných webů;
- Informace o případných chybových hlášení.

Scénář

Systém zobrazí

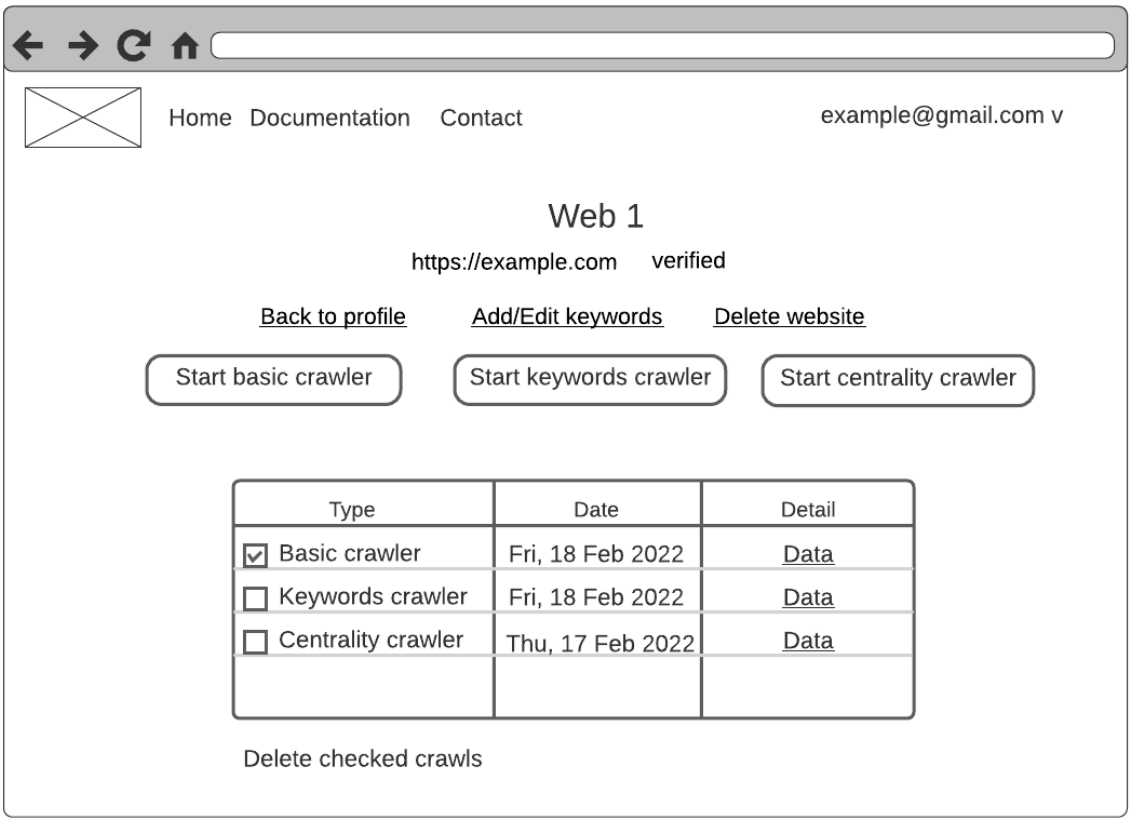
- Název a URL dané webové stránky;
- Štítek zobrazující informaci o tom, zda byl web verifikován.
- Tlačítko „Back to profile“
- Tlačítko pro spuštění funkce Basic crawler v případě verifikovaného webu a neprobíhající těžby dat;
- Tlačítko pro spuštění funkce Keywords crawler v případě verifikovaného webu a neprobíhající těžby dat;
- Tlačítko pro spuštění funkce Centrality crawler v případě verifikovaného webu a neprobíhající těžby dat;
- Tabulku s výčtem historických těžeb dat s jejich typem, datem ukončení těžby a odkazem na detailní informace o provedené těžbě a checkboxy pro jejich označení
- Tlačítko pro smazání historických těžeb dat
- Hlášku o tom, že web není verifikován – v případě neverifikovaného;
- Odkaz na stránku pro editaci klíčových slov webu v případě verifikovaného webu a neprobíhající těžbě dat;
- Hlášku o přesáhnutí povoleného počtu padesáti záznamů dat, pokud tak nastane. Systém pak neumožní spuštění těžebních skriptů
- Tlačítko pro smazání webu v případě neprobíhající těžby dat;

Po kliknutí na

- Tlačítko „Back to profile“ systém zobrazí stránku profilu uživatele;
- Odkaz na stránku pro editaci klíčových slov systém zobrazí stránku pro editaci klíčových slov;
- Tlačítko smazání webu systém zobrazí dialog s otázkou, zda si je uživatel jistý svým rozhodnutím a po potvrzení smaže záznamy o daném webu;
- Tlačítko smazání historických těžeb dat systém odstraní označené záznamy;
- Odkaz na detail výstupu z crawleru systém zobrazí příslušný detailní výpis;
- Tlačítko pro spuštění konkrétních crawlerů systém spustí příslušný skript a zobrazí informaci o probíhající těžbě. V případě že v začátku těžby bude zaznamenáno, že na stránce chybí verifikační značka, těžba bude ukončena

a uživatel bude na tento fakt upozorněn. Web bude označen za neverifikovaný. V případě spuštění funkce Keywords crawleru bez přiřazených klíčových slov k webu těžba neproběhne a uživatel bude na tento fakt upozorněn.

Logický design



Editace klíčových slov

Use case

Uživatel očekává

- Možnost definovat nová klíčová slova k webu;
- Možnost zobrazit již přiřazená klíčová slova k webu;
- Možnost vrátit se na detail webu;
- Možnost odstranit vybraná klíčová slova.

Scénář

Systém zobrazí

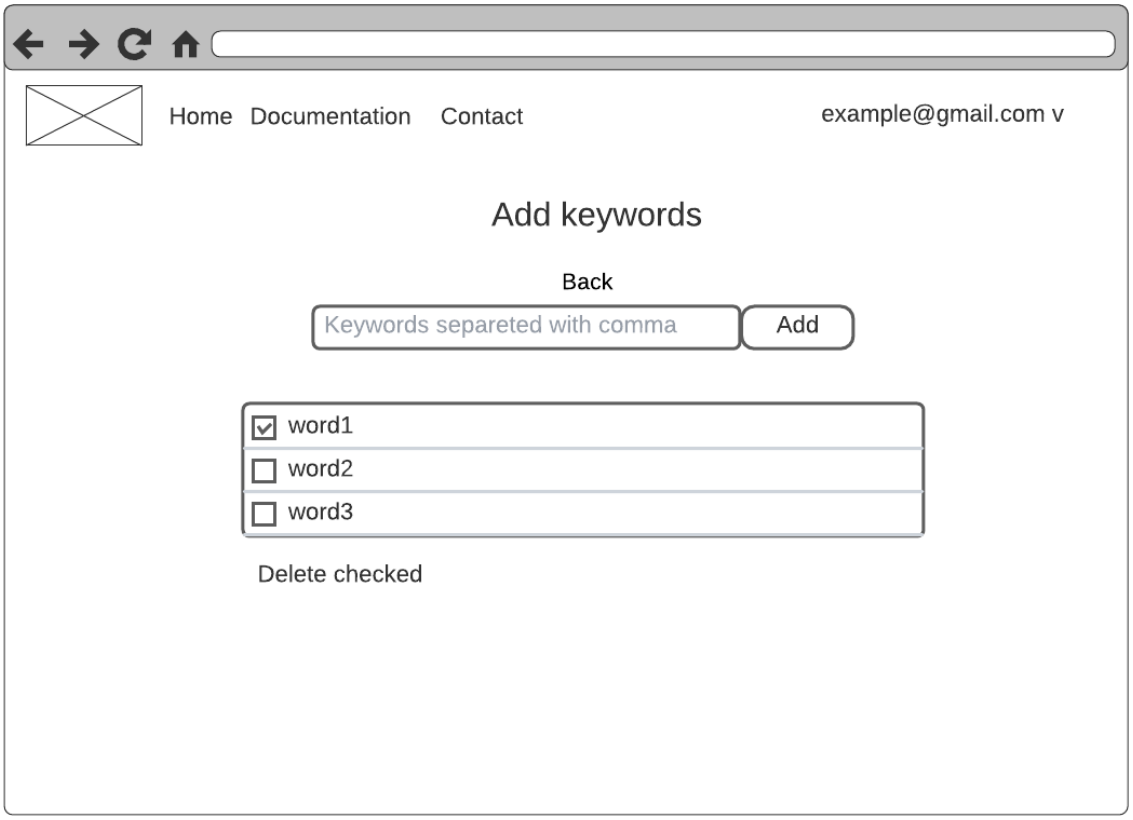
- Tlačítko pro navrácení zpět na detail webu;
- Pole pro přidání nových klíčových slov;

- Tlačítko pro odeslání sepsaných nových klíčových slov;
- Seznam s již přiřazenými klíčovými slovy s možností jejich označení;
- Tlačítko pro odstranění vybraných klíčových slov.

Po kliknutí na

- Tlačítko přidání klíčových slov systém provede uložení slov, v případě duplikace mezi slovy odstraní duplikáty;
- Tlačítko smazání klíčových slov systém smaže vybraná klíčová slova.

Logický design



Detail „Basic crawler“

Use case

Uživatel očekává

- Základní informace o provedené těžbě dat;
- Možnost návratu zpět na detail webu;
- Možnost zvolit si typ vypisovaných dat;
- Možnost zobrazení nápověd pro některé výstupy;

- Možnost zobrazení všech výstupů z provedené těžby dat.

Scénář

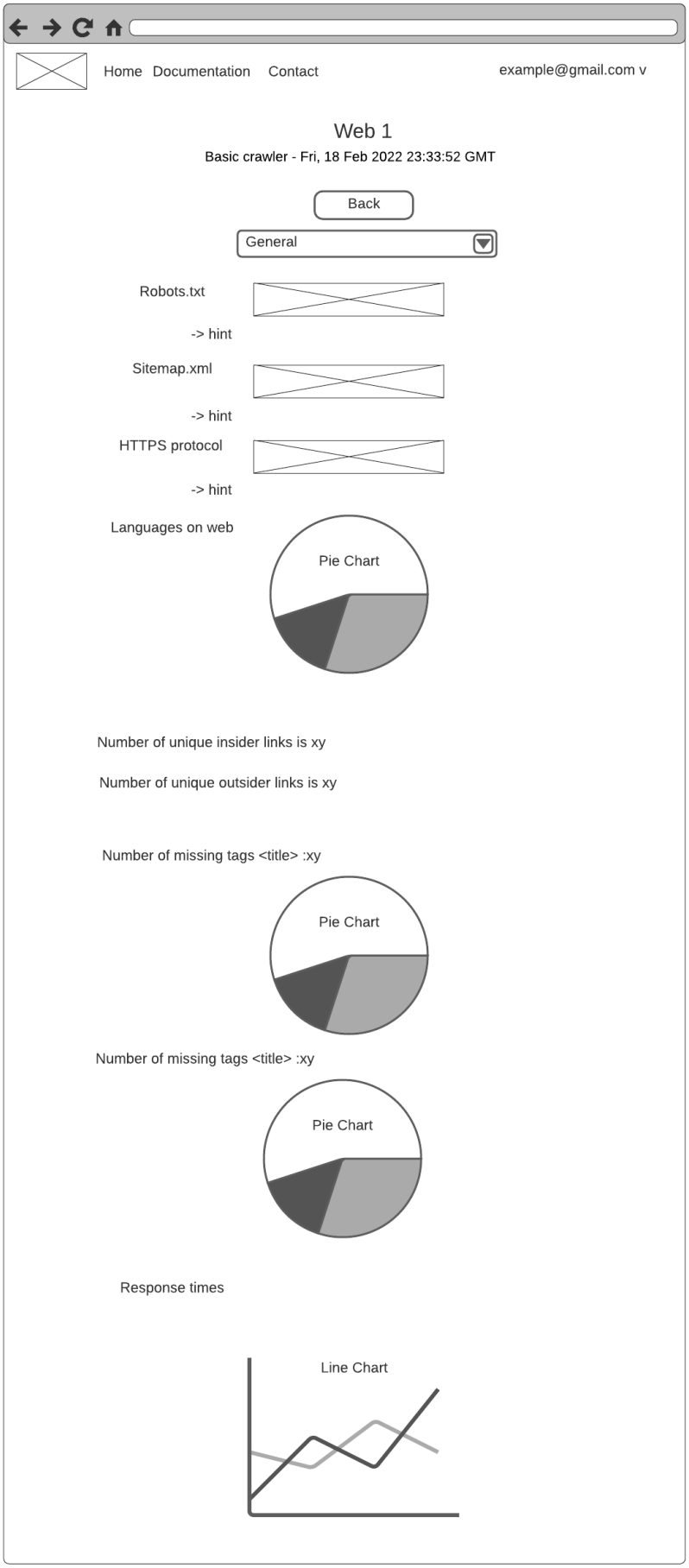
Systém zobrazí

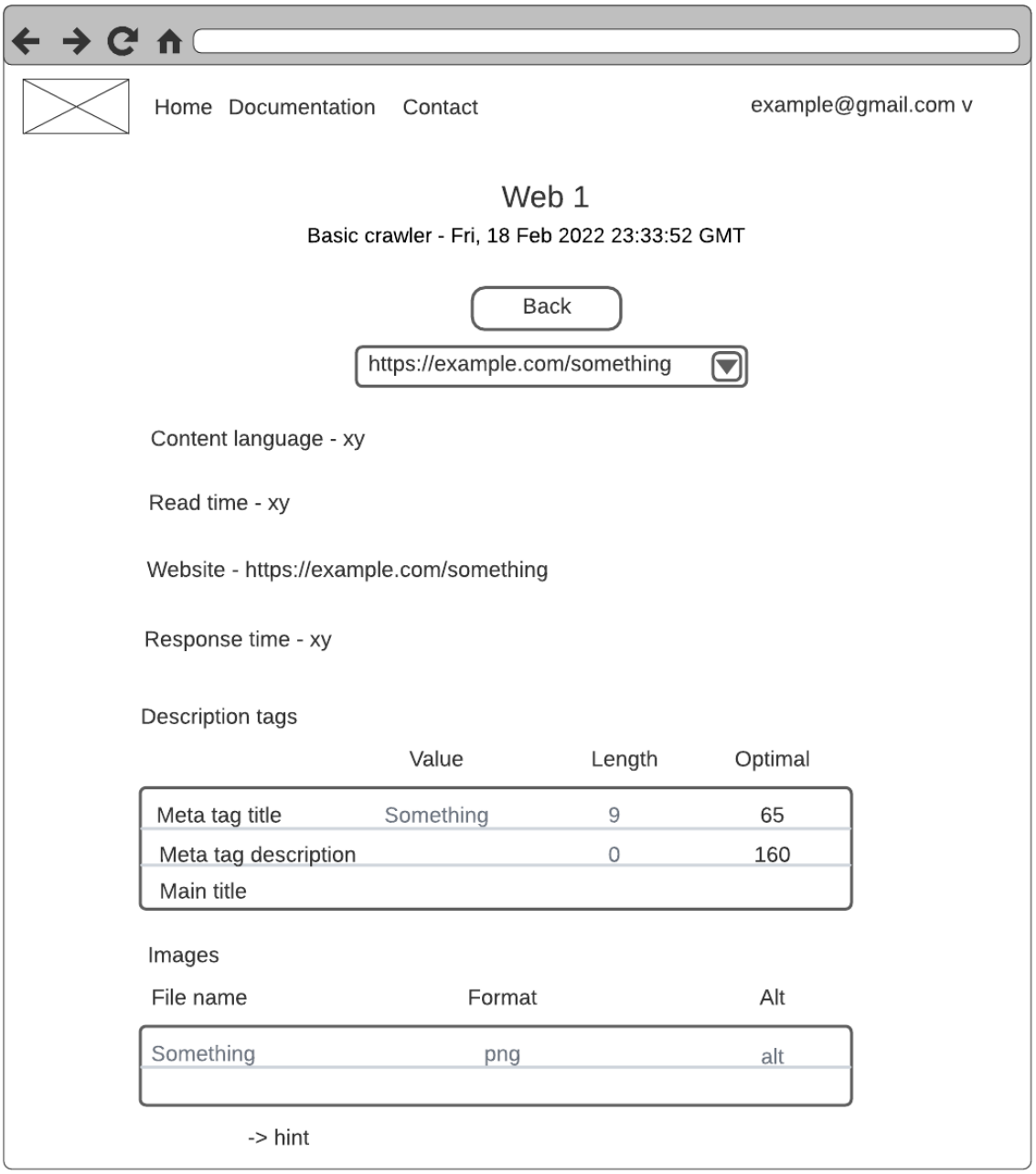
- Tlačítka pro návrat zpět na detail webu;
- Výběrový box pro zvolení typu vypisovaných dat;
- Výpis informací získaných z těžby dat.

Po kliknutí na:

- Tlačítko pro návrat zpět na detail webu systém zobrazí stránku detailu webu;
- Výběrový box zobrazí výběr typu vypisovaných dat – obecná data či data pro vybrané stránky webu, po výběru změní vypisovaná data vzhledem k volbě uživatele;

Logický design


Home Documentation Contact
example@gmail.com v
Web 1
Basic crawler - Fri, 18 Feb 2022 23:33:52 GMT
Back
General
Robots.txt
-> hint
Sitemap.xml
-> hint
HTTPS protocol
-> hint
Languages on web
Pie Chart
Number of unique insider links is xy
Number of unique outsider links is xy
Number of missing tags <title> :xy
Pie Chart
Number of missing tags <title> :xy
Pie Chart
Response times
Line Chart



Detail "Keywords crawler"

Use case

Uživatel očekává

- Základní informace o provedené těžbě dat;
- Možnost návratu zpět na detail webu;
- Možnost zvolit si typ vypisovaných dat;
- Možnost zobrazení všech výstupů z provedené těžby dat.

Scénář

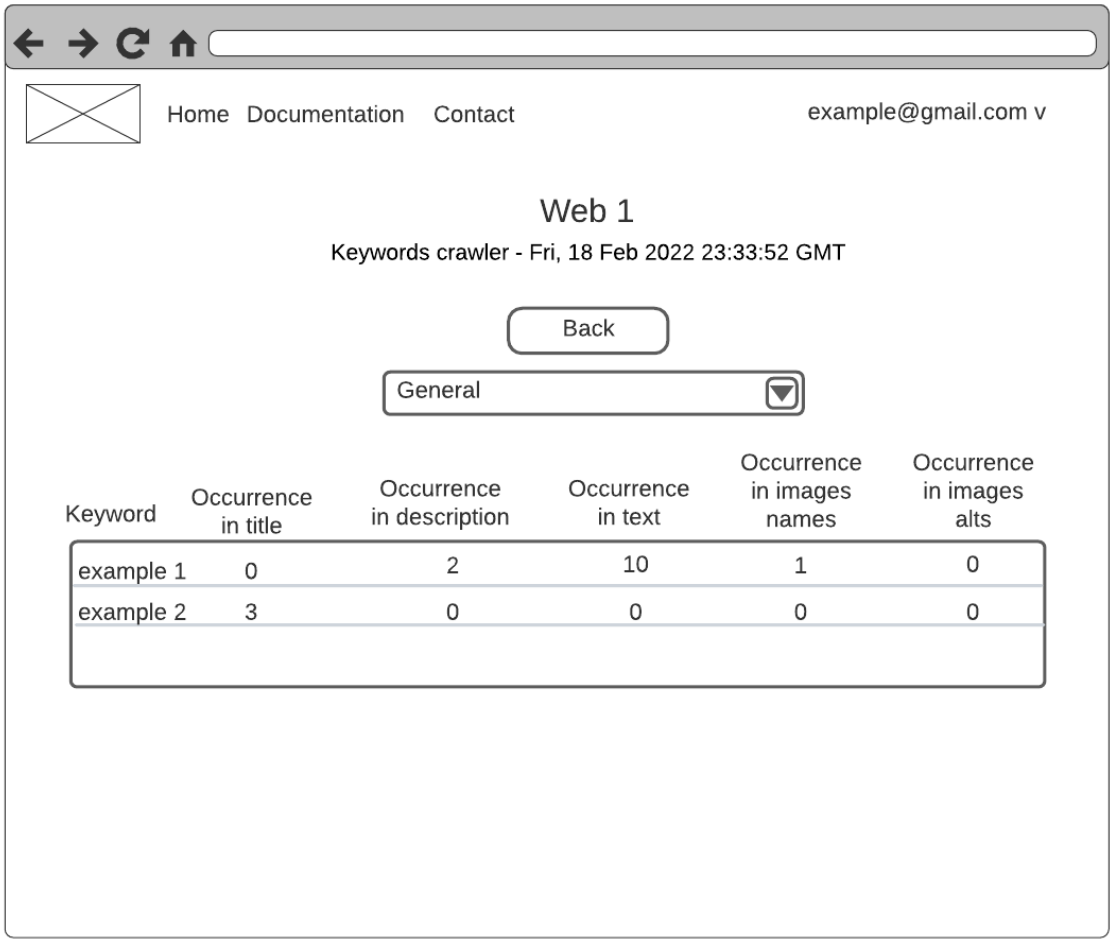
Systém zobrazí

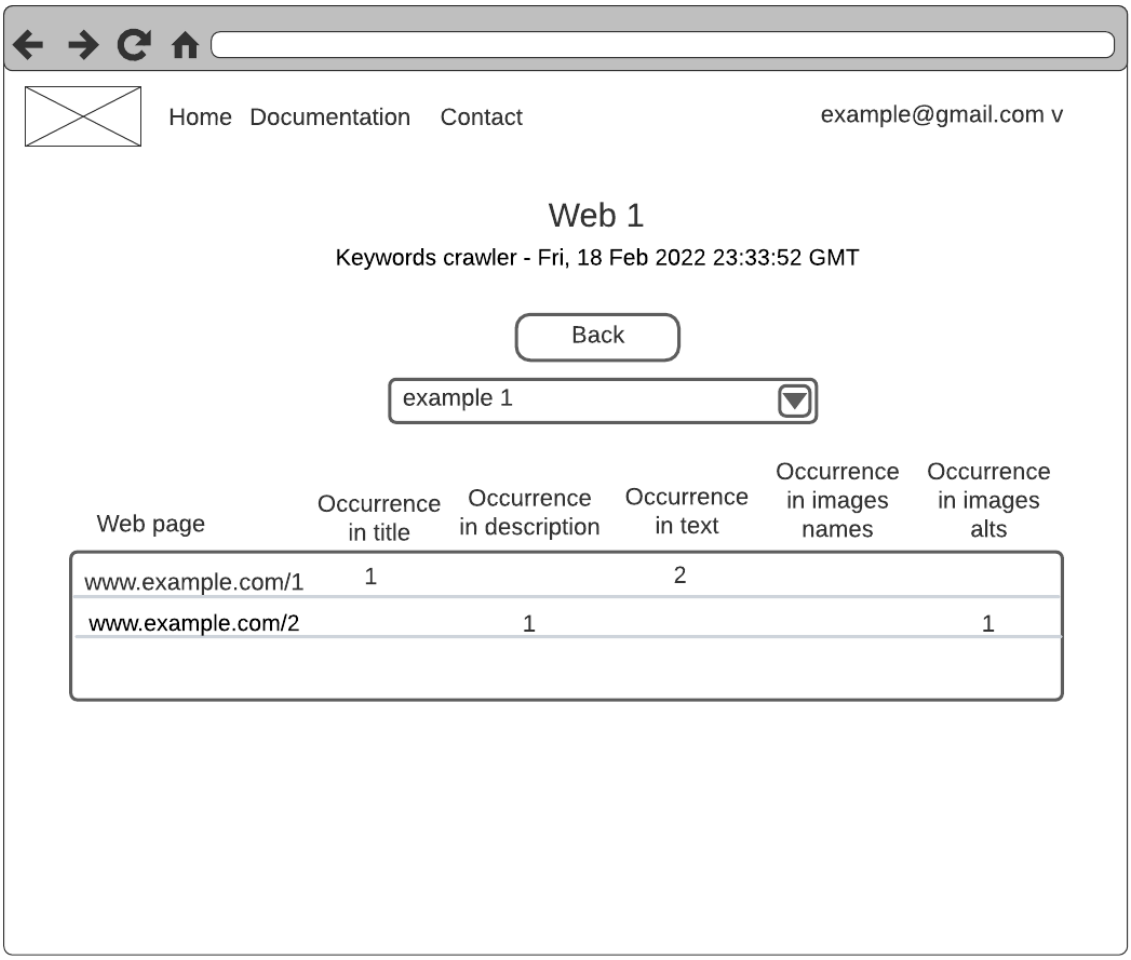
- Tlačítka pro návrat zpět na detail webu;
- Výběrový box pro zvolení typu vypisovaných dat;
- Výpis informací získaných z těžby dat ve formě tabulek s informacemi o nalezených uživatelem definovaných klíčových slovech v konkrétních částech webu.

Po kliknutí na:

- Tlačítko pro návrat zpět na detail webu systém zobrazí stránku detailu webu;
- Výběrový box systém zobrazí výběr typu vypisovaných dat – obecná data či data pro vybrané konkrétní vybrané klíčové slovo, po výběru změní vypisovaná data vzhledem k volbě;

Logický design





Detail "Centrality crawler"

Use case

Uživatel očekává

- Základní informace o provedené těžbě dat;
- Možnost návratu zpět na detail webu;
- Možnost zvolit si typ vypisovaných dat;
- Možnost zobrazení nápověd pro některé výstupy;
- Možnost zobrazení všech výstupů z provedené těžby dat
- Možnost změny vypsaného webového grafu v závislosti na zvolených webových stránkách, které mají být v grafu zahrnuty

Scénář

Systém zobrazí

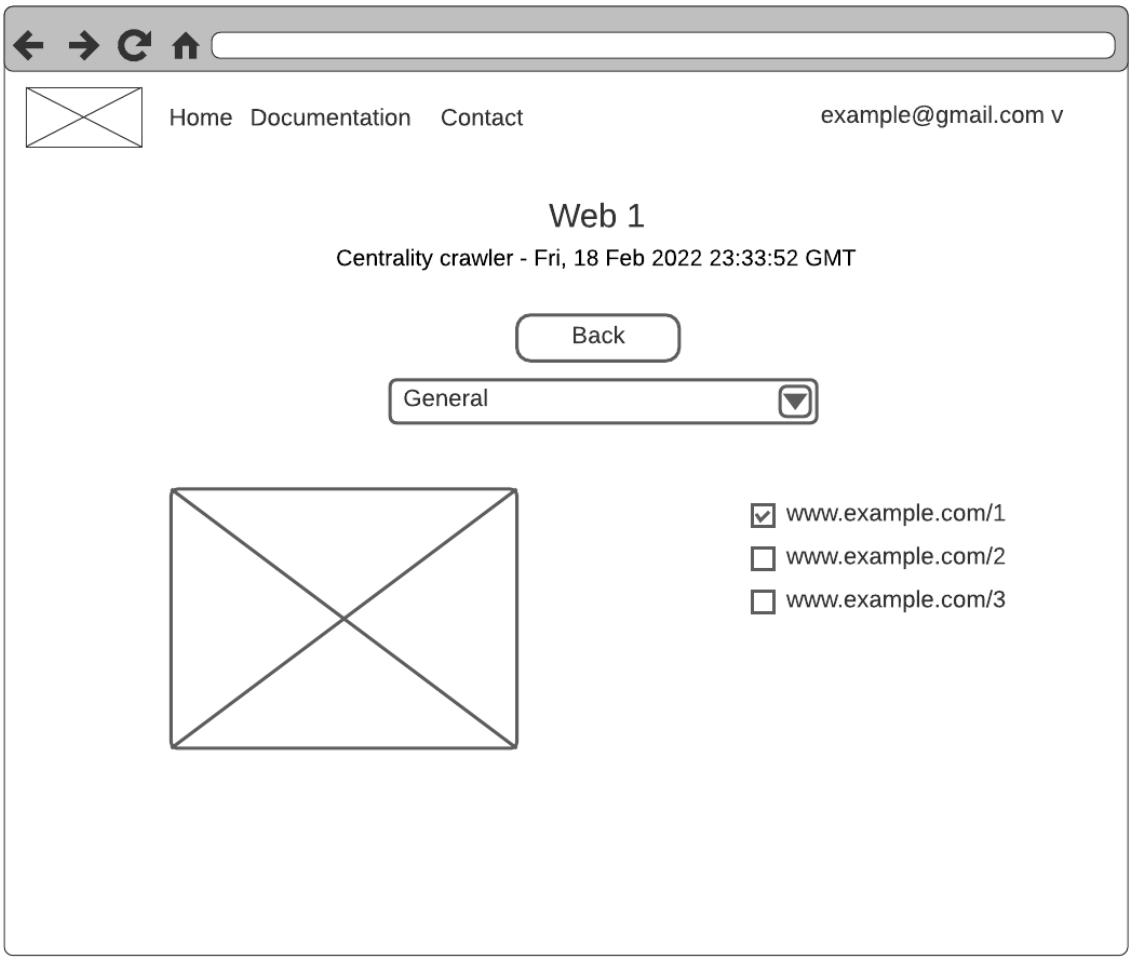
- Tlačítka pro návrat zpět na detail webu;

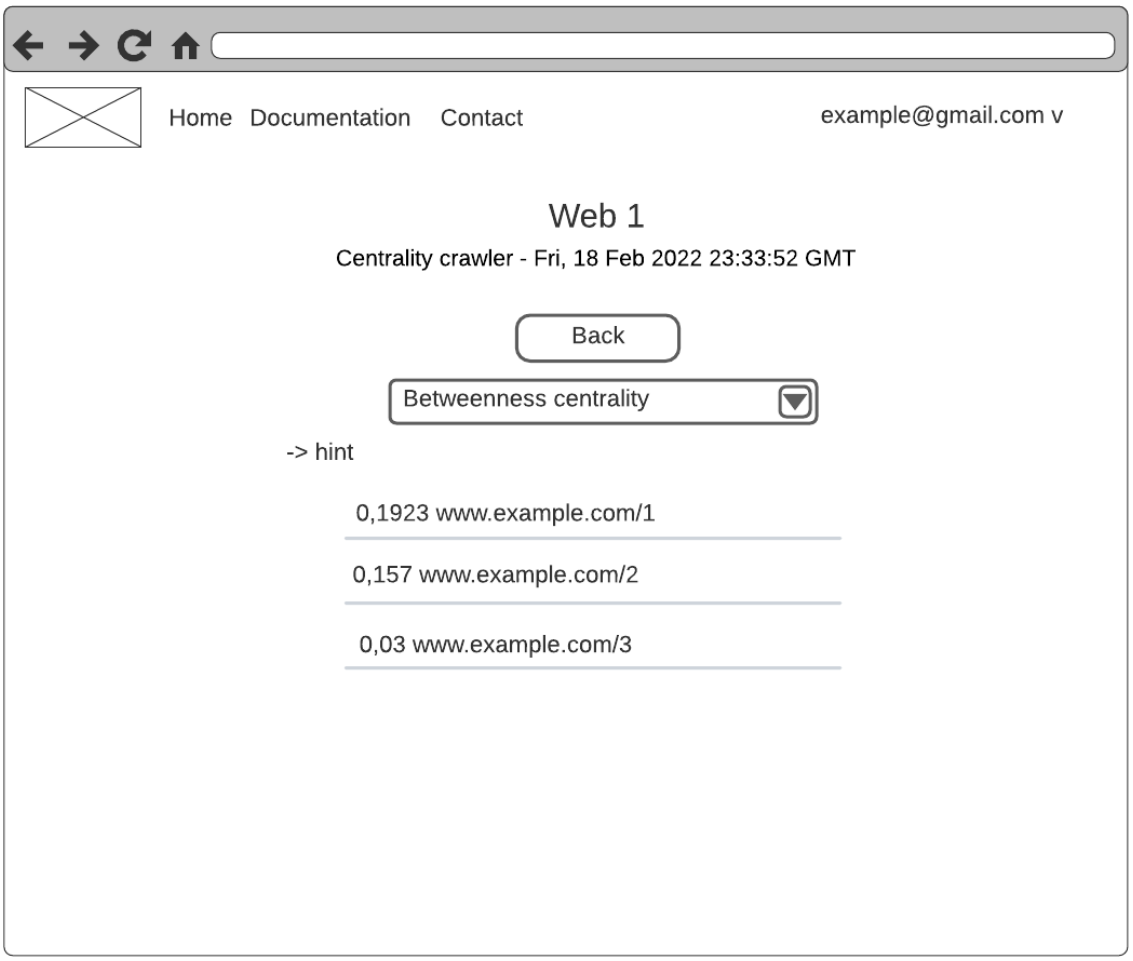
- Výběrový box pro zvolení typu vypisovaných dat;
- Výpis informací získaných z těžby dat v podobě tabulek s vypočítanými hodnotami centralit pro veškeré stránky webu a upravitelného webového grafu

Po kliknutí na:

- Tlačítko pro návrat zpět na detail webu systém zobrazí stránku detailu webu;
- Výběrový box systém zobrazí výběr typu vypisovaných dat – obecná data či data pro konkrétní míry centralit. V případě výběru obecných dat zobrazí webový graf s označitelnými webovými stránkami v seznamu. V případě jejich označení zahrne dané stránky do webového grafu

Logický design


Home Documentation Contact
example@gmail.com v
Web 1
Centrality crawler - Fri, 18 Feb 2022 23:33:52 GMT
Back
General
www.example.com/1
www.example.com/2
www.example.com/3


Home Documentation Contact
example@gmail.com v
Web 1
Centrality crawler - Fri, 18 Feb 2022 23:33:52 GMT
Back
Betweenness centrality
-> hint
0,1923 www.example.com/1
0,157 www.example.com/2
0,03 www.example.com/3