

**UNIVERZITA KONŠTANTÍNA FILOZOFA V NITRE**  
**FAKULTA PRÍRODNÝCH VIED A INFORMATIKY**

**NÁVRH WEBOVÉHO CRAWLERA**  
**BAKALÁRSKA PRÁCA**

**2022**

**MARTIN PECHO**

**UNIVERZITA KONŠTANTÍNA FILOZOFA V NITRE**  
**FAKULTA PRÍRODNÝCH VIED A INFORMATIKY**

**NÁVRH WEBOVÉHO CRAWLERA**  
**BAKALÁRSKA PRÁCA**

Študijný odbor: 18. Informatika  
Študijný program: Aplikovaná informatika  
Školiace pracovisko: Katedra informatiky  
Školiteľ: Mgr. Ľubomír Benko, Ph.D.

# ZADANIE ZÁVEREČNEJ PRÁCE



Univerzita Konštantína Filozofa v Nitre  
Fakulta prírodných vied a informatiky

## ZADANIE ZÁVEREČNEJ PRÁCE

**Meno a priezvisko študenta:** Martin Pecho  
**Študijný program:** aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)  
**Študijný odbor:** informatika  
**Typ záverečnej práce:** Bakalárska práca  
**Jazyk záverečnej práce:** slovenský  
**Sekundárny jazyk:** anglický  
**Názov:** Návrh webového crawlera  
**Anotácia:** Webový crawler je v informatike špecializovaný robot, ktorý slúži na prehľadávanie webových stránok za účelom získania veľkého množstva dát. Tým, že navštevuje automaticky všetky dostupné webové stránky, dokáže zaznamenávať informácie z daných webových stránok. Cieľom práce je analýza aktuálnych riešení pri navrhovaní webových crawlerov a implementácia webového crawlera pre vybranú oblasť. Charakter práce: aplikačný. Požiadavky na obsah podľa charakteru práce (nemyslí sa tým šablóna, tá je daná bez ohľadu na charakter práce): popis riešeného problému, návrh systému/hardvérového riešenia a pod. (modely, ..), metodika vývoja/tvorby, implementácia, popis vytvoreného riešenia, testovanie. Požiadavky na vedomosti a zručnosti študenta: dobrá znalosť anglického jazyka, dobrá znalosť z programovania, analytické myslenie.

**Školiteľ:** Mgr. Ľubomír Benko, Ph.D.  
**Oponent:** Mgr. Dominik Halvoník, PhD.  
**Katedra:** KI - Katedra informatiky  
**Dátum zadania:** 27.10.2020

**Dátum schválenia:** 05.04.2021

RNDr. Ján Skalka, PhD., v. r.  
vedúci/a katedry

## **POĎAKOVANIE**

Veľmi rád by som sa chcel poďakovať môjmu školiteľovi Mgr. Ľubomírovi Benkovi Ph.D., za jeho ochotu, ústretovosť, odborné rady a celkovú pomoc pri riešení bakalárskej práce.

## ABSTRAKT

PECHO, Martin: Návrh webového crawlera. [Bakalárska práca]. Univerzita Konštantína Filozofa v Nitre. Fakulta prírodných vied a informatiky. Školiteľ: Mgr. Ľubomír Benko, Ph.D. Stupeň odbornej kvalifikácie: Bakalár odboru Aplikovaná informatika. Nitra: FPVaI, 2022. 43 s.

Web je plný informácií. V dnešnej dobe, ak niekto hľadá informácie, väčšinou zamieri na internet. Aby sa rýchlo dostal k požadovaným výsledkom, je potrebné, aby bol internet neustále prechádzaný. Pri takom množstve dát ľudské sily nestačia. Preto túto úlohu vykonávajú robotické programy, ktoré sa nazývajú web crawlery. Web crawler slúži na prehľadávanie webových stránok za účelom získania veľkého množstva dát. Táto bakalárska práca sa zaoberá analýzou aktuálnych riešení pri navrhovaní webových crawlerov. Cieľom práce je návrh a vytvorenie webového prehľadávača, ktorý hľadá súbory RSS. Really Simple Syndication (RSS) obsahuje informácie o aktuálnom obsahu webu, ktoré sú poskytované prostredníctvom webových kanálov. Webová aplikácia umožňuje vyhľadávanie RSS na základe vstupného výrazu a pre zadanú URL overí, či pre danú lokalitu existuje.

Kľúčové slová: RSS. Crawler. Python. Web.

## **ABSTRACT**

PECHO, Martin: Design of web crawler. [Bachelor Thesis]. Constantine the Philosopher University in Nitra. Faculty of Natural Sciences and Informatics. Supervisor: Mgr. Ľubomír Benko, Ph.D. Degree of Qualification: Bachelor of Applied Informatics. Nitra: FNSaI, 2022. 43 p.

The web is full of information. Nowadays, when someone searches for information, they mostly use the Internet. In order to get the desired results quickly, the internet needs to be constantly crawled. Human strength is limited by too much data. Therefore, robotic programs called web crawlers perform this task. Web crawler is used to browse websites to obtain large amounts of data. This bachelor thesis deals with the analysis of current solutions in designing of web crawlers. The aim of the thesis is to design and create a web crawler that searches for RSS files. Really Simple Syndication (RSS) contains information about current web content, which is provided through web feeds. The web application allows RSS crawls based on the input term and verifies for the specified URL whether it exists for the given site.

Keywords: RSS. Crawler. Python. Web.

# OBSAH

<b>Úvod .....</b>	<b>9</b>
<b>1    Analýza súčasného stavu.....</b>	<b>10</b>
1.1    Webový Crawler.....	11
1.2    Internet .....	11
1.3    Ako funguje Web.....	11
1.4    Web scraping .....	12
1.5    Architektúra crawlera .....	12
1.6    Typy crawlerov.....	13
1.7    Algoritmy .....	15
1.8    Vyhľadávacia politika.....	16
1.9    Etika crawlovania .....	18
<b>2    Ciele záverečnej práce.....</b>	<b>21</b>
Podciele alebo čiastkové ciele:.....	21
<b>3    Metódy riešenia projektu.....</b>	<b>22</b>
3.1.    Používané technológie.....	22
3.1.1    Python.....	22
3.1.1    DuckDuckGo .....	23
3.1.2    Flask .....	23
3.1.3    Angular .....	24
<b>4    Výsledky riešenia a ich zhodnotenie .....</b>	<b>25</b>
4.1    Návrh webovej aplikácie .....	25
4.1    Implementácia .....	26
4.2    Testovanie a riešenie problémov.....	31
4.3    Získanie a analýza dát.....	33
<b>Záver .....</b>	<b>36</b>
<b>Zoznam bibliografických odkazov .....</b>	<b>37</b>
<b>Zoznam príloh .....</b>	<b>41</b>

## ZOZNAM ILUSTRÁCIÍ A TABULIEK

Obrázok 1 Ročný rast dát .....	10
Obrázok 2 Architektúra webového crawlera .....	13
Obrázok 3 Rozdelenie webu.....	15
Obrázok 4 Vývoj veličín čerstvosti a veku.....	17
Obrázok 5 Príklad RSS súboru denníka Pravda .....	22
Obrázok 6 Ukážka súboru constants.py .....	24
Obrázok 7 Diagram vytvorený pomocou Draw.io .....	25
Obrázok 8 Paleta najčastejších farieb.....	26
Obrázok 9 Návrh úvodnej obrazovky v softvéri Figma .....	26
Obrázok 10 Nastavenie limitera .....	27
Obrázok 11 Nájsené RSS pre slovo „news“ .....	28
Obrázok 12 Ukážka dynamického formulára.....	28
Obrázok 13 Ukážka filtrovania v režime „s emailom“.....	29
Obrázok 14 Ukážka emailu s výsledkami .....	30
Obrázok 15 Zobrazenie výsledkov.....	31
Obrázok 16 Ukážka overeného RSS z adresy webu.....	31
Obrázok 17 Výsledky testov.....	32



# ÚVOD

Každý z nás máme v sebe túžbu po poznaní. Človek prirodzene hľadá, skúma nové informácie. Postupom rokov sa ľudstvo posunulo od tradovania, cez knihy až k modernej dobe internetu. Dnes si väčšina z nás nevie predstaviť život bez internetu. Veľakrát sa ani nezamýšľame nad tým, aké technológie stoja za našim jedným kliknutím. Na druhej strane počas toho každú sekundu pribúdajú nové a nové webové stránky plné informácií.

Crawlovanie je proces, v ktorom automatizovaný robot prelieza web. Tento robot má za úlohu zbierať špecifický typ informácií z webových stránok. Možno to znie jednoducho, ale vytvoriť efektívny prehľadávač je náročné. Medzi základné vlastnosti, ktoré by mal spĺňať patrí rýchlosť, bezpečnosť, spoľahlivosť a efektívnosť.

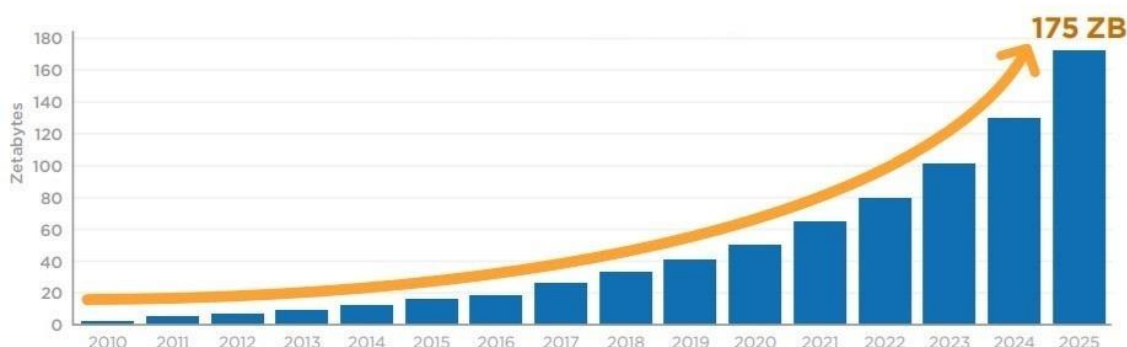
V našej práci sa budeme snažiť navrhnúť webového crawlera, s ktorého pomocou budeme hľadať RSS súbory z rôznych kútov sveta. Požiadavky pre tento systém vznikli v spolupráci s firmou Dominanz. Pre bezpečné získavanie URL odkazov by sme chceli použiť knižnicu DuckDuckGo. Za pomoci knižnice BeautifulSoup z nich budeme získavať obsah, v ktorom budeme zisťovať, či pre daný web alebo subdoménu existuje RSS súbor. K hlavným funkciám programu bude používateľ pristupovať pomocou služby Flask. Okrem serverovej časti by sme radi vytvorili prívetivé používateľské rozhranie pomocou moderného frameworku Angular. Medzi čiastkové ciele si kladieme navrhnúť, implementovať a pretestovať aplikáciu. Následne pomocou nej by sme chceli získať údaje, ktoré budeme vo výsledkoch analyzovať.

Hlavným zdrojom informácií tejto práce budú najmä odborné články a citácie. Prácu sme rozdelili do štyroch kapitol. V prvej kapitole si povieme o informáciách a ako s nimi vieme pracovať, taktiež si vysvetlíme pojem crawlovanie a všetko s ním súvisiace. V druhej si priblížime, ciele našej práce, ktoré chceme dosiahnuť. V tretej si popíšeme najdôležitejšie technológie použité pri tvorbe softvéru. V štvrtej časti popíšeme pracovné postupy a dosiahnuté ciele, predstavíme program, ktorý vytvoríme a taktiež si zanalyzujeme nadobudnuté dáta. Celú bakalársku prácu v závere zhrnieme.

# 1 ANALÝZA SÚČASNÉHO STAVU

Prístup k informáciám je životne dôležitý pre sociálny, politický a ekonomický pokrok. Tradične sa informácie širili v rôznych formátoch, ktoré boli široko dostupné, často prostredníctvom verejných knižníc. Mnoho jednotlivcov sa pri informáciách spoliehalo aj na iných ľudí a médiá. V súčasnosti je zaznamenaný značný exponenciálny nárast hodnoty nehmotných a digitálnych komodít, ako sú dáta. Údaje sa teraz považujú za najcennejší a najzraniteľnejší zdroj na svete. Samozrejme, úroveň zraniteľnosti okolo údajov sa neustále zvyšuje s rastom zdroja (Brown a Duguid, 2000).

Mark Allinson (2021) uverejnil článok, v ktorom tvrdí, že v roku 2014 existovalo na svete celkovo 10 miliárd zariadení pripojených k internetu, pričom v tom čase odborníci predpovedali, že do konca roka 2020 ich počet dosiahne 20 miliárd (Allinson, 2021). Aj preto každým dňom počet dát neustále narastá. Do roku 2025 spoločnosť IDC predpovedá, že počet vygenerovaných dát bude pozostávať zo 175 zettabajtov. Ich stiahnutie by trvalo 1,8 miliardy rokov (Reinsel a kol., 2018).



Obrázok 1 Ročný rast dát<sup>1</sup>

Vytvorenie celosvetovej siete v roku 1989 spôsobilo revolúciu v našej histórii komunikácie a prenosu informácií. Mapovaním tejto siete a toho, ako boli jednotlivé webové stránky navzájom prepojené sa začalo s prehľadávaním webu (web crawling). Prehľadávače sa používajú na zhromažďovanie informácií, ktoré sa potom môžu použiť a spracovať na klasifikáciu dokumentov a poskytovanie prehľadov o zhromaždených údajoch (Picot, 2016).

<sup>1</sup> Zdroj obrázok: <https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf>

## **1.1 WEBOVÝ CRAWLER**

Webový crawler je automatizovaný robot, ktorý sťahuje a indexuje obsah z celého internetu. Cieľom takého robota je zistiť, o čom je každá webová stránka na internete, aby bolo možné informácie získať, kedykoľvek to bude potrebné. Webové prehľadávače sa im hovorí, pretože indexové prehľadávanie je výraz pre automatický prístup na webovú stránku a získavanie údajov o nej. Použitím vyhľadávacích algoritmov sa údaje zhromažďujú. Vyhľadávacie nástroje môžu poskytovať relevantné odkazy v reakcii na vyhľadávacie dotazy používateľov a generovať zoznam webových stránok, ktoré sa zobrazia potom, ako používateľ zadá kľúčový výraz do vyhľadávacieho nástroja. Najdôležitejšou úlohou webového prehľadávača je prehľadávať veľké dáta na internete, nájsť efektívne informácie a uložiť potrebné údaje do lokálnej databázy (Hamid a Hashem, 2016).

## **1.2 INTERNET**

Internet je celosvetová dátová sieť, ktorá je dostupná pre každého. Skladá sa z množstva sietí. Jednou z nich je sieť s paketovým prepínaním, pri ktorom sa používa Internet Protocol. Tou najvyužívanejšou možnosťou sú web stránky (Švec, Reichel, Kuna, 2020).

## **1.3 AKO FUNGUJE WEB**

Web je všeobecný názov World Wide Web, podmnožiny internetu pozostávajúceho zo stránok, ku ktorým je prístup prostredníctvom webového prehliadača. Mnoho ľudí predpokladá, že web je rovnaký ako internet, a tieto termíny používajú zameniteľne. Pojem internet však v skutočnosti označuje globálnu sieť serverov, ktorá umožňuje zdieľanie informácií prostredníctvom webu. Web síce tvorí veľkú časť internetu, ale nie sú jedno a to isté.

Webové stránky sú formátované v jazyku nazývanom HTML. Tento jazyk používateľom umožňuje klikáť na stránky na webe prostredníctvom odkazov. Web používa hypertextový protokol HTTP na prenos údajov a zdieľanie informácií. Na prístup k webovým dokumentom alebo webovým stránkam, ktoré sú prepojené prostredníctvom odkazov, slúžia prehliadače (Techopedia, 2012). Web je dynamický priestor, ktorý nemá stanovený štandard pre dátové formáty a štruktúry. Zhromažďovanie údajov vo formáte, ktorý je zrozumiteľný pre stroje, môže byť problémom kvôli nedostatočnej jednotnosti. Proces extrakcie údajov sa stáva

náročným, keď webové prehľadávače potrebujú štruktúrované údaje vo veľkom meradle. Problém sa zosilní, keď musia extrahovať údaje z tisícov webových zdrojov týkajúcich sa konkrétnej schémy.

## **1.4 WEB SCRAPING**

Web scraping (škrabanie webu) je pojem, ktorý zoskupuje metódy získavania informácií z internetu. Väčšinou ide o extrakciu dát z webových stránok pomocou skriptov, či umelej inteligencie za účelom ďalšieho spracovania. Škrabanie webu je možné robiť manuálne a automaticky.

### **Manuálne škrabanie webu**

Manuálne škrabanie zahŕňa kopírovanie a vkladanie webového obsahu, čo si vyžaduje veľa úsilia a veľmi sa opakuje v spôsobe, akým sa vykonáva. Ide o najjednoduchší spôsob a nie sú potrebné nové znalosti. Vďaka ľudskej kontrole každého dátového bodu je zabezpečené selektovanie nepodstatných informácií. Toto je účinný, ale nepraktický spôsob v boji proti zablokovaniu prístupu k webovej stránke. V praxi sa vyskytuje len zriedka, pretože automatizované škrabanie je oveľa rýchlejšie a lacnejšie (Radware, 2017).

### **Automatické škrabanie webu**

Automatizované nástroje na zoškrabovanie webu sa stávajú čoraz obľúbenejšími vďaka ich jednoduchému použitiu a úspore času a nákladov. V krátkom čase poskytuje extrakciu veľkého množstva dát. Aktuálne nástroje sú ľahko použiteľné a umožňujú uloženie dát vo vybranom formáte. Nevýhodou môže byť dnes už časté blokovanie robota webovými stránkami a spracované veľké množstvo dát bez ľudskej kontroly. Medzi najznámejšie metódy patrí HTML parsing, ktorý spracováva a analyzuje obsah HTML stránok, DOM Parsing, ktorý využíva formát Document Oriented Model pre spracovanie XML súborov a Indexing, ktorý slúži k zberu hypertextových odkazov (Perez, 2021).

## **1.5 ARCHITEKTÚRA CRAWLERA**

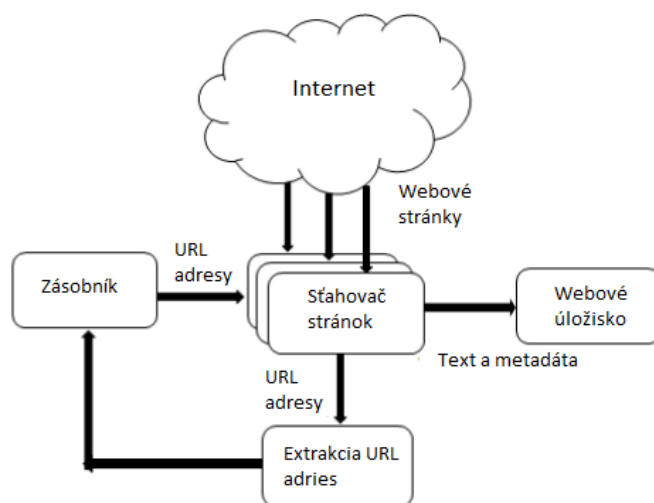
Vo všeobecnosti sa architektúra webového prehľadávača skladá z troch hlavných častí: zásobník, sťahovač stránok a webové úložisko.

Zásobník ukladá zoznam adres URL, ktoré sa majú navštíviť. Cyklus získavania a extrahovania adresy URL pokračuje, kým nie je prázdny alebo kým iná podmienka

nespôsobí jeho zastavenie. Samotné extrahovanie adries URL prebieha na základe stanovenej priority (Amudha a Phil, 2017).

Hlavnou úlohou sťahovača je načítavať stránku z internetu pre zodpovedajúce URL adresy, ktoré získava zo zásobníka. K tomu je nevyhnutný HTTP klient, pomocou ktorého sa vykoná odoslanie HTTP požiadavky a následné spracovanie odpovede (Amudha a Phil, 2017).

Poslednou časťou je webové úložisko, ktoré slúži na ukladanie a správu dát z webových stránok. Ukladajú sa iba štandardné HTML stránky. Všetky ostatné typy médií a dokumentov crawler ignoruje (Amudha a Phil, 2017).



Obrázok 2 Architektúra webového crawlera<sup>2</sup>

## 1.6 TYPY CRAWLEROV

Webové crawlery zaraďujeme do viacerých kategórií. V nasledujúcej časti si popíšeme len niektoré z nich.

### Paralelný crawler

S rastúcou veľkosťou webov bolo čoraz ťažšie získať údaje z celej web stránky alebo aspoň podstatnú časť pomocou jediného procesu. Preto mnohé crawlery často spúšťajú viacero procesov paralelne, takže rýchlosť sťahovania je maximalizovaná. Tento typ sa nazýva paralelný crawler (Ahuja, Singh, Varnica, 2014).

### Inkrementálny crawler

Prírastkový crawler pracuje na princípe aktualizácie existujúcej množiny stiahnutých stránok namiesto opätovného spúšťania prehľadávania od začiatku.

---

<sup>2</sup> Zdroj obrázok: <https://www.irjet.net/archives/V4/i2/IRJET-V4I225.pdf>

To zahŕňa určitý spôsob určenia, či sa stránka od posledného prehľadávania zmenila. Crawler neustále prehľadáva celý web na základe určitej sady cyklov prehľadávania. Používa sa adaptívny model, ktorý na základe údajov z predchádzajúcich cyklov rozhoduje o tom, ktoré stránky by sa mali skontrolovať, či boli aktualizované, čím sa dosiahne vysoká aktuálnosť dát (Ahuja, Singh, Varnica, 2014).

### **Distribuovaný crawler**

V distribuovanom systéme crawlery prerozdeľujú úlohy iným medzi sebou. Vzdialený server komunikuje a synchronizuje s uzlami. Existujú dve architektúry pre distribuovaný systém prehľadávania webu, a to Master Slave a Peer to Peer architektúra (Yuhao, 2018).

### **Hidden crawler**

Internet to nie sú len verejne indexované stránky, ktoré sú prístupné pomocou hypertextových odkazov, pričom ignorujú vyhľadávacie stránky a formuláre, ktoré vyžadujú autorizáciu alebo predchádzajúcu registráciu. Viacero údajov na webe sa v skutočnosti nachádza v databáze a je možné ich získať iba zaslaním vhodných dopytov alebo vyplnením formulárov na webe. Tento druh sa nazýva deep (hlboký) web alebo hidden (skrytý) web a je definovaný ako všetok obsah, ktorý nie je indexovaný webovým prehľadávačom. Väčšina crawlerov prehľadáva iba verejne dostupnú časť webu. Tú ťažšie prístupnú časť prechádza tzv. hidden crawler (Chaitra a kol., 2020).



Obrázok 3 Rozdelenie webu<sup>3</sup>

### Webový crawler so zameraním

Tento crawler dôsledne vyhľadáva webové stránky relevantné pre konkrétnu tému. Pokúša sa získať relevantnejšie stránky s vyššou úrovňou presnosti a snaží sa vyhýbať stránkam, ktoré nesúvisia s témou. Dosahuje sa to uprednostňovaním webových stránok. Na získanie relevantných stránok z webu sa používajú rôzne prístupy, ako napríklad crawler založený na prioritách, štruktúrovaný crawler, crawler založený na učení a prehľadávanie zamerané na kontext (Gupta a Anand, 2015).

## 1.7 ALGORITMY

Pri získavaní informácií pomocou crawlovania je možnosť využiť viacero techník a algoritmov. Výber správneho algoritmu závisí od toho, aké sa očakávajú výsledky.

### Algoritmus Breadth-First

Crawler sa snaží, čo najjednoduchším spôsobom prechádzať web, pričom sa sústreďuje na kvantitu spracovaných stránok. Stránky sú spracovávané v poradí, v akom sa ocitli v zásobníku. Tento algoritmus je jednoduchý, ale menej efektívny, keďže výsledky nemusia byť vždy významné (Shrivastava a Sahai, 2019).

### Algoritmus Best-First

Algoritmus je postavený na vyhodnocovaní relevantnosti stránok. Stránky sú spracovávané od tej, ktorá má najvyššie skóre. Na základe toho nie sú irelevantné

<sup>3</sup> Zdroj obrázok: [https://www.wikidata.org/wiki/Q221989#/media/File:Surface\\_Web\\_&\\_Deep\\_Web.jpg](https://www.wikidata.org/wiki/Q221989#/media/File:Surface_Web_&_Deep_Web.jpg)

stránky prehľadované. Miera relevantnosti je rôzna, v závislosti od použitia (Shrivastava a Sahai, 2019).

### **Algoritmus Fish-Search**

Algoritmus Paula De Bra simuluje ryby, ktoré si hľadajú potravu. Keď sa nájde jedlo (relevantná informácia), príde viac rýb (crawler). Hlavnou myšlienkou je, že používateľ určí hĺbku a šírku vyhľadávania a pôvodnú URL adresu. Následne sa vytvorí zoznam prístupových adries podľa priority. V tomto prípade sú potom uprednostňované lokality, kde už boli zaznamenané relevantné informácie a je im priradené vyššia priorita (Guiyang a kol., 2004).

### **Algoritmus Shark-Search**

Tento algoritmus je dokonalejšou verziou, ktorá vznikla z Fish-Search algoritmu. Oproti Fish-Search má mnoho vylepšení, pretože dokáže presnejšie analyzovať relevantnosť dokumentov a s lepším odhadom. Relevancia susedných stránok môže byť presnejšie načítaná a analyzovaná, a môže prehľadávať webové stránky hlbšie v oblastiach, kde šanca na nájdenie relevantných informácií šetrí čas. Tieto ciele dosahuje tromi spôsobmi:

- hodnotením predkov, z ktorých sa dostal k aktuálnej stránke,
- určením priority na základe textu na stránke,
- mierou relevantnosti kontextu v okolí odkazu.

Toto všetko vykoná v rovnakom čase ako Fish-Search (Shrivastava a Sahai, 2019).

## **1.8 VYHĽADÁVACIA POLITIKA**

Webový crawler musí spĺňať rôzne úlohy a ciele, ktoré treba vykonávať s určitou dávkou opatrnosti. Taktiež zdroje, ktoré má k dispozícii musí využívať efektívne, vrátane šírky pásma siete, ktorá musí vykazovať vysoký stupeň paralelizmu bez preťaženia webového servera. Z tohto dôvodu je potrebné dodržiavať základnú vyhľadávaciu politiku. Medzi základné princípy patrí zásada výberu, zásada opätovnej návštevy, zásada zdvorilosti a zásada paralelného spracovania (Isuwar a Nath, 2013).

### **Politika výberu stránok**

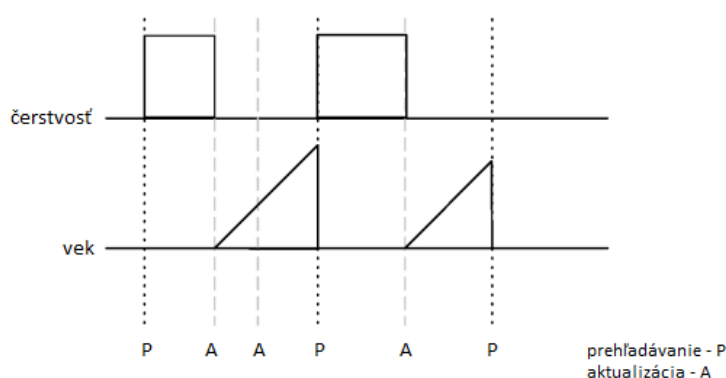
Politika výberu určuje podmienky, ktoré stránky sa majú sťahovať. Z výskumu, ktorý spracoval Gulli a Signorini (2015) vieme, že aj tie najlepšie nástroje dokážu zindexovať len 40 až 70% dostupných stránok. Z toho dôvodu je potrebné, aby nájdené stránky boli relevantné pre používateľa. Na základe tohto zistenia sa spracovali viaceré techniky uprednostňovania webových stránok využívajúce hodnotiace parametre



ako napr. kvalita a popularita odkazov a návštev. Najpopulárnejšími metódami sú PageRank, Adaptive PageRank, Trust Rank, HITS a OPIC (Narayan Das a kol., 2013).

### Politika opätovnej návštevy

Vzhľadom na veľkosť a dynamiku webu, je potrebné pre zachovanie aktuálnosti opätovne navštevovať už navštívené stránky. Politika opätovnej návštevy určuje časový interval, ako často môže crawler pristupovať a kontrolovať zmenené dáta na stránkach. Medzi najčastejšie používané metriky patrí čerstvosť, ktorá určuje zhodu s originálom a vek, ktorý hovorí o tom, aký starý je posledný záznam (Baeza-Yates a kol., 2002).



Obrázok 4 Vývoj veličín čerstvosti a veku<sup>4</sup>

### Zásada zdvorilosti

Crawler by bol schopný zahltiť webový server viacerými požiadavkami v krátkom, prípadne ho zablokovat' sťahovaním veľkých súborov. Aby nedochádzalo k tejto situácii, mal by crawler dodržiavať zásadu zdvorilosti, ktorá hovorí, ako často možno navštevovať rovnaký web. Jej hlavnou úlohou je zabrániť, aby nedochádzalo k preťaženiu serverov (Amudha a Phil, 2017).

### Princíp paralelizácie

Hlavnou úlohou tohto princípu je pomôcť koordinovať distribuované procesy prehľadávania. Amudha a Phil (2017) vo svojej práci uvádzajú, že cieľom paralelného crawlera je spúšťať viacero procesov súčasne, pričom dochádza k maximalizovanej rýchlosti sťahovania, minimalizuje sa režia z paralelizácie a zabráňuje sa opakovanému sťahovaniu tej istej stránky. Aby sa predišlo sťahovaniu tej istej stránky viac ako raz, systém vyžaduje zásadu priradovania nových adries URL objavených počas procesu

<sup>4</sup> Zdroj obrázok: <http://www.cs.cornell.edu/courses/cs4300/2013fa/lectures/thurs-sept-12-4pp.pdf>

prehľadávania. Bez nej by bol crawler schopný rovnakú adresu URL nájsť pomocou dvoch rôznych procesov.

## 1.9 ETIKA CRAWLOVANIA

Pri prehľadávaní je potrebné prihliadať na etické aspekty používania webových prehľadávačov. Taktiež je dôležité neustále zdôrazňovať rozdiel medzi legálnym a etickým využívaním týchto technológií, ako aj meniacu sa hranicu medzi etickým a neetickým správaním. Webové crawlery, ktoré automaticky vyhľadávajú a sťahujú webové stránky, sa stali nevyhnutnými pre modernú spoločnosť. Podľa Thelwalla a Stuarta (2006) je toto tvrdenie výsledkom nasledujúcich dôvodov:

- dôležitosť webu pre publikovanie a vyhľadávanie informácií,
- nevyhnutnosť používania vyhľadávacích nástrojov, napr. Google, na vyhľadávanie informácií na webe,
- spoliehanie sa vyhľadávacích nástrojov na webových crawlerov pri získavaní väčšiny ich nespracovaných surových dát.

Prvé crawlery vytvárali a používali výlučne počítačoví vedci, ktorí si boli vedomí toho, čo robia. Avšak internet už dávno neslúži len na vedecké účely. Bežní používatelia webu si väčšinou nevšimnú robotov, ktoré automaticky z internetu sťahujú informácie. Aj napriek tomu sú to potenciálne veľmi výkonné nástroje so schopnosťou spôsobovať problémy v sieti. Z toho dôvodu sú potrebné etické pravidlá pre používanie webových crawlerov. Hlavným súborom pravidiel pre fungovanie je protokol robots.txt. Ale je potrebné doplniť, že roboty môžu ignorovať súbor robots.txt. Hlavne malvérové roboty, ktoré prehľadávajú web kvôli bezpečnostným chybám, a zberače e-mailových adries. Pre skrytie informácií, je potrebné mať webovú stránku zabezpečenú nejakou formou autentifikácie.

Majitelia webových stránok používajú súbor robots.txt na poskytovanie pokynov a pravidiel o ich webovej stránke webovým robotom. Tento pojem sa taktiež nazýva ako protokol o vylúčení robotov. Predtým ako robot začne prechádzať celú webovú stránku, napr. <https://priklad.sk>, mal by skontrolovať <https://priklad.sk/robots.tx>. Tento súbor môže obsahovať nasledujúce kľúčové pojmy:

- User-agent,
- Disallow,
- Allow (nie je podporované všetkými vyhľadávačmi),
- Sitemap (voliteľná položka).

User-agent obsahuje meno robota, na ktorého sa vzťahuje nasledovný príkaz. Ak chceme určiť pravidlá pre všetkých naraz, použijeme symbol hviezdčky „\*“. Pokiaľ by sme chceli zakázať všetkým robotom pracovať na našej stránke, použijeme výraz „Disallow: /“. Taktiež je možné obmedziť konkrétny adresár, napr. „Disallow: /assets/“ (The Web Robots Pages, 2007). Opakom je príkaz Allow, ktorý povoľuje používanie robotmi konkrétnej časti. Sitemap, hovorí o tom, kde sa nachádza umiestnenie mapy webovej lokality. URL adresa sitemap musí mať presne stanovený názov domény. Súbor Sitemap sú dobrým spôsobom, ako určiť, ktorý obsah má Google indexovať, na rozdiel od toho, ktorý obsah môže alebo nemôže indexovať (Google developers, 2022).

Medzi najznámejšie a najčastejšie problémy, ktoré môžu webové crawlery vyvolať patrí: odmietnutie služby, zvýšené náklady na prevádzku, strata súkromia a porušenie autorských práv.

### **Odmietnutie služby (DoS)**

Nevedomky alebo cielene môže crawler svojimi nadmerne častými požiadavkami zahltiť server, ktorý začne brániť všetkým používateľom k prístupu na web. Takýto server môže pomaly reagovať na ostatných používateľov, čo oslabuje jeho primárny účel. K tomu, aby sa zamedzilo tomuto problému je potrebné zabezpečiť, aby crawler nekládol vysoké nároky na servery (Frankenfield, 2022).

### **Zvýšené náklady na prevádzku**

Každý, kto si zakladá stránku, si musí vopred premyslieť viacero dôležitých faktorov, ktoré ovplyvnia výšku jeho investície. Okrem iného je potrebné zaplatiť za šírku pásma, ktorá zodpovedá množstvu údajov, ktoré môže web preniesť cieľovým používateľom za určitý čas. Ak je stanovená hranica prekročená, majiteľ za to väčšinou musí zaplatiť, dokonca môže dôjsť až k zablokovaniu samotnej stránky (Thelwall a Stuart, 2006).

### **Strata súkromia a porušovanie autorských práv**

Tento problém je veľmi citlivý a názory naň sa líšia. Jednou z príčin je rýchly technologický postup, ktorému nestíhajú právne subjekty. Okrem toho sa zákony a názory líšia podľa krajín. Podľa tvrdenia Karunakarana (2018), sa v reálnom svete za verejné údaje považuje každá webová stránka alebo zdroj, ktorý nevyžaduje registráciu a je možné ju indexovať. V takom prípade by mohli zbierať roboty dáta zo všetkých voľne dostupných stránok. Špeciálnou záležitosťou sú sociálne siete,

ktoré obsahujú obrovské množstvo informácií a po registrácii sú dostupné takmer každému. K tejto téme sa vyjadril aj profesor Harvardskej univerzity, Laurence Tribe, ktorého výrok zaznamenal Karunakaran (2018) vo svojom článku. Počas súdneho pojednávania medzi spoločnosťami LinkedIn a hiQ Labs povedal: „Ak niekoho vylúčíte zo stránok ako LinkedIn, Facebook a Twitter, vylúčíte ho z modernej verzie mestského námestia.“ Tento výrok len potvrdzuje nejasnosť celej problematiky a cenu dát v dnešnej dobe. V tomto prípade firma s použitím crawlera zozbierala údaje o 500 miliónoch používateľoch z LinkedIn a predávala ich na internete. Aj keby to bolo úplne legálne, stále existujú etické zásady, prečo to nerobiť. Jedným z nich je ochrana osobných údajov, ktorá môže mať aj právne dôsledky. V Európskej únii sú používatelia chránení nariadením Európskeho parlamentu o GDPR. Podľa nariadenia existujú tri situácie, kedy je zhromažďovanie údajov občanov EÚ podľa zákona GDPR povolené:

- existuje legitímny dôvod na zhromažďovanie údajov, ktoré spadajú do kontextu vzťahu medzi klientom a spoločnosťou,
- údaje sa musia zbierať, aby sa zabránilo podvodom a bola zabezpečená bezpečnosť systému spoločnosti,
- spoločnosť chce zbierať údaje na účely priameho marketingu (Nariadenie Európskeho parlamentu a rady, 2016).

## 2 CIELE ZÁVEREČNEJ PRÁCE

Cieľom bakalárskej práce je analýza aktuálnych riešení pri navrhovaní webových crawlerov a implementácia webového crawlera pre vybranú oblasť. To znamená vytvorenie softvéru, ktorý zjednoduší vyhľadávanie potrebných informácií na webových stránkach a zobrazí usporiadané údaje podľa požiadaviek. Touto prácou by sme chceli ľuďom pomôcť efektívnejšie pracovať s veľkým množstvom informácií. Ciele a postupy praktickej časti vychádzajú z požiadaviek firmy DOMINANZ spol. s r. o.

Praktická časť má pozostávať zo serverovej a webovej časti. Používateľ bude pristupovať k systému pomocou webového rozhrania. Aplikácia bude verejne dostupná a bude fungovať v dvoch módoch; bez zadaného emailu a so zadaným emailom. Hlavným cieľom je vytvoriť web, kde bude používateľovi umožnené vyhľadávať RSS súbory pomocou internetového vyhľadávacieho enginu DuckDuckGo. Po zadaní kľúčových slov a filtračných nastavení služba vráti zoznam RSS súborov, ktoré budú podľa zadaných kritérií, pričom poskytne aj analýzu týchto linkov. Ak sa používateľ rozhodne neposkytnúť svoju emailovú adresu, bude môcť kľúčové slová zadávať len po jednom. Toto vyhľadávanie bude pracovať v obmedzenom režime a používateľovi bude zobrazovať len zopár prvých RSS súborov bez bližšej analýzy. Ak sa rozhodne poskytnúť svoj email, dostane sa do módu, kde mu bude umožnené zadávať viac kľúčových slov, nastavovať filtre a výsledok mu bude po vykonaní požiadavky poslaný na email. Pre oba módy bude prístupná možnosť overiť, či existuje pre zadanú webovú stránku RSS súbor. Ak existuje, tak sa zobrazí. Údaje je potrebné ukladať do súborov typu JSON.

### PODCIELE ALEBO ČIASTKOVÉ CIELE:

- návrh webovej aplikácie,
- vytvorenie webovej aplikácie,
- test funkcionality,
- analýza dát.

### 3 METÓDY RIEŠENIA PROJEKTU

K tomu, aby sme mohli vytvoriť crawler hľadajúci RSS súbory, je dôležité oboznámiť sa s pojmom RSS. Ide o typ formátu, ktorý v zjednodušenej verzii zobrazuje správy, obsah stránok, podcasty a články blogov. Každý takýto súbor obsahuje metadáta, ktoré pomáhajú zachytávať dôležité základné informácie o webe. V hlavičke sa väčšinou zobrazuje názov webu, popis webu, autor, odkaz URL, jazyk, prípadne aj informácia o autorských právach. Treba však poznamenať, že nie všetky kľúče sú povinné a verzie RSS sa medzi sebou môžu líšiť.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <title>Pravda - Správy</title>
    <link>https://www.pravda.sk</link>
    <description>správy, ktorým môžete veriť</description>
    <language>sk</language>
    <copyright>(c) OUR MEDIA SR a. s.</copyright>
    <pubDate>2022-03-03T18:04:40+01:00</pubDate>
    <ttl>5</ttl>
    <image>
      <url>http://ipravda.sk/res/portal2011/Pravda_logo.gif</url>
      <link>https://www.pravda.sk</link>
      <title>Pravda</title>
      <width>69</width>
      <height>22</height>
    </image>
    <docs>http://blogs.law.harvard.edu/tech/rss</docs>
    <lastBuildDate>2022-03-03T05:00:00+01:00</lastBuildDate>
  </channel>
  <item>
    <title>ONLINE: Mariupol volá o pomoc, Cherson padol. Ukrajinci zabili veliteľa Specnaz</title>
    <link>https://spravy.pravda.sk/svet/clanok/618771-online-z-ukrajiny-vojna-vyhnila-uz-milion-ludi/?utm_source=pravda&utm_medium=rss&utm_campaign=rss</link>
    <description>Kľúčové mesto na juhovýchode Ukrajiny je „obklúčené“ ruskými silami a čelí humanitárnej kríze.</description>
    <pubDate>2022-03-03T05:00:00+01:00</pubDate>
    <guid isPermalink="false">1e67b908447e5593d73c1809bd7a513e</guid>
    <category>Svet</category>
    <author>ČTK, SITA, TASR, Pravda</author>
    <enclosure url="https://ipravda.sk/res/2022/03/03/thumbs/kyjev-strednaw.jpg" length="" type="image/jpeg"/>
    <comments>https://debata.pravda.sk/debata/618771-online-z-ukrajiny-vojna-vyhnila-uz-milion-ludi/</comments>
  </item>
</rss>
```

Obrázok 5 Príklad RSS súboru denníka Pravda<sup>5</sup>

#### 3.1. POUŽITÉ TECHNOLOGIE

V tejto časti popisujeme technológie využívané pri implementácii a vývoji aplikácie. Základ celej aplikácie je postavený na jazyku Python, s podporou knižníc DuckDuckGo a Flask, a frameworku Angular.

##### 3.1.1 Python

Python je veľmi atraktívny, skriptovací, objektovo orientovaný programovací jazyk. Vývoj aplikácií s jeho pomocou prebieha rýchlo a dynamicky. Jeho výhodou je podpora modulov a balíkov (Python, 2001). V našom prípade sme pre vývoj crawleru použili verziu Python 3.9.1. v prostredí programu Visual Studio Code.

Prehľad nami inštalovaných knižníc:

- beautifulsoup4 (4.10.0),
- certifi (2021.10.8),

<sup>5</sup> Zdroj obrázok: <https://spravy.pravda.sk/rss/xml/>

- duckduckgo-search (0.7),
- feedparser (6.0.8),
- Flask (2.0.2),
- langdetect (1.0.9),
- lxml (4.6.3),
- requests (2.26.0),
- urllib3 (1.26.7).

### 3.1.1 DuckDuckGo

DuckDuckGo je spoločnosť zaoberajúca sa ochranou súkromia na internete, ktorá vytvorila vyhľadávaciu službu, umožňujúcu kontrolovať osobné údaje každého používateľa podľa jeho predstáv. Knižnica duckduckgo-search, k získaniu výsledkov používa API tejto spoločnosti. Využitie tejto knižnice pre získanie URL adries v praktickej časti, bola jedna z požiadaviek firmy z praxe. Pre naše potreby postačuje využívať funkciu ddg, ktorá ako vstupné parametre ponúka zadať zoznam kľúčových slov, oblasť podľa jazyka (region), nastaviť úroveň bezpečnosti vyhľadávania (safesearch), aktuálnosť (time) a maximálny počet výsledkov (max\_results). Aby nedošlo k blokovaniu, bolo potrebné zabezpečiť časové rozostupy volaní tejto služby; v našom prípade sme zvolili interval 2 sekúnd. Možnosť výberu regiónu, maximálny počet prejdenných stránok a aktuálnosť je umožnené na základe požiadaviek len používateľovi, ktorý zadal emailovú adresu. Avšak maximálny počet nesmie presiahnuť nami stanovenú hornú hranicu 100 stránok. Bezpečnosť vyhľadávania je pre všetkých nastavená na najvyššiu úroveň.

### 3.1.2 Flask

Flask je jednou z najpoužívanejších knižníc pri tvorbe webovej aplikácie v Pythone. Na základe obľúbenosti a jednoduchšej implementácií sme sa rozhodli použiť Flask pre potreby komunikácie s frontendom. Hlavné parametre, ktoré je potrebné nastaviť sú IP adresa hostiteľa a port. Tieto parametre sú uložené v súbore constants.py pod názvom SRV\_IP a SRV\_PORT. Počas vývoja bolo potrebné riešiť viacero prekážok spojených so serverom. Jedným z nich je nastavenie CORS politiky. Musíme nastaviť v hlavičke odpovede Access-Control-Allow. Tento mechanizmus slúži, aby sa prehliadač a server dohodli na tom, z ktorej domény je povolené volať služby servera. Toto nastavenie je definované pod názvom ALLOW\_ORIGIN. Službám

spracovávajúcim úlohy, bol nastavený maximálny počet volaní raz za sekundu a celkovo 100 za deň. Životnosť relácie z dôvodu, aby sa zamedzilo preťaženiu, bola nastavená na 20 minút. Pre účely zasielania výsledkov, bol dočasne vytvorený účet na platforme Gmail. Aby bola zabezpečená SMTP komunikácia, bol zadefinovaný port, adresa a prihlasovacie údaje ku kontu (Obrázok 6).

```
class constants:
    REGION='region'
    MAXIMUM="maximum"
    EMAIL="email"
    TIME = 'time'
    KEYS="keys"
    KEY="key"

    VERSION=1.0
    UTF_ENCODING='utf-8'

    MAX_RESULTS=100
    MIN_RESULTS=5
    PERMANENT_SESSION_LIFETIME=20

    ALLOW_ORIGIN = 'http://localhost:4200'

    SMTP_PORT = 465
    SMTP_SERVER = "smtp.gmail.com"
    SENDER_EMAIL = "rss.dominanz@gmail.com"
    SENDER_PASSWORD = "[REDACTED]"

    SRV_IP = '0.0.0.0'
    SRV_PORT = '5000'
```

Obrázok 6 Ukážka súboru constants.py<sup>6</sup>

### 3.1.3 Angular

Angular je moderný framework pre tvorbu webových aplikácií, ktorý poskytuje komplexnú sadu nástrojov a možností. Vytvorila ho spoločnosť Google. Pri vývoji sa využíva HTML a TypeScript. V našom prípade sme zvolili verziu Angular 11 a TypeScript 4. Okrem klasickej sady Node Modules sme pridali ďalšie 4 moduly:

- @ng-bootstrap/ng-bootstrap,
- @types/file-saver,
- angular-animations,
- bootstrap,
- file-saver,
- ngx-spinner,
- xlsx.

V súbore environment je potrebné konfigurovať adresu a port servera.

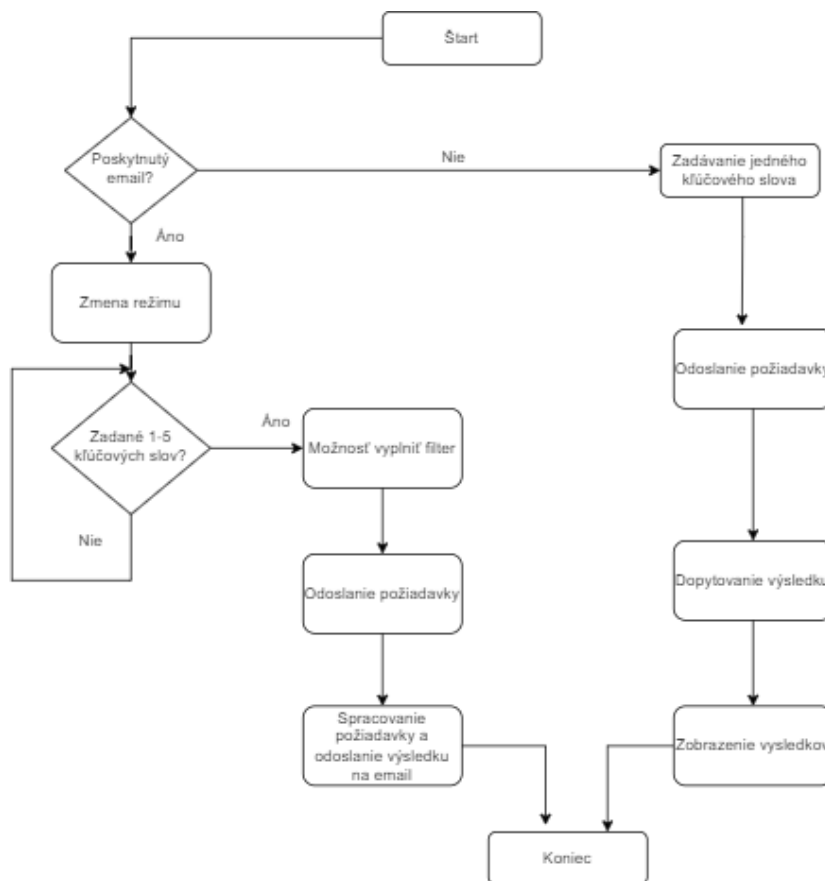
---

<sup>6</sup> Zdroj obrázok: Autor práce



## 4 VÝSLEDKY RIEŠENIA A ICH ZHODNOTENIE

V tejto časti sme popísali cestu k dosiahnutiu stanovených cieľov. Pred tvorbou každého projektu je potrebné analyzovať požiadavky, ktoré sme spísali v kapitole Ciele záverečnej práce.



Obrázok 7 Diagram vytvorený pomocou Draw.io<sup>7</sup>

### 4.1 NÁVRH WEBOVEJ APLIKÁCIE

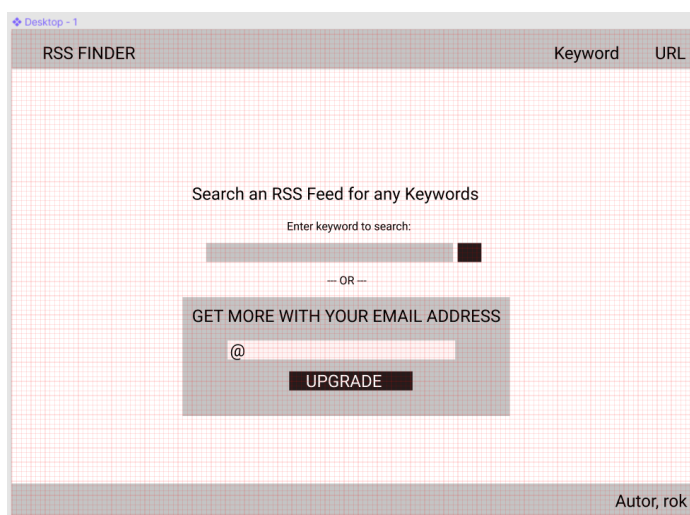
Návrh prototypu vznikol v spolupráci s firmou z praxe, pomocou softvérov Draw.io a Figma. V návrhu boli zohľadnené všetky požiadavky. Pre vývoj sme navrhli šesť farieb, ktoré boli vsadené do dizajnu, ktorý je spracovaný v súbore styles.scss.

<sup>7</sup> Zdroj obrázok: Autor práce



Obrázok 8 Paleta najčastejších farieb<sup>8</sup>

Aplikácia sa mala skladať z hlavnej lišty s dvoma sekciami, dolnej lišty s informáciou o autorských právach a centrálnej časti, kde sa má zobrazovať hlavný obsah. V časti Keyword malo byť umožnené používateľovi vyhľadávať RSS súbory na základe kľúčového slova. Ak by sa rozhodol uviesť svoj email, sprístupnili by sa mu rozšírené možnosti vyhľadávania s maximálne piatimi kľúčovými slovami súčasne. Po dokončení procesu by bol na uvedený email zaslaný odkaz na výsledky. V sekcii URL by bolo zabezpečené overovanie existencie RSS súboru pre konkrétnu stránku. Podľa požiadavky firmy bola celá aplikácia navrhnutá tak, aby sa údaje ukladali do súborov s formátom JSON.



Obrázok 9 Návrh úvodnej obrazovky v softvéri Figma<sup>9</sup>

## 4.1 IMPLEMENTÁCIA

Po dokončení návrhu, začala fáza implementácie návrhu. Samotná realizácia prebiehala iteratívno-inkrementálnou formou. Jadro serverovej časti sme vsadili do súboru app.py. Pomocou webového rámca Flask bol vypracovaný zoznam volaní, ktoré boli následne implementované:

- POST /setUser,
- GET /getRss,

<sup>8</sup> Zdroj obrázok: Autor práce

<sup>9</sup> Zdroj obrázok: Autor práce

- GET /task,
- GET /isCompleted,
- POST /tasks,
- GET /detail,
- GET /getRegions,
- GET /getPeriods.

V aplikácii bolo zakomponované zaznamenávanie aktivity pomocou logovacej knižnice do súboru records.log. Obmedzovanie počtu žiadostí servera sme dosiahli použitím knižnice Limiter. Tento rámec bol nastavený tak, aby počet volaní nepresiahol 100 za jeden deň a služby neboli volané viac ako jedenkrát za sekundu.

```

32 limiter = Limiter(
33     app,
34     key_func=get_remote_address,
35     default_limits=["100 per day"]
36 )
37
38 @app.post("/setUser")
39 @limiter.limit("1/second", override_defaults=False)

```

Obrázok 10 Nastavenie limitera<sup>10</sup>

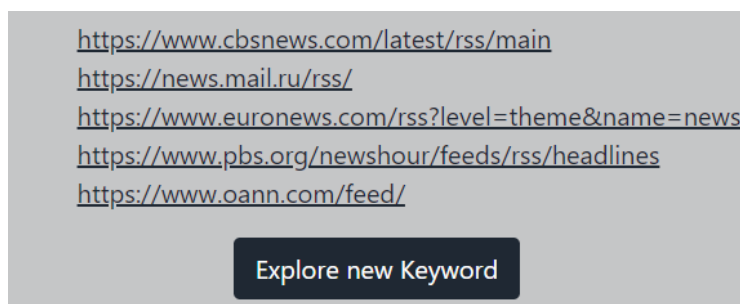
Vyhľadávanie RSS bolo naprogramované tak, aby aplikácia zahŕňala dva stavy; s emailom a bez emailovej adresy. Používateľ má preto možnosť zadať svoj email odoslaním požiadavky POST /setUser, čím získa viaceré výhody. Server po prijatí prečíta parameter email a ukladá ho do súboru emails.json. Tento parameter sa taktiež nastaví aj v SessionStorage. Ukončením relácie v prehliadači zaniká. Pomocou podmienky sme zabezpečili, aby nedochádzalo k duplicite uložených emailových adries.

Na hlavnej obrazovke Keyword pre jednoduchý mód "bez adresy", bola vytvorená služba GET /task s parametrom key, v ktorej je potrebné odosielať vyplnený kľúč. Pre spracovávanie hodnoty kľúča sme aplikovali proces ukladania požiadaviek do poradovníka. Miesto celej požiadavky sa ukladá hash s kľúčovým slovom. Vytvoriť takýto mechanizmus bolo potrebné, aby server vedel v krátkom časovom intervale spracovať každú požiadavku. Vyhľadávanie RSS na webe je časovo

---

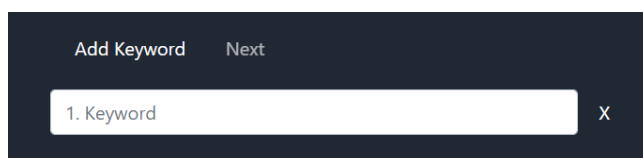
<sup>10</sup> Zdroj obrázok: Autor práce

náročná operácia. Preto, ak nie je k dispozícii email, bolo potrebné eliminovať počet prehľadaných webov. Funkcia worker zabezpečuje postupné vyberanie požiadaviek zo zásobníka. Keď sa požiadavka dostane na rad, proces sa pozastaví na 2 sekundy pre dodržanie odporúčaného počtu žiadostí knižnice DuckDuckGo. Na strane používateľa sa spustí načítavanie. V určitých intervaloch dochádza k automatickému dopytovaniu GET /isCompleted, či už bola požiadavka vykonaná. Po získaní zodpovedajúcich odkazov sa spustí hľadanie RSS súborov. Proces končí, ak bolo nájdených päť súborov alebo uplynula doba 30 sekúnd. Výsledky sa zobrazujú v hlavnej časti a sú zobrazované na základe zhody hashu.



Obrázok 11 Nájdené RSS pre slovo „news“<sup>11</sup>

V tomto móde sa používajú predvolené nastavenia pri vyhľadávaní bez bližšieho špecifikovania požiadaviek. V prípade zadania emailovej adresy sme pridali možnosť zadávania viacerých kľúčových slov súčasne. Túto funkcionality sme postavili na dynamickom formulári, ktorý je možné odoslať len ak je zadané aspoň jedno kľúčové slovo. Maximálny počet kľúčových slov bol stanovený na päť vyhľadání.



Obrázok 12 Ukážka dynamického formulára<sup>12</sup>

Kliknutím na tlačidlo Next, sa sprístupní obrazovka pre filtrovanie. Vsadili sme do nej dva rozbaľovacie komponenty, ktoré sa naplňajú volaním služieb GET /getPeriods a /getRegions, a jeden číselný pre určenie maximálneho počtu prehľadaných stránok.

---

<sup>11</sup> Zdroj obrázok: Autor práce

<sup>12</sup> Zdroj obrázok: Autor práce

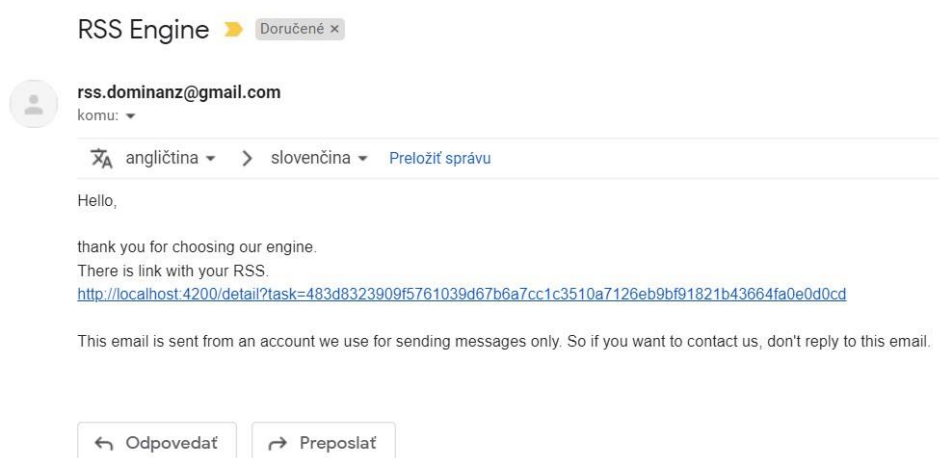
Obrázok 13 Ukážka filtrovania v režime „s emailom“<sup>13</sup>

Tieto služby získavajú požadované dáta a posielajú ich vo formáte JSON. Údaje v nich sú prispôsobené pre korektné fungovanie knižnice DuckDuckGo. S cieľom obmedziť vkladanie väčšej hodnoty ako 100, sme vytvorili direktívu maximum. Splnením všetkých kritérií a odoslaním požiadavky POST /tasks s filtrom, server spracuje telo požiadavky a odosiela ho do nového vlákna, pričom je používateľ informovaný, že výsledky mu budú zaslané na zadaný email. Vo vlákne sa najprv overia nastavenia filtra, maximálny počet prehľadaných odkazov na jeden kľúčový výraz je 100. Z týchto výrazov sa postupne získavajú odkazy prostredníctvom DuckDuckGo a vypočítava sa hash pre potreby zobrazenia výsledkov. Pri vyhľadávaní sa zohľadňujú požiadavky na región, časový interval a maximálny počet prehľadaných stránok. Následne sa cyklicky sťahuje obsah každej URL adresy, v ktorom prebehne lokalizácia umiestnenia RSS súborov danej webovej lokality. Tento proces prebieha prostredníctvom inštalovanej knižnice BeautifulSoup vo funkcii get\_rss(url). V obsahu webu sa hľadá značka link typu application/rss+xml. Pred ďalším spracovávaním je RSS súbor overený, či vracia status 200. Prostredníctvom podmienky sme sa snažili dosiahnuť, aby nedochádzalo k duplikovaniu už nájdených lokalít RSS zdrojov. Pre korektné spracovanie obsahu RSS sme sa rozhodli získavať ho pomocou knižnice feedparser. Ako sme už spomenuli, štruktúra RSS protokolov sa môže líšiť. Jednou z požiadaviek firmy bola snaha približne detegovať jazyk stránky. Detekciu sme implementovali v časti detect\_language, pričom sme sa snažili určovať jazyk s najväčšou pravdepodobnosťou. Pri detekcii sme využili langdetect pre zistenie jazyka popisu webu a popisu prvého článku. Trojicu sme doplnili o voliteľný príznak jazyka z obsahu RSS, ktorý však nie vždy zodpovedá skutočnosti. Po ich získaní dochádza

---

<sup>13</sup> Zdroj obrázok: Autor práce

k porovnávaniu a hľadaniu, čo najväčšej zhody medzi nimi. Na základe toho sa predikuje jazyk pre konkrétny web. Potom sa URL adresa, jazyk stránky a kľúčový výraz zhromažďujú do zoznamu. Po spracovaní všetkých kľúčových výrazov a prejení všetkých odkazov je list uložený spolu s kontaktným emailom, hashom a časom požiadavky do súboru results.json. Z hashu sme zabezpečili, aby sa vytvoril odkaz, na ktorom sa budú zobrazovať výsledky. V kóde sme definovali šablónu a predmet správy. Odkaz sa vkladá do textu, ktorý je následne zaslaný na emailovú adresu prostredníctvom protokolu SMTP. Email je odoslaný pomocou služby Gmail, v ktorej sme vytvorili účet, len pre odosielanie výsledkov. Pre potreby vývoja sme výsledky zobrazovali na lokalite localhost.



Obrázok 14 Ukážka emailu s výsledkami<sup>14</sup>

Po otvorení odkazu sa volá služba GET /detail s konkrétnou hodnotou hashu. Ak sa našiel aspoň jeden RSS súbor, výsledky sa zobrazia v tabuľke. Používateľ má okrem zobrazenia možnosť exportovať výsledky do súboru rss.xlsx.

---

<sup>14</sup> Zdroj obrázok: Autor práce

Timestamp request: March 7, 2022, 7:03 PM GMT+01:00		Export	
Nr.	URL	Language	Keyword
1	<a href="https://www.theguardian.com/world/ukraine/rss">https://www.theguardian.com/world/ukraine/rss</a>	en	ukraine
2	<a href="https://wikitravel.org/rss/english.rss">https://wikitravel.org/rss/english.rss</a>	en	ukraine
3	<a href="https://southfront.org/feed/">https://southfront.org/feed/</a>	en	ukraine
4	<a href="https://www.ukrinform.net/rss/block-lastnews">https://www.ukrinform.net/rss/block-lastnews</a>	en	ukraine
5	<a href="https://nypost.com/tag/ukraine/feed/">https://nypost.com/tag/ukraine/feed/</a>	en	ukraine

Obrázok 15 Zobrazenie výsledkov<sup>15</sup>

Obrazovku „URL“ sme vytvorili pre všetkých používateľov rovnakú. V jej hlavnej časti sa nachádza textový komponent, do ktorého možno zadávať URL adresu. Po odoslaní požiadavky GET /getRss sa spustí proces hľadania RSS. Ak server našiel RSS, vráti naň odkaz, ktorý sa následne zobrazí na obrazovke. V prípade, že sa ho nepodarí nájsť, vráca hodnotu null a informuje o tom používateľa.

Enter url to search:

http(s)://

Submit

RSS: <https://www.info.sk/rss/>

Obrázok 16 Ukážka overeného RSS z adresy webu<sup>16</sup>

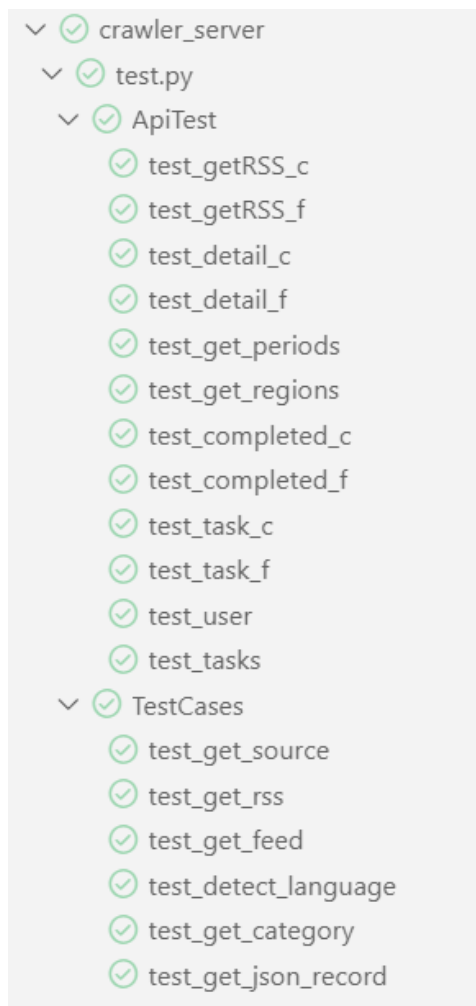
## 4.2 TESTOVANIE A RIEŠENIE PROBLÉMOV

Pre testovanie základných funkcií sme si vybrali rámec unittest, v ktorom sme vytvorili automatické testy. Pomocou nich sme preverili funkcie v súboroch main.py a app.py. Na začiatku bolo potrebné importovať knižnicu. Následne sme zadefinovali triedu TestCases, v ktorej sme vytvorili 6 testov a triedu ApiTest, pre kontrolu webových služieb. Spúšťanie testov sme vykonali veľmi jednoducho v programe VS Code, ktorý v bočnej lište poskytuje intuitívnu časť Testing. Testovanie prebehlo na korektných aj chybných vstupoch. Triedu TestCases bolo možné spustiť naraz v časti Testing alebo zadaním „python -m unittest test.TestCases“ do príkazového riadku. Avšak, ak sme sa rozhodli spustiť naraz všetky testy v ApiTest, server vrátil chybu 429

<sup>15</sup> Zdroj obrázok: Autor práce

<sup>16</sup> Zdroj obrázok: Autor práce

pre blokovanie nadmerného množstva požiadaviek. Tým sme zároveň otestovali náš limiter. Pre dosiahnutie korektných výsledkov sme testy v tejto triede spúšťali postupne, ale úspešne. Keď sme sa rozhodli testovať získané dáta, zistili sme, že približne 5,7% nájdených webov už neexistovalo alebo nebolo dostupných. Tento test potvrdil obrovskú dynamiku webu, lebo stránky nielen vznikajú, ale aj zanikajú. Testovanie webovej časti sme vykonali manuálne.



Obrázok 17 Výsledky testov<sup>17</sup>

Počas vývoja sme sa stretli s dvoma problémami, ktoré bolo nutné vyriešiť. Prvým z nich bol problém s overovaním certifikátu, ktorý vznikol v dôsledku toho, že dôveryhodné certifikáty v systéme už modul Python nepoužíval ako predvolené. Miesto toho sa používali certifikáty dodané prostredníctvom balíka certifi. Problém sme odstránili aktualizáciou tohto balíka, spustením príkazu `pip install --upgrade certifi`.

---

<sup>17</sup> Zdroj obrázok: Autor práce



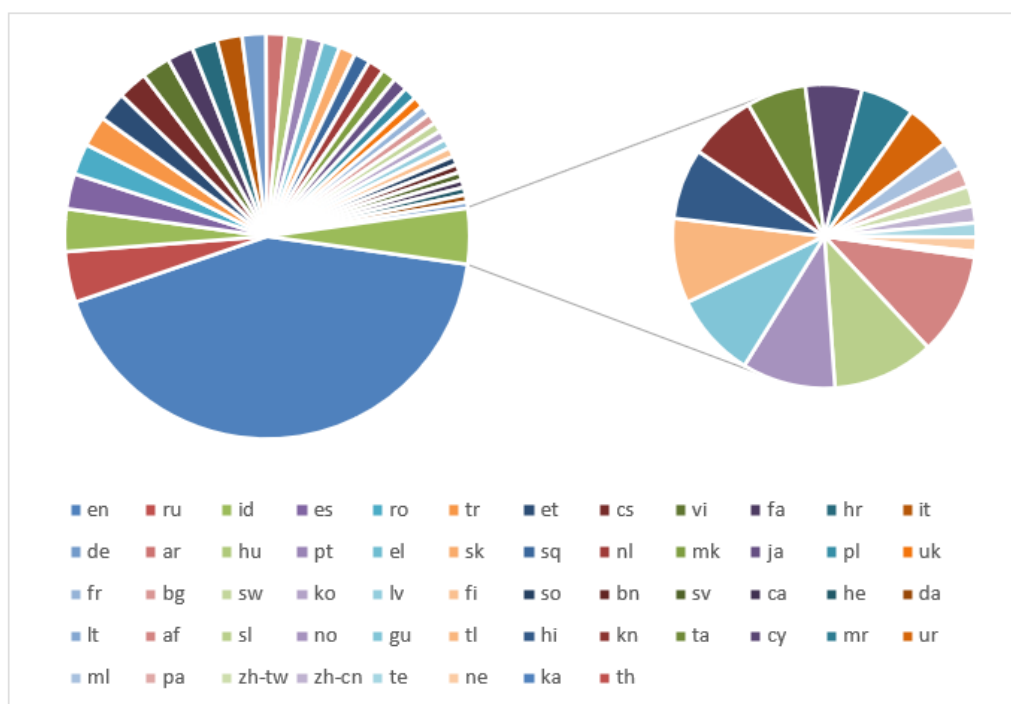
Druhý problém mal bezpečnostný dôvod, pričom súvisel s webovým prehliadačom a nastavením servera. Vzhľadom k tomu, že chceme, aby na náš server prichádzali požiadavky výhradne len z nášho webu, bolo potrebné nastaviť CORS. Ide o techniku, ktorá využíva hlavičky protokolu HTTP a umožňuje aplikáciám v prehliadači pristupovať k údajom. Náš server vo vývojárskom režime sa spúšťal pod súkromnou IP adresou a portom 5000, ale webová aplikácia na adrese localhost a porte 4200. Preto sme sa rozhodli do hlavičky odpovede vložiť nastaviteľnú konštantu `ALLOW_ORIGIN` zo súboru `constants.py`.

### **4.3 ZÍSKANIE A ANALÝZA DÁT**

Pre zozbieranie štatistických dát sme sa rozhodli modifikovať nášho crawlera. Dočasne sme doplnili ukladanie jedinečných RSS do súboru `rss.json`, kde sme už okrem spomínaných vlastností hľadali kategórie a domény najvyššej úrovne (TLD). Do crawlera sme vložili 542 výrazov, nad ktorými prebehol proces prehľadávania. Aby sme získali široké spektrum výsledkov a zabezpečili ich pestrosť, kľúčové slová boli v 104 rôznych jazykoch. Výrazy boli náhodne vybrané z viacerých oblastí, vrátane medicíny, ekonomiky, noviniek, vojny, nerastných surovín, životného štýlu, či technológií. Celkový počet prejdených stránok sme nezaznamenávali. Analýzu sme vytvorili v programe Excel. Podarilo sa nám zozbierať 7524 RSS súborov v 56 jazykoch. Znamená to, že na jeden výraz pripadá necelých 14 nových jedinečných RSS. Najväčší podiel tvorili weby v anglickom (43%), ruskom (4%), indonézskom (3%), španielskom (3%) a rumunskom jazyku (2%). Aj napriek tomu, že náš odhad jazyka bol len približný, zodpovedal trendom a obľúbenosti anglického jazyka vo svete internetu (Statista, 2020).

Tabuľka 1 Tabuľka desiatich najčastejších detegovaných jazykov<sup>18</sup>

Jazyk stránky	Kód jazyka	Počet stránok
angličtina	en	3200
ruština	ru	305
indonézština	id	255
španielčina	es	212
rumunčina	ro	186
turečtina	tr	183
estónčina	et	177
čeština	cs	176
vietnamčina	vi	172
perzština	fa	158

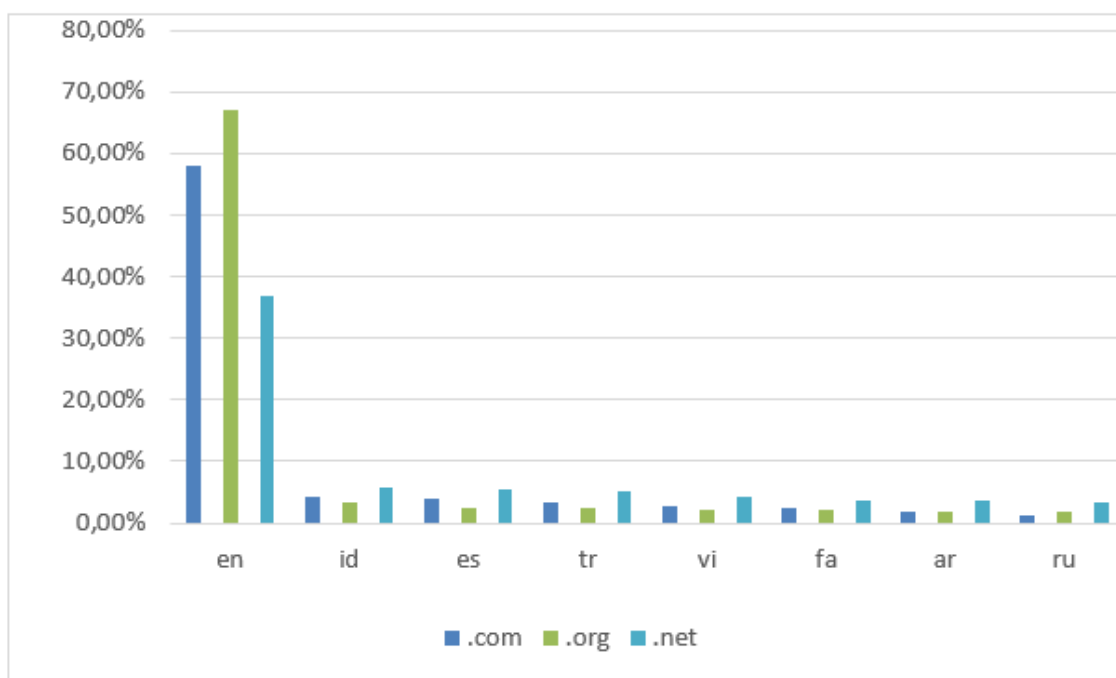


Graf 1 Podiel jazykov na stránkach<sup>19</sup>

Spomedzi domén najvyššej úrovne mali najväčšie zastúpenie .com (48,76%), .org (4,89%) a .net (3,99%). Na koniec sme vytvorili kontingenčnú tabuľku, v ktorej sme porovnali vzťah medzi TLD a jazykom.

<sup>18</sup> Zdroj tabuľka: Autor práce

<sup>19</sup> Zdroj graf: Autor práce



*Graf 2 Výstup z kontingenčnej tabuľky troch najpočetnejších TLD<sup>20</sup>*

<sup>20</sup> Zdroj graf: Autor práce

## ZÁVER

Hlavným cieľom našej bakalárskej práce bolo navrhnúť webového crawlera. Cieľ sme splnili aj vďaka vytvoreniu návrhu pomocou externých softvérov. V našej práci sa nám ho podarilo navrhnúť, ale aj implementovať, pretestovať a získať výstupné dáta. Preto si myslíme, že sa nám podarilo naplniť aj naše stanovené čiastkové ciele. Vzhľadom na to, že je náš crawler závislý od vstupných nastavení a vhodných kľúčových výrazov, tak aj výsledky sme museli brať s rezervou. Aj napriek tomu medzi výhody zaradujeme, že zozbierané dáta, boli ďalej využité v praxi firmou Dominanz.

Do budúca v našej aplikácii vidíme možnosť použitia emailového klienta, prípadne implementovať ukladanie RSS a pri rovnakých vyhľadávaniach opätovne používať už nájdené RSS súbory. Zaujímavé by bolo aj ukladanie jednotlivých RSS a porovnávanie s predchádzajúcimi verziami. Vzhľadom na popularitu čítania článkov bez reklamy a iných rušivých elementov by stálo za zmienku doplnenie vlastného zobrazovača RSS.

Celkovo výslednú aplikáciu hodnotíme kladne, veľa sme sa počas nej naučili, najmä v oblasti prehľadávania webu. Myslíme si, že má potenciálne využitie nielen pre študentov pri hľadaní informácií, ale aj pre širokú verejnosť.

## ZOZNAM BIBLIOGRAFICKÝCH ODKAZOV

AHUJA, M., SINGH, J., VARNICA, B. 2014. *Web Crawler: Extracting the Web Data*. International Journal of Computer Trends and Technology (IJCTT), vol. 13, 2014. ISSN: 2231-2803.

ALLINSON, M. 2021. *How has Data Become the World's Most Valuable Commodity?* Robotics & Automation News – Market trends and business perspectives [online] 2021-07-22. [cit. 2022-02-01]. Dostupné na internete: <<https://roboticsandautomationnews.com/2021/07/22/how-has-data-become-the-worlds-most-valuable-commodity/44267/>>.

AMUDHA, S., PHIL, M. 2017. *Web Crawler for Mining Web Data*. International Research Journal of Engineering and Technology (IRJET), vol. 4, 2017. ISSN 2395-0072.

BAEZA-YATES, R., SAINT-JEAN, F., CASTILLO, C. 2002. *Web Structure, Dynamics and Page Quality*. 117-130. 10.1007/3-540-45735-6\_12.

BROWN, J. S., DUGUID P. 2000. *The Social Life of Information*. Boston: Harvard Business School Press, Boston, MA, 2000. ISBN 0 875 847625.

CHAITRA, G., DEEPTHI V., VIDYASHREE, K.P., RAJINI, S. 2020. *A Study on Different Types of Web Crawlers*. In: Choudhury S., Mishra R., Mishra R., Kumar A. (eds) *Intelligent Communication, Control and Devices. Advances in Intelligent Systems and Computing*, vol. 989. Springer, Singapore. [online] 2020. [cit. 2022-02-18]. Dostupné na internete: <[https://doi.org/10.1007/978-981-13-8618-3\\_80](https://doi.org/10.1007/978-981-13-8618-3_80)>.

FRANKENFIELD, J. 2022. *Denial-of-Service (DoS) Attack*. Investopedia.com [online] 2022-01-26. [cit. 2022-02-21]. Dostupné na internete: <<https://www.investopedia.com/terms/d/denial-service-attack-dos.asp>>.

GOOGLE DEVELOPERS, 2022. *Create and submit a robots.txt file* [online] 2022. [cit. 2022-02-28]. Dostupné na internete: <<https://developers.google.com/search/docs/advanced/robots/create-robots-txt>>.

GUIYANG, S., JIANHUA, L., YINGHUA, M., SHENGHONG, L., JUPING, S. 2004. *New Focused Crawling Algorithm*. Department of Electronic Engineering,

- Shanghai Jiaotong University, Shanghai 200030, P. R. China Vol.16, No.1 , 2005, pp. 199-203.
- GULLI, A., SIGNORINI, A. 2005. *The indexable web is more than 11.5 billion pages*. Special interest tracks and posters of the 14th international conference on World Wide Web. ACM Press. pp. 902–903, 2005.
- GUPTA, A., ANAND, P. 2015. *Focused web crawlers and its approaches*. In: 2015 1st International Conference on Futuristic Trends on Computational Analysis and Knowledge Management ABLAZE 2015, pp. 619–622, 2015.
- HAMID, D., HASHEM, S. 2016. *Web Pages Retrieval by Using Proposed Focused Crawler*. Journal of Al-Nahrain University-Science. 19. 154-164.  
10.22401/JNUS.19.2.20. Dostupné na internete:  
<<https://www.cloudflare.com/learning/bots/what-is-a-web-crawler/>>.
- ISWARY, R., NATH, K. 2013. *WEB CRAWLER*. International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 10, ISSN: 2319-5940.
- KARUNAKARAN, K. 2018. *Can businesses access any public data in the web for their use?* Scrapeworks. [online] 2018-02-21. [cit. 2022-02-28]. Dostupné na internete: <<https://scrape.works/blog/can-businesses-access-any-public-data-in-the-web-for-their-use/>>.
- NARAYAN, N., KUMAR, E., SHEETAL, S. 2013. *Approaches of Page Ranking Algorithms: Review*. International Journal of Computer Applications. 82. 31-38. 10.5120/14090-2094.
- NARIADENIE EURÓPSKEHO PARLAMENTU A RADY (EÚ) 2016/679 | Úrad na ochranu osobných údajov Slovenskej republiky. NARIADENIE EURÓPSKEHO PARLAMENTU A RADY (EÚ) 2016/679 z 27. apríla 2016 o ochrane fyzických osôb pri spracúvaní osobných údajov a o voľnom pohybe takýchto údajov, ktorým sa zrušuje smernica 95/46/ES [online] 2016-05-04. [cit. 2022-03-02]. Dostupné na internete: <<https://eur-lex.europa.eu/legal-content/SK/TXT/?qid=1559210115221&uri=CELEX:32016R0679#d1e1883-1-1>>.

- PEREZ, M. 2021. *Web Scraping Techniques: How to Scrape Data from the Internet*. ParseHub Blog [online] 2021-04-21. [cit. 2021-10-22]. Dostupné na internete: <<https://www.parsehub.com/blog/web-scraping-techniques/>>.
- PICOT, J. 2016. *An introduction to web crawlers - OnCrawl*. Oncrawl | Enterprise Technical & Data SEO Platform for Smarter SEO [online] 2016-03-08. [cit. 2022-02-01]. Dostupné na internete: <<https://www.oncrawl.com/technical-seo/introduction-web-crawler/>>.
- PYTHON, 2001. *What is Python? Executive Summary*. Python.org. Welcome to Python.org [online] 2001. [cit. 2022-03-05]. Dostupné na internete: <<https://www.python.org/doc/essays/blurb/>>.
- RADWARE, 2017. *What does Scraping, Crawling and Indexing*. [online] 2017-01-17 [cit. 2021-10-22]. Dostupné na internete: <<https://www.shieldsquare.com/what-does-scraping-crawling-and-indexing-mean/>>.
- REINSEL, D., GANTZ, J., RYDNING, J. 2018. *The Digitalization of the World – From Edge to Core*. IDC White Paper [online]. 2018. [cit. 2022-02-01]. Dostupné na internete: <<https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf>>.
- SHRIVASTAVA, V., SAHAI, P. 2019. *Comparative Analysis of Web Crawling Algorithms for Improvement in Web Crawler*. Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur - India, [online] 2019-02-23. [cit. 2022-03-02]. Dostupné na internete: <<https://ssrn.com/abstract=3356275>>.
- STATISTA, 2020. *Internet: most common languages*. Statista - The Statistics Portal for Market Data, Market Research and Market Studies [online]. 2022. [cit. 2022-03-20]. Dostupné na internete: <<https://www.statista.com/statistics/262946/share-of-the-most-common-languages-on-the-internet/>>.
- ŠVEC, P., REICHEL, J., KUNA, P. 2020. *Princípy počítačových sietí*. Nitra: Univerzita Konštantína Filozofa v Nitre, 2020. 209 s. ISBN 978-80-558-1582-4.
- TECHOPEDIA, 2012. *What is the Web? - Definition from Techopedia*. Techopedia.com [online] 2012-09-21. [cit. 2021-10-12]. Dostupné na internete: <<https://www.techopedia.com/definition/5613/web>>.

- THE WEB ROBOTS PAGES, 2007. *About /robots.txt* [online] 2007. [cit. 2022-02-28].  
Dostupné na internete: <<https://www.robotstxt.org/robotstxt.html>>.
- THELWALL, M., STUART, D. 2006. *Web crawling ethics revisited: Cost, privacy, and denial of service*. J. Am. Soc. Inf. Sci., 57: 1771-1779. [online] 2006.  
[cit. 2022-02-21]. Dostupné na internete: <<https://doi.org/10.1002/asi.20388>>.
- YUHAO, F. 2018. *Design and implementation of distributed crawler system based on Scrapy*. In: IOP Conference Series: Earth and Environmental Science, 1–5, 2018.



## **ZOZNAM PRÍLOH**

Príloha A – Zdrojový kód a zozbierané dáta vo formáte ZIP

## **PRÍLOHA A**

## PRÍLOHA A

Súbor prílohy.zip obsahuje tri časti:

- crawler\_server,
- crawler\_ui,
- rss.json.

Súbor crawler\_server obsahuje:

- app.py,
- constants.py,
- emails.json,
- main.py,
- periods.json,
- records.log,
- regions.json,
- results.json,
- test.py.

Súbor crawler\_ui obsahuje:

- app.component.html,
- app.component.scss,
- app.component.spec.ts,
- app.component.ts,
- app.module.ts,
- app-routing.module.ts,
- dashboard.component.html,
- dashboard.component.ts,
- detail.component.html,
- detail.component.ts,
- email.service.ts,
- environment.prod.ts,
- environment.ts,
- excel.service.ts,
- index.html,
- max.directive.ts,
- package.json,

- `rss.service.ts`,
- `rssFromUrl.component.html`,
- `rssFromUrl.component.ts`,
- `styles.scss`.

Odkaz na GIT: [https://github.com/martinpecho/Navrh\\_weboveho\\_crawlera\\_BP](https://github.com/martinpecho/Navrh_weboveho_crawlera_BP)