Alati za učitavanje nestrukturiranih podataka

Galić, Slaven

Master's thesis / Diplomski rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:328068

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

ALATI ZA UČITAVANJE NESTRUKTURIRANIH PODATAKA

Diplomski rad

Slaven Galić

Mentor: doc.dr.sc. Josip Job

Osijek, 2017.

SADRŽAJ

1. UVOD 1

2. MODELI PODATAKA 2

2.1. PODATAK 2

2.2. NESTRUKTURIRANI PODATCI 3

2.2.1. UIMA standard 4

2.2.2. BUDUĆNOST NESTRUKTURIRANIH PODATAKA 5

2.3. OSTALI TIPOVI PODATAKA PO STRUKTURI 6

2.3.1. STRUKTURIRANI PODATCI 6

2.3.2. POLUSTRUKTURIRANI PODATCI 6

3. UČITAVANJE PODATAKA 8

3.1. TEHNIKE UČITAVANJA PODATAKA 8

3.1.1. SCREEN SCRAPING 8

3.1.2. WEB SCRAPING 9

3.1.3. REPORT MINING 10

3.2. ALATI ZA UČITAVANJE NESTRUKTURIRANIH PODATAKA 11

3.2.1. BIBLIOTEKE 11

3.2.2. ALATI 13

3.2.3. DODATCI ZA PREGLEDNIKE 16

3.2.4. USPOREDBA USLUGA I ALATA 19

4. PRAKTIČNI DIO 20

4.1. RAZVOJNO OKRUŽENJE 20

4.2. PRIMJENJENE TEHNOLOGIJE 21

4.3. ALAT ZA UČITAVANJE PODATAKA 24

4.4. ANALIZA FUNKCIONALNOSTI APLIKACIJE NA STVARNOM PRIMJERU 31

4.5. MOGUĆE NADOGRADNJE APLIKACIJE 32
5. ZAKLJUČAK 34
LITERATURA 35
SAŽETAK 37
ŽIVOTOPIS 38
PRILOZI 39

1. UVOD

Porastom korištenja interneta javno dostupna količina podataka je dosegla velike razmjere. S tolikom količinom koja ima određenu snagu i potencijal, pretpostavlja se da bi na svako pitanje trebao postojati odgovor, ali sama dostupnost ne donosi nužno i rješenje. Eksplozija podataka je neizbježan trend s obzirom na veliki tehnološki napredak, ljudi i organizacije sve više ovise o računalnim uređajim i izvorima informacija na Internetu. Danas raspolažemo sa golemim količinama sveprisutnih podataka. Većina podataka se nalazi u nestrukturiranom ili polustrukturiranom obliku gdje se podatak nalazi na mjestu predviđenom samo za otvaranje i čitanje određenog dokumenta. Današnja interakcija uglavnom se temelji na dokumentima. Dokumente primamo i šaljemo, generiramo ih iz vlastitih sustava i aplikacija ili ih izrađujemo ručno. Nestrukturirani i nepohranjeni podatci smanjuju iskoristivost, mogućnost analize i korištenje tih dokumenata. Veći dio dokumenata je nestrukturiran i pohranjen u različitim formatimaunutarnjih izvora kao što su papir, e-mail poruke, MS Word dokumenti, tehnička dokumentacija itd. Osim unutarnjih izvora, nestrukturirani podatci nalaze se i u vanjskim izvorima kao što su blogovi, forumi, internetske stranice te društvene mreže. Podatci koji se nalaze u ovim izvorima mogu odgovoriti na razna pitanja i pomoći pri rješavanju gorućih problema s kojima se susreću razni poslovni sustavi.

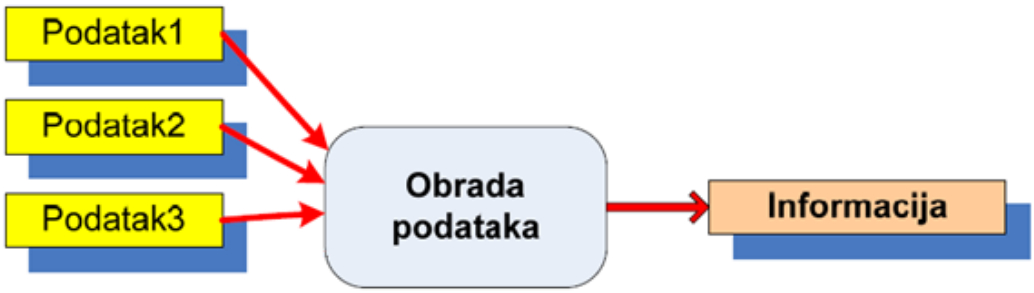
Za masivne količine strukturiranih i nestrukturiranih podataka upotrebljava se naziv „veliki podatci“ (engl. *Big Data*). Tako velike količine podataka gotovo je nemoguće obrađivati tradicionalnim alatima za upravljanje bazama podataka i aplikacijama za obradu. Koncept „velikih podataka“ ne odnosi se samo na količinu podataka, nego i na tehnologije potrebne za rad s velikim količinama podataka, njihovo prikupljanje i skladištenje. Prikupljanje podataka jedna je od bitnih stavki funkcioniranja cijelog koncepta. Podatci koji se nalaze na vanjskim izvorima prikupljaju se alatima za učitavanje nestrukturiranih podataka (engl. *Web-Scraping tools*). Učitavanje nestrukturiranih podataka je proces ekstrakcije informacija iz izvora koji nisu obrađivani i nisu strukturirani na pravilan način. Trenutno najveći problem jest nedostatak prikladnih alata specijaliziranih za integraciju svih tehnologija potrebnih za izvođenje procesa analize podataka. Kompletan alat trebao bi sadržavati cjelokupni proces od pronalaženja točnih pojmova i zahtjeva, prikupljanja informacija te obrade i skladištenja u ispravnom obliku.

2. MODELI PODATAKA

2.1. PODATAK

Podatak je vrlo jednostavna neobrađena činjenica koja ima neko značenje i na osnovu koje se oblikuje nekakva informacija. On je nematerijalne prirode i sam po sebi ne znači puno, ukoliko se ne zna njegova interpretacija. To je registrirano zapažanje tokom nekog procesa ili događaja. Podatak može biti u više oblika, najčešći su zvučni, slikovni, brojčani i tekstualni. Sastoji se od skupa kvantitativnih parametara koji se mogu zapisati kao nizovi znakova ili nizovi brojeva. U računalima, koja su do nedavno nazivana i strojevima za automatsku obradu podataka, ti se nizovi, radi pohrane, obrade i sl. pretvaraju u nizove bitova. Podatak jednostavno postoji u našim mislima i nema značenje unutar ili izvan svog postojanja ili o samom sebi, pa se zbog toga pridružuje značenju kojim opisujemo svojstva objekata. Može postojati u bilo kojem obliku bio upotrebljiv ili ne. On sadrži opis svojstva nekog entiteta, zapažanja tokom nekog procesa, registrirane činjenice ili događaje. Sama struktura podatka je apstraktna jer sadrži više svojstava, značenje, vrijeme i vrijednost [1]. Podatci koji su kombinirani u strukturi čine informaciju.

Riječ informacija potječe od lat. *Informatio* što znači predodžba, tumačanje, pojam. Skup podataka s pripisanim značenjem, osnovni element komunikacije koji, primljen u određenoj situaciji, povećava čovjekovo znanje [2]. Informacija je skup logički povezanih podataka, odnosno organiziranih i obrađenih činjenica koje predstavljaju nekakvu vijest. Nakon što je interpretirana, stavljena u kontekst ili kad joj je dano značenje, ona postaje znanje. Od samog početka korištenja računala, obrada različitih vrsta podataka bila je jedan od osnovnih zadataka. U slučajevima kada je jako bitno donijeti kvalitetnu odluku, veliku ulogu igra organizacija podataka sa kvalitetnim informacijama. Informacije se mogu strukturirati kako bi imale određenu namjenu, taj proces se naziva strukturiranje informacija. Informacije su često strukturirane prema kontekstu u interakciji s korisnicima ili većim bazama podataka. Baza podataka je organizirana zbirka određene količine podataka.



Sl. 2.1. Podatci nakon obrade postaju informacija

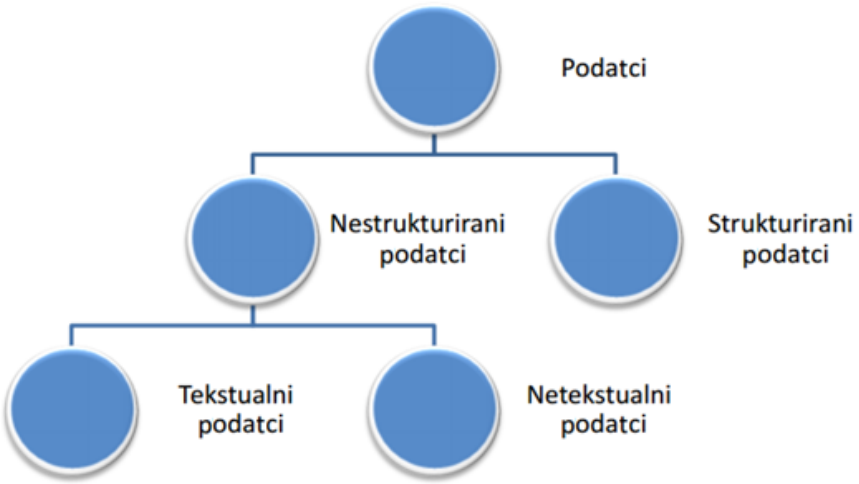
Tradicionalne baze podataka imaju dugotrajnu poziciju u većini procesa vezanih za jednostavno skladištenje podataka. Klasični relacijski model, gdje se baza sastoji od relacija, je star preko 40 godina, osmislio ga je Edgar Frank Codd, 1970. godine u svom *"A Relational Model of Data for Large Shared Data Banks"* [3]. Dizajn klasičnih relacijskih baza podataka nije se značajno mijenjao posljednjih desetak godina i danas sve teže može pratiti zahtjeve velikih količina podataka, tj. koncepta „velikih podataka" s kojima se moraju nositi sadašnje i buduće aplikacije. Također, porastom broja korisnika Interneta sve više i više podataka se razmjenjuje putem mreže, a ti podatci nisu baš strukturirani te su heterogeni i često nepotpuni.

2.2. NESTRUKTURIRANI PODATCI

Nestrukturirani podatci su „sirovi" i neorganizirani podatci. Nedostatak strukture čini obradu takvih podataka vremenski i energetski zahtjevnim zadatkom. Nestrukturirani podatci dolaze u mnogim oblicima i veličinama. Može ih se pohraniti u dokumente, izvješća, proračunske tablice, web stranice ili digitalne medije (slike, audio i video). Prvi korak u obradi tih podataka je dokumentiranje, konsolidacija i upravljanje. Iako se većina baza podataka danas može nositi s nestrukturiranim podatcima, smjer industrije je razvoj aplikacija koje bi upravljale sadržajem baze i tako upravljale nestrukturiranim podatcima. Bilo bi korisno za tvrtku u svim poslovnim slojevima pronaći mehanizam analize podataka kako bi se smanjili troškovi obrade nestrukturiranih podataka koji nastaju u organizaciji.

Postupak dodavanja strukture nestrukturiranim podatcima sastoji se od:

- Upotreba uzoraka teksta kao što su redovni izrazi za male ili velike strukture
- Upotreba tabličnog pristupa za prepoznavanje zajedničkih odjeljaka
- Upotreba tekstualne analize kako bi se razumio tekst i povezao sa ostalim informacijama



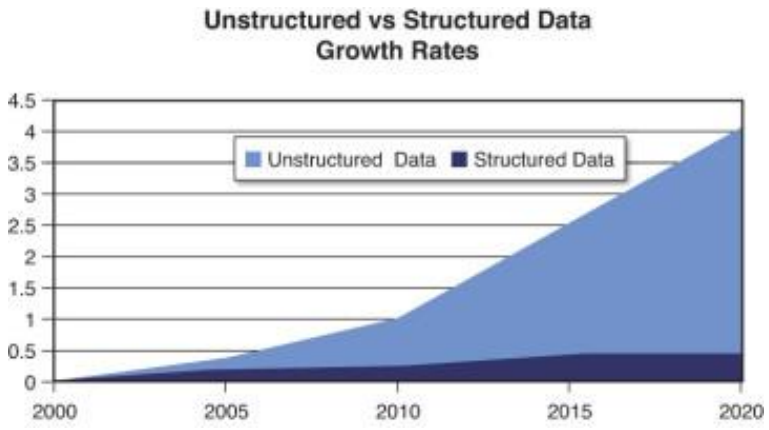
Sl. 2.2. Prikaz vrste podataka

2.2.1. UIMA standard

UIMA (engl. *Unstructured Information Management Architecture*) je standard koji koristi arhitekturu upravljanja nestrukturiranim informacijama i komponenta je arhitekture i implementacije programskog okvira za analizu nestruktuririranog sadržaja kao što su tekstualni, video i audio podatci. Napravljen je od strane IBM-a i pruža kompletnu programsku arhitekturu za razvoj, otkrivanje, sastav i implementaciju analitike za analizu nestrukturiranih podataka i integraciju sa tehnologijama pretraživanja. Motivacija za razvoj takvog prograsmkog okvira bila je izgradnja zajedničke platforme za nestrukturirane informacije, poticanje ponovne uporabe dijelova analize i smanjenja dupliciranja analize [4]. Jednostavna UIMA arhitektura omogućuje uključivanje vlastitih dijelova analize i kombinaciju sa postojećim.

Glavni cilj je transformacija nestrukturirane informacije u strukturiranu informaciju koristeći analizu kako bi se otkrili entiteti i odnosi i izgradio model koji ima približno strukturiranu ili polu-strukturiranu arhitekturu.

2.2.2. BUDUĆNOST NESTRUKTURIRANIH PODATAKA



Sl.2.3. Prikaz rasta strukturiranih i nestrukturiranih podataka

IZVOR: http://ptgmedia.pearsoncmg.com/images/chap1_9780133837964/elementLinks/01fig02.jpg (pristupljeno 11.6.2017.)

Budućnost pripada nestrukturiranim podatcima i vrijednim poslovnim spoznajama koje sadrži. Tvrtke trebaju razvijati i ažurirati svoje procese poslovne inteligencije kako bi uključile nestrukturirane podatke i otključale njihovu vrijednost. Razvijanjem odgovarajuće poslovne strategije, u kombinaciji s pravim praksama podataka i alatom za analizu, može se otkriti kako rješavati trenutne poslovne izazove i prikazati budućnost tvrtke.

2.3. OSTALI TIPOVI PODATAKA PO STRUKTURI

2.3.1. STRUKTURIRANI PODATCI

Strukturirani podatci su bazirani na shemi što znači da je struktura unaprijed isplanirana i strogo poznata. Nakon što se definira shema, podatci se mogu unositi na temelju kriterija zadanih u njoj. Najjednostavnije rečeno, strukturirani podatci su svi oni koji su organizirani u oblikovani repozitorij, obično je to baza podataka, tako da se njeni elementi mogu adresirati radi učinkovitije obrade i analize. Struktura podataka je vrsta spremišta koja organizira informacije u tu svrhu. U bazi podataka, na primjer, svako polje je diskretno i njezine se informacije mogu dohvatiti odvojeno ili zajedno s podatcima iz drugih polja, u različitim kombinacijama. Snaga takve strukture je njena sposobnost da obuhvati sve podatke i pruži korisne informacije. Strukturirani podatci se uglavnom odnose na informacije sa visokim stupnjem organizacije, tako da je uključivanje u relacijsku bazu neprekinuto i lako se može pretražiti jednostavnim, izravnim algoritmima određene tražilice ili pomoću drugih pretraživačkih operacija. Kada su informacije dobro strukturirane i predvidljive, tražilice mogu lakše organizirati i prikazati podatke na kreativan način. Kod označavanja strukturiranih podataka obično se koristi shema.org vokabular. Ovaj projekt koji su pokrenule Google, Bing i Yahoo tražilice služi za stvaranje strukturiranog označavanja podataka koje sve tražilice mogu razumjeti. Ovaj vokabular zahtjeva oblik koda koji se može dodati web stranici kako bi se definirali različiti elementi, poput datuma, slika i drugo. To omogućava pretraživačima učitavanje relevantnih dijelova web stranice u obliku isječaka ili privlačnih oglasa koji povećavaju stopu pregleda i klikova [5].

Ovakvi podatci su najzastupljeniji u razvoju aplikacija i pružaju najjednostavniji način obrade podataka. Bitno je naglasiti da strukturirani podatci čine 5-10% ukupnog postotka svih informatičkih podataka.

2.3.2. POLUSTRUKTURIRANI PODATCI

Kod polustrukturiranih podataka (engl. *semi-structured data*) ne postoji rigorozno definirana shema nego je način označavanja opcionalan. U nekim oblicima polustrukturiranih podataka shema uopće ne postoji, dok kod nekih postavlja samo labava ograničenja nad podatke. Podatke koji imaju labava ograničenja karakterizira nepravilna i implicitna struktura te fleksibilnost.

Primjer polustrukturiranih podataka su XML (engl. *eXtensible Markup Language*), JSON (engl. *Javascript Object Notation*), NoSQL (engl. *Not Only SQL*) baze podataka. Slično kao i kod strukturiranih, na polustrukturirane otpada samo 5 do 10% informatičkih podataka.

Prednost polustrukturiranih podataka je u tome što uključuju mogućnost prikaza podataka koje nije lako ograničiti shemom te sama fleksibilnost u pogledu prijenosa podataka, sposobnost prikaza strukturiranih podataka kao i sposobnost mijenjanja strukture tijekom određenog vremena.

Polustrukturirani podatci uglavnom imaju sljedeće karakteristike:

- Podatci su modelirani u obliku stabla ili grafova
- Podatci mogu postojati sa ili bez sheme

3. UČITAVANJE PODATAKA

Učitavanje podataka (engl. *Data Scraping*) je tehnika koja se koristi za ekstrakciju velikih količina podataka s lokalnog izvora, baze podataka ili Interneta. Obično se prijenos podataka između programa postiže pomoću podatkovnih struktura pogodnih za automatsku obradu podataka. Učitavanje je postupak dohvaćanja podataka, obično nestrukturiranih ili strukturiranih, iz izvora podataka kako bi se oni iskoristili za daljnju obradu ili pohranu. Uvoz u sustav međustrukturiranja obično prati pretvorba podataka i dodavanje metapodataka. Ekperimentalni podatci se prvo uvoze u računalo iz primarnih izvora, kao što su uređaji za mjerenje ili snimanje. Današnji elektronički uređaji obično predstavljaju električni priključak (npr. USB) pomoću kojeg se „sirovi podatci“ mogu prenijeti u osobno računalo [6].

3.1. TEHNIKE UČITAVANJA PODATAKA

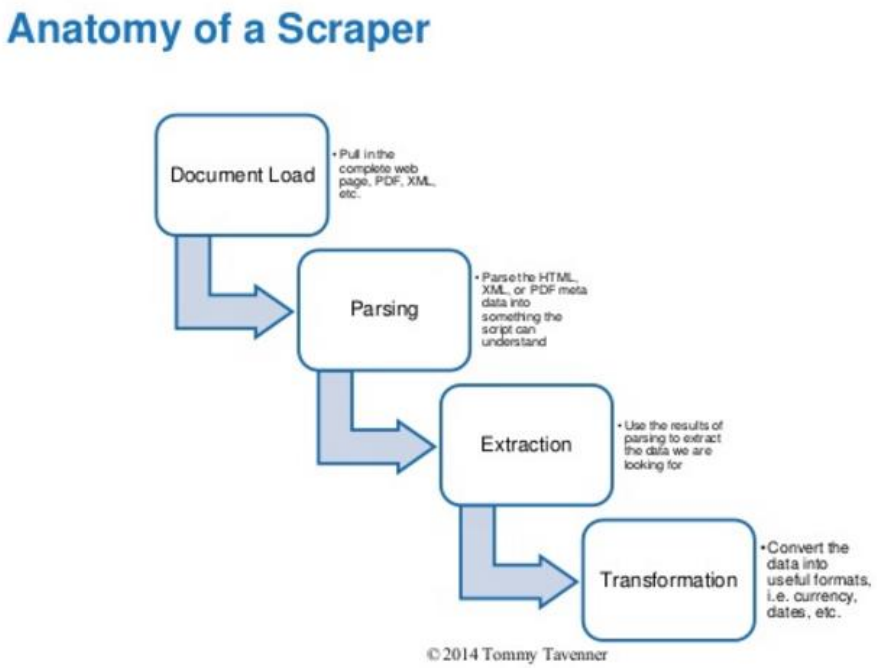
3.1.1. SCREEN SCRAPING

Sredinom 80-ih godina, pružatelji financijskih podataka kao što su Quotron, Telerate i Reuters prikazali su podatke u formatu 24 × 80 prilagođen čovjeku. Korisnici tih podataka, posebice banke za investicije, napisali su aplikacije za snimanje i pretvaranje takvih podataka u numeričke kako bi se iskoristili za izračune odluka o trgovanju. Screen scraping obično je povezan sa programskim skupljanjem vizualnih podataka iz nekakvog izvora, umjesto raščlanjivanja podataka kao kod web scrapinga. Izvorno, screenscraping odnosi se na praksu čitanja tekstualnih podataka sa zaslona računala. To je uglavnom učinjeno čitanjem memorije terminala kroz pomoćni priključak ili spajanjem izlaznog izlaza terminala jednog računalnog sustava s ulaznim priključkom na drugom. Izraz "struganje" (engl. *scraping*) zaslona također se obično koristi za dvosmjernu razmjenu podataka [7]. To bi mogli biti jednostavni slučajevi u kojima se kontrolni program kretao kroz korisničko sučelje ili složenije scenarije gdje kontrolni program unosi podatke u sučelje namjenjeno ljudima. Kao konkretan primjer klasičnog screen scraping alata, može se navesti računalo s korisničkim sučeljima iz razdoblja 80-tih.

Često korišteno samo kroz tekstualne terminale koji nisu bili mnogo više od virtualnih printera. Želja za sučeljem takvog sustava u modernijim sustavima je česta. Kvalitetno rješenje često zahtijeva više stvari koje nisu na raspolaganju, poput izvornog koda, dokumentacije sustava, API-ja ili programera s višegodišnjim iskustvom u računalnom sustavu. U takvim slučajevima, jedino moguće rješenje može biti napisati screen scraper. Screen scraper može se povezati s sustavom putem Telnet-a, oponašati tipke potrebne za navigaciju starim korisničkim sučeljem, obraditi rezultirajući izlazni prikaz, izvući željene podatke i proslijediti ga modernom sustavu. Moderne tehnike screen scraping-a obuhvaćaju snimanje bitmap podataka s ekrana i prikazivanje kroz OCR (engl. *Optical character recognition*) motor ili usklađivanje bitmap podataka s nekakvim očekivanim rezultatima. Putem aplikacija sa grafičkim sučeljem i s kontrolnim upitima, niz ekrana se automatski bilježi i pretvara u bazu podataka [8].

3.1.2. WEB SCRAPING

Web stranice izrađuju se pomoću tekstualnih jezika (HTML i XHTML), a često sadrže bogatstvo korisnih podataka u obliku teksta. Međutim, većina web stranica namijenjena je ljudima kao krajnjim korisnicima, dok strojevi teško iskorištavaju takve podatke za obradu ili pretvaranje u korisne informacije. Zbog toga su izrađeni alati koji mogu učitati sadržaj iz web stranica i pretvoriti ih u pogodan oblik. Web scraper je API ili alat za učitavanje podataka s web stranica. Web scraping koristi tehniku učitvanja velikih količina podataka s internetskih stranica nakon kojih se podatci sažimaju i spremaju u lokalnu datoteku na računalu ili u bazu podataka [9]. Podatci koje sadrže razne internetske stranice mogu se prikazivati samo putem internetskog preglednika. Internetski preglednici ne nude funkcionalnosti kao što je spremanje kopije podataka za kasniju upotrebu. Jedina mogućnost je kopirati i zalijepiti podatke. Stoga, web scraping omogućuje automatsku realizaciju procesa spremanja kopije podataka u vrlo kratkom vremenu. Web scraping je vrlo sličan indeksiranju, procesu kojim pretraživači indeksiraju internetski sadržaj. Razlika je u tome što internetski preglednici imaju pravila postavljena u *robot.txt* datoteci i moraju ih poštovati, dok web scraperi ne.



Sl. 3.1. Anatomija Web Scrapera

IZVOR: https://image.slidesharecdn.com/scrapingdatafromthewebanddocuments-140806122932-phpapp01/95/scraping-data-from-the-web-and-documents-11-638.jpg?cb=1407328408

(preuzeto 10.6. 2017)

3.1.3. REPORT MINING

Report mining je prikupljanje podataka iz izvješća koja su čitljiva i prilagođena tako da ih ljudi mogu čitati. Prikupljanje podataka zahtjeva vezu sa sustavom koji sadrži izvorni program, korištenje odgovarajućih standarda ili programskih sučelja i složene upite. Korištenjem standardnih opcija koje su dostupne pri izradi izvješća i usmjeravanjem na određenu datoteku umjesto na pisač, može se generirati statično izvješće prikladno za analizu bez pristupa internetskoj vezi [10]. Ovaj pristup može izbjeći prekomjerenu upotrebu glavnih dijelova računala tijekom radnog vremena, može smanjiti cijenu određenih sustava za sakupljanje podataka te ponuditi vrlo brzo i prilagođeno rješenje za izradu izvješća. Razlika između report mining proces i drugih tehnika prikupljanja podataka je ta da report mining uključuje izdvajanje podataka iz datoteka u ljudima čitljivom obliku, kao što su HTML, PDF ili tekst. Ovaj proces može pružiti brz i jednostavan način za prikupljanje podataka bez potrebe za programiranjem i izradom vlastitih alata.

3.2. ALATI ZA UČITAVANJE NESTRUKTURIRANIH PODATAKA

3.2.1. BIBLIOTEKE

REQUESTS

Requests je Apache2 licencirana HTTP biblioteka, napisana u Pythonu [11]. Namijenjena je ljudima da ih koriste za interakciju sa PHP ili Python programskim jezikom. Requests omogućuje slanje HTTP zahtjeva koristeći PHP programski jezik. Kod korištenja ove biblioteke nema potrebe za ručnim dodavanjem nizova upita URL-ovima ili oblikovanja kodiranih POST podataka.

```
1  // Dodavanje Requests biblioteke
2  include('/path/to/library/Requests.php');
```

Sl. 3.2. Primjer korištenja Requests biblioteke

Najosnovniji primjer korištenja Requests biblioteke je onaj u kojem se koristi GET metoda. Zahtjev se sprema u varijablu *requests* koja postaje objekt s kojim se manipulira i koji sadrži podatke.

```
1  // Izrada varijable za spremanje podataka
2  $response = Requests::get('https://github.com/');
```

Sl. 3.3. Izrada varijable za spremanje podataka pomoću Requests biblioteke

GUZZLE

GUZZLE je PHP HTTP klijent za izvlačenje podataka iz HTML i XML datoteka. Ova biblioteka pruža nekoliko jednostavnih metoda i idioma za navigaciju, pretraživanje i izmjenu dobivenih podataka. Predstavlja jednostavan alat za analizu dokumenta i vađenje podataka koji su nam potrebni [12]. GUZZLE automatski pretvara dolazne dokumente u Unicode i odlazne dokumente u UTF-8. GUZZLE može slati sinkrone i asinkrone zahtjeve pomoću istog sučelja.

```php
use GuzzleHttp\Stream\Stream;
$response = $client->request('GET', 'http://httpbin.org/get');

echo $response->getBody()->read(4);
echo $response->getBody()->read(4);
echo $response->getBody()->read(1024);
var_export($response->eof());
```

Sl. 3.4. Primjer korištenja GUZZLE klijenta

Klijent koristi tok (engl. *stream*) za prijenos i preuzimanje podataka. GUZZLE klijent će prema zadanim postavkama pohraniti tijelo poruke u stream koji koristi privremene PHP tokove.

```php
GET
$client->get('http://httpbin.org/get', [/** options **/])
POST
$client->post('http://httpbin.org/post', [/** options **/])
HEAD
$client->head('http://httpbin.org/get', [/** options **/])
PUT
$client->put('http://httpbin.org/put', [/** options **/])
DELETE
$client->delete('http://httpbin.org/delete', [/** options **/])
OPTIONS
$client->options('http://httpbin.org/get', [/** options **/])
PATCH
$client->patch('http://httpbin.org/put', [/** options **/])
```

Sl. 3.5. Različiti načini dohvaćanja podataka putem klijenta

HTTPFul

HTTPFul je visokokvalitetna HTML i XML biblioteka napisana u PHP. Jedinstvena je po tome što kombinira brzinu i cjelovitost XML značajki sa jednostavnim izvornim PHP API-jem koji je uglavnom kompatibilan, ali i superioran nad drugim API-jima [13]. HTTPFul također funkcionira na principu zahtjeva u koji se spremaju podatci koji se nakon toga ispisuju na jednostavan način.

```php
$url = "https://api.github.com/users/nategood";
$response = \Httpful\Request::get($url)
    ->expectsJson()
    ->withXTrivialHeader('Just as a demo')
    ->send();

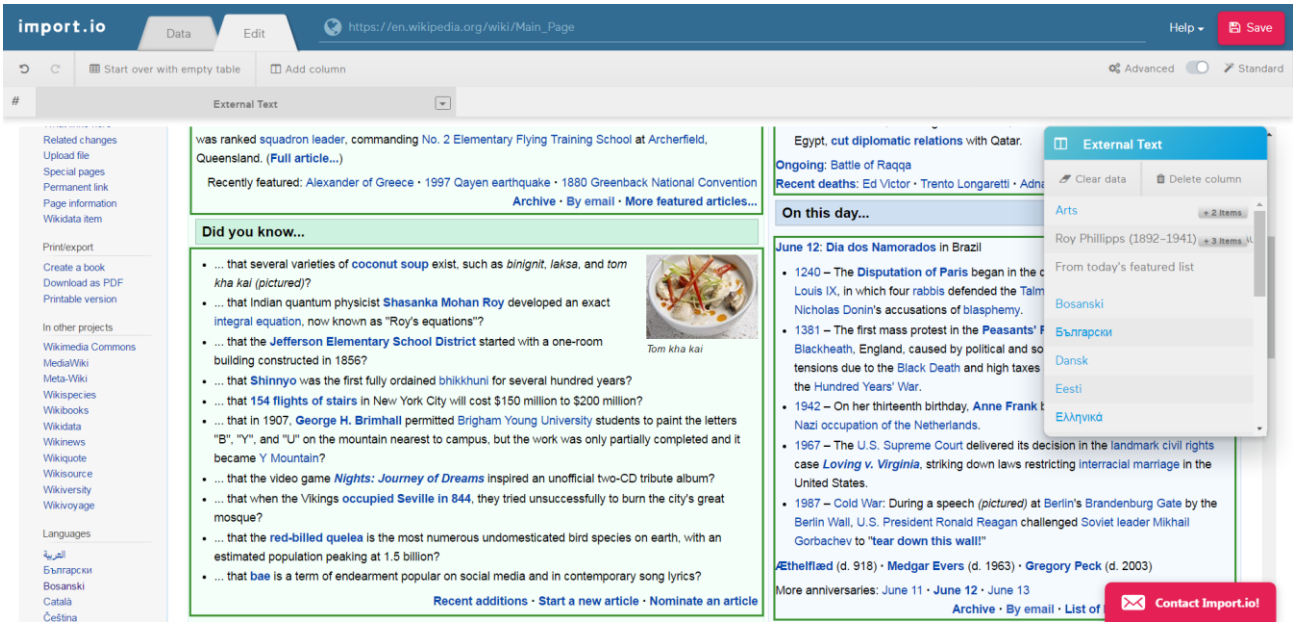
echo "{$response->body->name} joined GitHub on " .
                    date('M jS', strtotime($response->body->created_at)) ."\n";
```

Sl. 3.4. Primjer korištenja HTTPFul bibilioteke

3.2.2. ALATI

IMPORT.IO

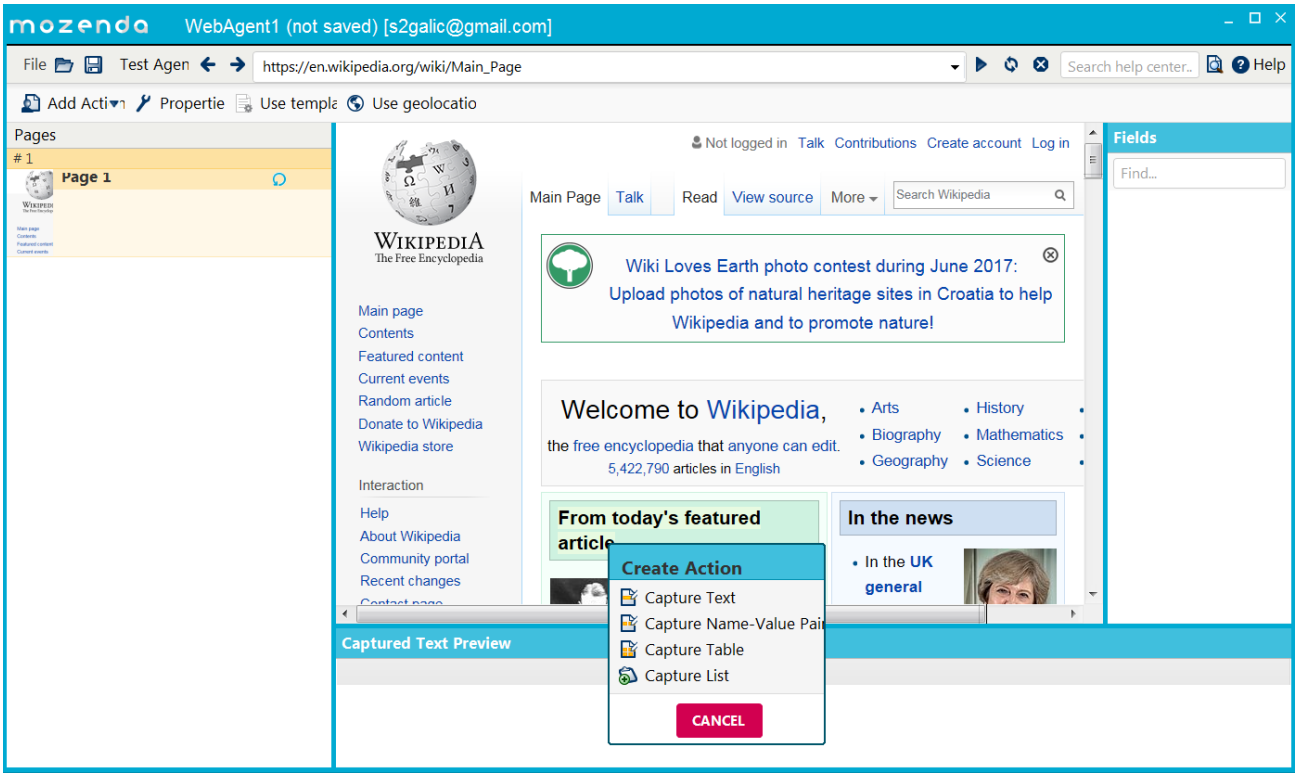
Import.io je web-based platforma za izdvajanje podataka s web stranica. Alat koji omogućuje pretvaranje nestrukturiranih web podataka u strukturirani format za upotrebu u strojnom učenju, umjetnoj inteligenciji, praćenju maloprodajnih cijena, pretraživanju lokacija, kao i akademskim i drugim istraživanjima [14]. Korisnici unesu URL i aplikacija pokuša automatski izvući podatke za koje misli da su korisniku potrebni, ako automatsko izdvajanje ne daje točno ono što vam je potrebno, sučelje omogućuje da se filtrira pretraga i količina podataka svede na konkretne. Podatci koje prikupljaju korisnici pohranjuju se na poslužiteljima oblaka Import.io i mogu se preuzeti kao CSV, Excel, Google tablice, JSON ili pristupiti putem API-ja. Korisnici mogu jednostavno integrirati internetske podatke uživo u vlastite aplikacije ili softver za analizu i vizualizaciju. Više izvora podataka može se izdvojiti istodobno.



Sl. 3.6. Primjer korištenja Import.io programskog alata

Mozenda

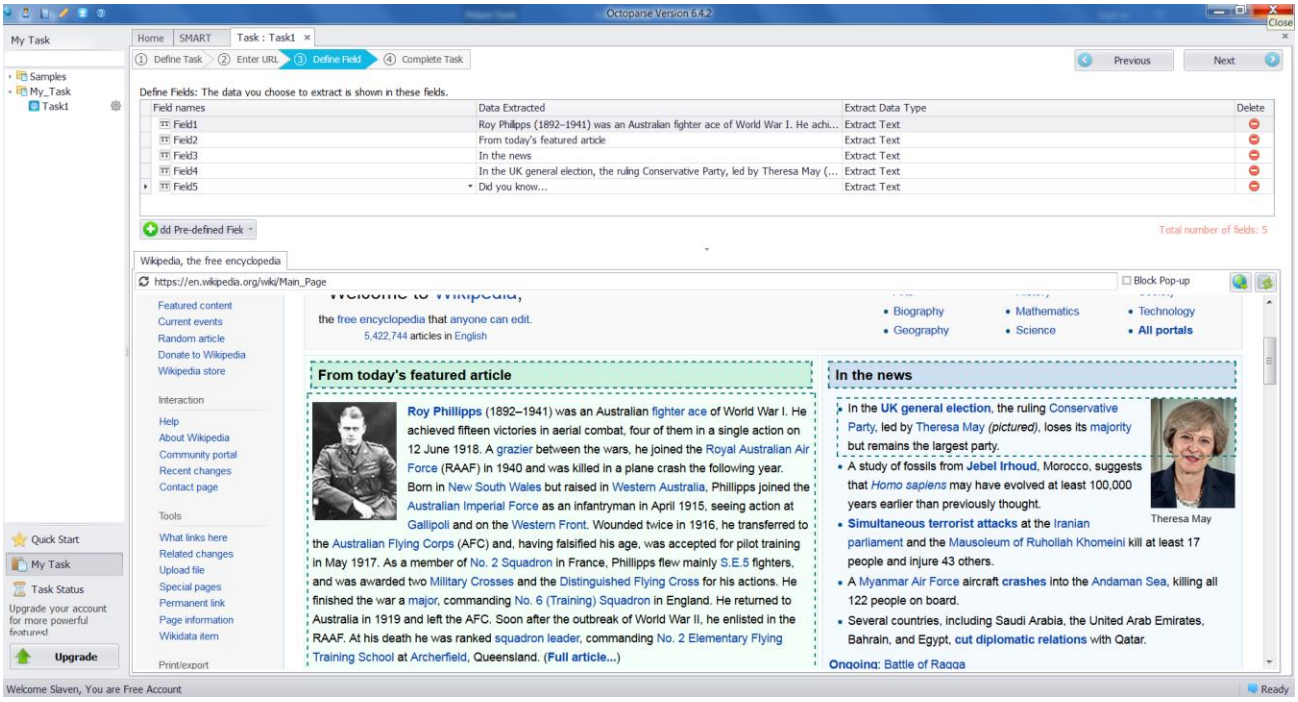
Mozenda je aplikacijski okvir za indeksiranje web stranica i vađenje strukturiranih podataka koji se mogu koristiti za širok raspon korisnih aplikacija, poput rudarenja podataka, obrade podataka ili povijesnog arhiviranja [15]. Iako je Mozenda alat izvorno dizajniran za učitavanje kompletnih web stranica, a može se koristiti i za ekstrakciju podataka pomoću API-ja (kao što su Amazon Associates Web Services) ili kao web pretraživač opće namjene. Mozenda koristi opciju Agent Buildera. Agent Builder podržava stvaranje agenata koji prikupljaju određene podatke s web stranica. Agent ima opcije koje se mogu mjenjati kako bi se postigla što konkretnija pretraga i dobio kvalitetniji rezultat.



Sl. 3.7. Primjer korištenja Mozenda pogramskog alata

OCTOPARSE

Octoparse je besplatan, jednostavan za korištenje, ali moćan alat za učitavanje podataka sa klijentske strane koji može biti od velike pomoći u pojednostavljenju procesa učitavanja internetskih stranica, povećavajući učinkovitost i optimiziranjem performansi [16].

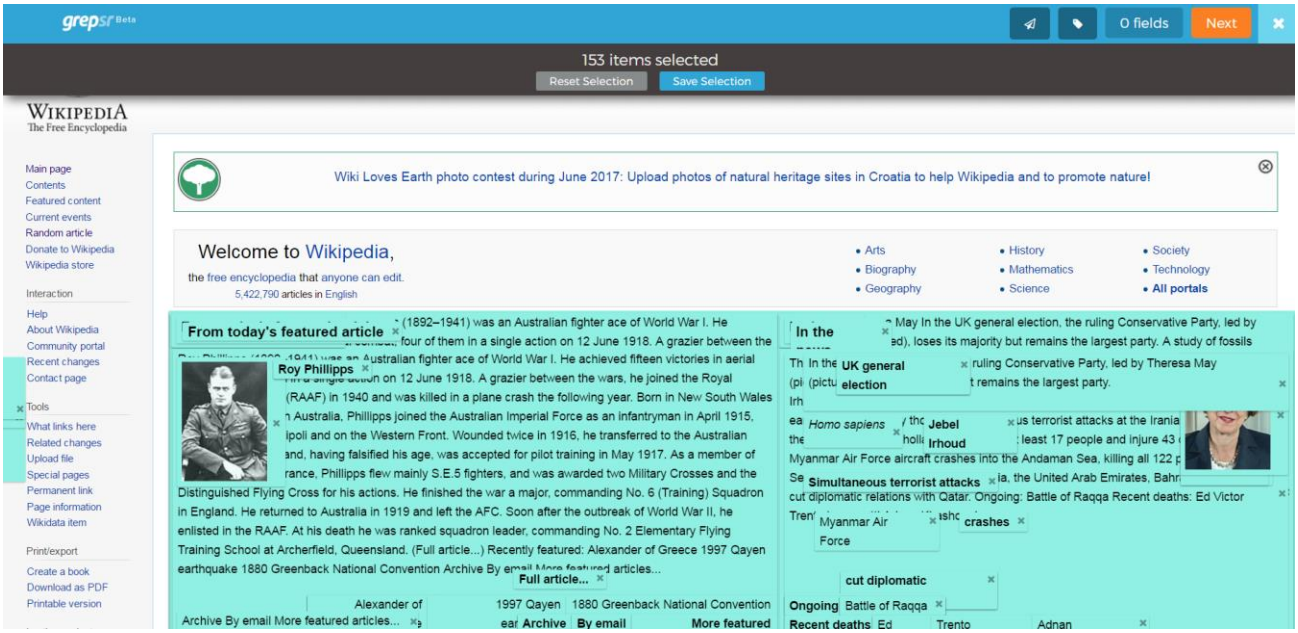


Sl. 3.8. Primjer korištenja Octoparse alata

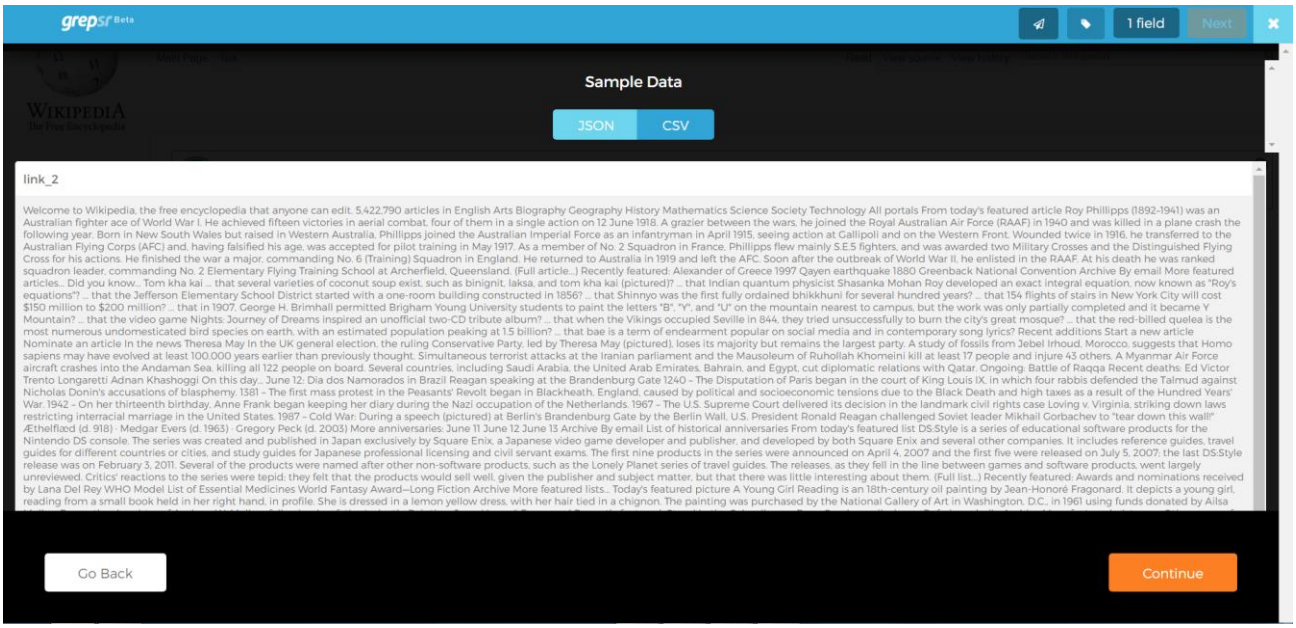
3.2.3. DODATCI ZA PREGLEDNIKE

GREPSR

Grepsr je besplatni dodatak za Google Chrome preglednik koji vam omogućuje jednostavno izdvajanje podataka bilo koje web stranice pomoću intuitivnog point and click načina i pretvaranje tih podataka u formate poput .CSV (engl. *Comma Separated Value*) ili .JSON (engl. *JavaScript Object Notation*) [17].



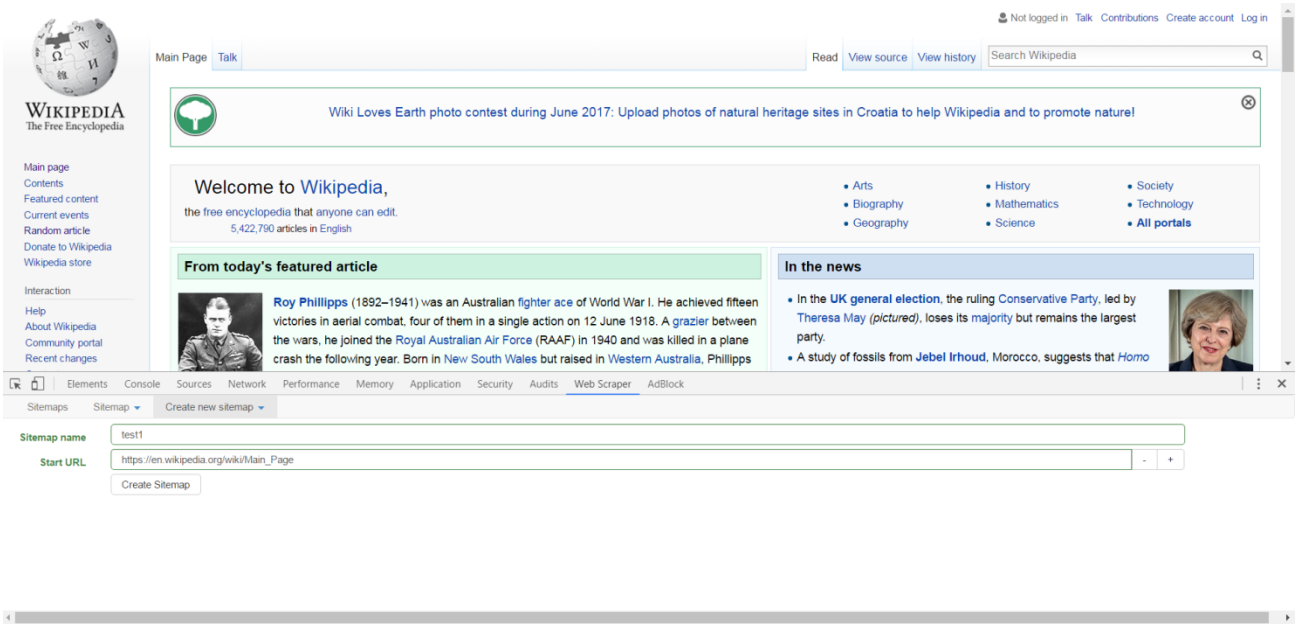
Sl. 3.9. Primjer korištenja Grepsr dodatka



Sl. 3.10. Spremanje podataka u polustrukturiranom obliku

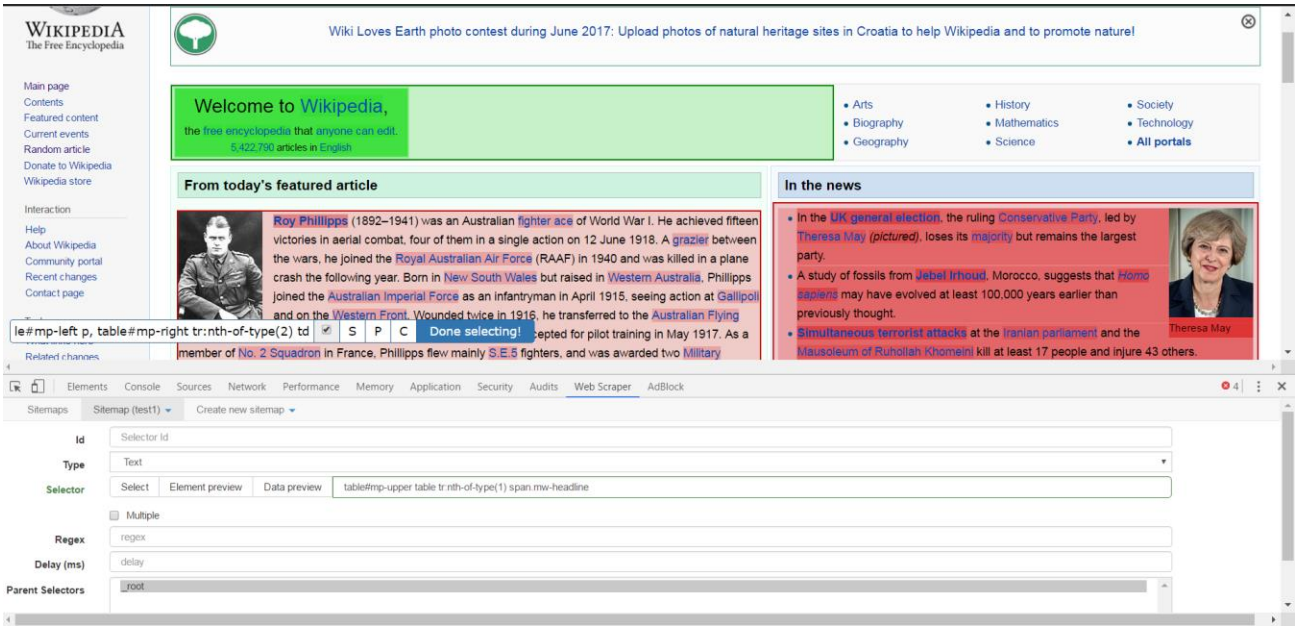
WEB SCRAPER CHROME DODATAK

Web Scraper je proširenje za Google Chrome preglednik izgrađeno za ekstrakciju podataka s web stranica [18]. Pomoću ovog proširenja možete izraditi plan (engl. *Sitemap*) kako bi trebalo proći web stranicu i što bi trebalo izvući. Pomoću ovih planova Web Scraper prelazi kroz internetsku stranicu prema planu i izvlači sve podatke. Kasnije kopirani podatci mogu se izvesti u .CSV formatu.



Sl. 3.11. Primjer korištenja Web Scraper dodatka za Google Chrome preglednik

Prvo što se mora napraviti je izraditi plan (engl. *Sitemap*), gdje pri kreiranju stavljamo naziv i početnu poveznicu. Nakon toga može definirati detalje koji će se koristiti u planu, odnosno postaviti *selector* po kojima će alat pretražiti internetsku stranicu. Dodatak nudi opciju point and click, tako da se određeni dijelovi samo odaberu klikom miša.



Sl. 3.12. Odabir *selectora* u Web Scraper dodatku

3.2.4. USPOREDBA USLUGA I ALATA

USPOREDBA WEB SCRAPING ALATA

USLUGE I BIBLIOTEKE	IMPORT.IO	Mozenda	Octoparse	GREPSR	WEB SCRAPER CHROME DODATAK
CIJENA	Besplatna proba	30 dana besplatno	Besplatno 75$/mjesečno	Besplatan dodatak 99$/mjesec	Besplatno
BRZINA	★★★★★	★★★	★★★★	★★★★	★★★
LAKOĆA KORIŠTENJA					
POTREBAN CLIENT/AGENT	NE	DA	DA	NE	NE
PODRŠKA	DA	DA	DA	DA	NE
DOSTUPNA DOKUMENTACIJA	DA	DA	DA	DA	DA
OS	Cloud usluga	DA	Windows 7,8,10	Google Chrome	Google Chrome

Sl. 3.13. Usporedba alata za učitavanje podataka

4. PRAKTIČNI DIO

Praktični dio rada, alat za učitavanje podataka, implementiran je kao web aplikacija koristeći sve tehnologije koje su navedene u ovom radu. Na strani klijenta (engl. *front-end*) korištene su tehnologije HTML, CSS, Bootstrap. Na strani poslužitelja korišten je PHP programski jezik.

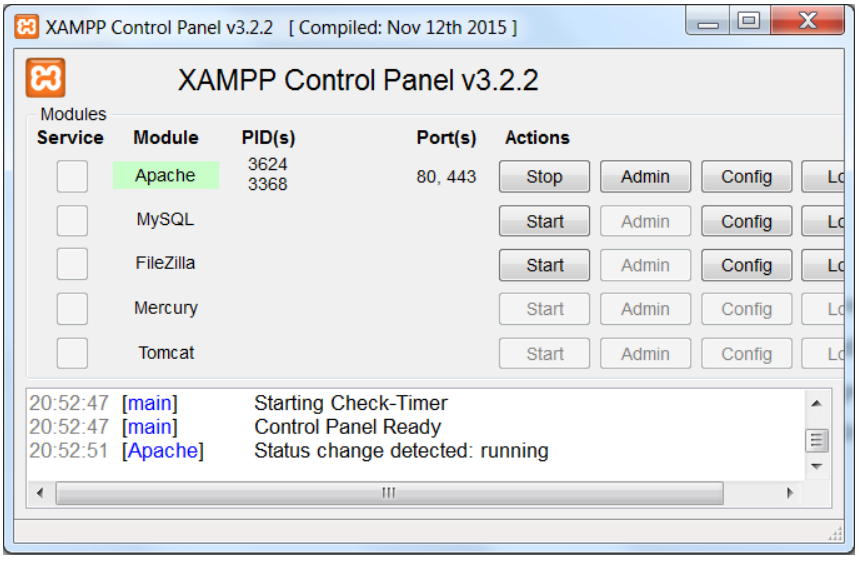
4.1. RAZVOJNO OKRUŽENJE

Prije početka izrade svakog programskog alata potrebno je pripremiti razvojno okruženje. Ovaj projekt je napravljen u Windows operacijskom sustavu, a za njegovu izradu bilo je potrebno instalirati:

1. XAMPP serverski paket
2. Uređivač teksta – Notepad ++

XAMPP instalacija

Sa službene stranice može se preuzeti instalacijski paket (*.exe*) za Windows operacijski sustav. Nakon što je instalacija završena, potrebno je omogućiti pokretanje Apache platforme u Windows operativnom sustavu. Odabirom na početnu tipku „Start“ u kontrolnom prozoru pokreće se Apache web platforma koja služi kao server potreban za razvoj dinamičkih web stranica.



Sl. 4.1. Kontrolni prozor XAMPP paketa

Nakon što se pokrenio Apache web servis, aplikacija se može izraditi i testirati u stvarnom vremenu, što znači da svaka promjena i funkcionalnost može biti trenutno vidljiva.

4.2. PRIMJENJENE TEHNOLOGIJE

HTML

HTML (engl. *Hyper Text Markup Language*) je jezik za izradu Internet stranica. Nastao je kako bi se tekst, slika i zvuk lakše strukturirali i prezentirali putem internetskog preglednika. Zadatak HTML-a je uputiti internetski preglednik kako treba prikazati neki Internet dokument. Neke od većih zadaća HTML-a su zadovoljiti potrebe današnjih modernih Internet stranica i aplikacija u pregledniku. Cilj HTML-a je umanjiti količinu potrebnih vanjskih dodataka, proširenja i skripti (engl. *plugin*) [19]. U nastavku teksta na slici 4.2. dan je programski kod izrađene aplikacije.

```php
<?php
include "simple_html_dom.php";
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>Web Scraper</title>

    <!-- CSS -->
    <link href="assets/bootstrap/css/bootstrap.min.css" rel="stylesheet" media="screen">
    <link href="assets/css/font-awesome.min.css" rel="stylesheet" media="screen">
    <link href="assets/css/simple-line-icons.css" rel="stylesheet" media="screen">
    <link href="assets/css/animate.css" rel="stylesheet">

    <!-- Custom styles CSS -->
    <link href="assets/css/style.css" rel="stylesheet" media="screen">

    <script src="assets/js/modernizr.custom.js"></script>


</head>
<body>
```

Sl. 4.2. Dio HTML programskog dijela

CSS

CSS (engl. *Cascading Style Sheets*) je skup pravila kojima se stiliziraju HTML elementi. Prvenstveno je napravljen kako bi se odvojio dizajn Internet stranice od sadržaja dokumenta. Kako su danas mobilni internetski preglednici jedni od najrasprostranjenijih načina pristupa Internetu, CSS ima mogućnost prilagodbe elemenata prema razlučivosti ekrana. Novije Internet stranice koriste takav pristup koji se naziva responzivni web dizajn [20].

```css
body {
    font: 300 14px/1.8 'Raleway', sans-serif;
    color: #666;
    overflow-x: hidden;
}

img {
    max-width: 100%;
    height: auto;
}

a {
    color: #E7746F;
}

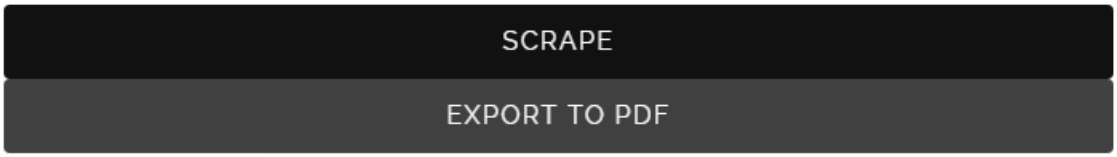
a:hover {
    text-decoration: none;
    color: #999;
}

```

Sl. 4.3. Dio CSS programskog dijela

BOOTSTRAP

Za izradu korisničkog sučelja aplikacije primjenjen je još jedan vrlo popularan skup alata koji se se koriste za izradu Internet stranica, proširenje (engl. *framework*) Bootstrap. Bootstrap služi za izradu i dizajniranje sučelja interaktivnih Internet aplikacija [21]. Ovaj skup alata omogućava brzu izradu različitih elemenata kao što su polja za upis, tipke za odabir, tipke za potvrđivanje, izbornici i drugo. Pri izradi aplikacije može se koristiti gotov predložak sa setom izrađenih elemenata kako bi se olakšao i ubrzao postupak. Neke od mogućnosti stiliziranja HTML elemenata koristeći Bootstrap prikazane su na slici 4.4. gdje je prikazana tipka za učitavanje i tipka za izvoz u PDF format.



Sl. 4.4. Prikaz gotovog predloška tipke izrađene u Bootstrap alatu

PHP

PHP je skriptni jezik pomoću kojeg je moguće kreirati HTML stranicu na poslužitelju prije nego što se ona, popunjena dinamičkim sadržajem, pošalje i prikaže korisniku. Na ovakav način gdje se HTML sadržaj generira onemogućuje se korisniku da vidi izvorni PHP programski kod koji je generirao sadržaj, već je vidljiv samo HTML dio [22]. Za izradu ove aplikacije, PHP se najviše koristio za dohvaćanje i učitavanje podataka. Kombiniranjem gotovih biblioteka olakšao se postupak izvlačenja podataka sa različitih izvora tj. poveznica. Jedna od biblioteka korištenih je PHP Simple HTML Dom Parser. Ta biblioteka je napisana u PHP programskom jeziku i služi za manipulaciju HTML kodom na vrlo jednostavan način.

```php
// Uključi PHP biblioteku
include('simple_html_dom.php');

// Preuzmi HTML dio sa određene poveznice
$html = file_get_html('http://www.etfos.unios.hr/~s2galic/index.php');

// Pronađi sve podatke koje sadrže oznaku a i ispiši ih
foreach($html->find('a') as $e)
    echo $e->href . '<br>';

// Pronađi sve podatke koje sadrže oznaku img i ispiši ih
foreach($html->find('img') as $e)
    echo $e->src . '<br>';

// Pronađi sve slike koje sadrže oznaku img i ispiši tekst koji je uz njih
foreach($html->find('img') as $e)
    echo $e->outertext . '<br>';
```

Sl. 4.5. Primjeri korištenja PHP Simple HTML Dom Parsera

4.3. ALAT ZA UČITAVANJE PODATAKA

Izrađena aplikacija se temelji na korištenju modernih tehnologija za izradu kvalitetnih, grafički privlačnih Internet stranica koje su ujedno i jednostavne za korištenje svim korisnicima. Aplikacija se sastoji od naslovne stranice gdje su sve funkcionalnosti aplikacije, polje za unos željene stranice te opcije za učitavanje ili izvlačenje podataka.

Zbog lakšeg pregleda rezultati se prikazuju na istoj naslovnoj stranici u formatiranom obliku ovisno o odabiru opcija. Sučelje izrađene aplikacije prikazano je na slici 4.6.



Sl. 4.6. Sučelje izrađene aplikacije

U aplikaciji su implementirane sljedeće mogućnosti(funkcionalnosti):

- Unos poveznice na određenu stranicu
- Odabir osnovnih elemenata koji sadrže podatke u obliku HTML atributa(naslov, podnaslov, poveznice)
- Odabir naprednih elemenata koji sadrže podatke u obliku HTML atributa(paragraf, putanja na element, podnože, zaglavlje)
- Dodavanje opcije učitavanja sa *class* selektorom
- Dodavanje opcije učitavanja sa više *class* selektora
- Dodavanje opcije učitavanja sa *id* selektorom
- Dodavanje opcije učitavanja sa više *id* selektora
- Tipka za učitavanje podataka
- Tipka za vraćanje na vrh stranice



Sl. 4.7. Izgled funkcionalnosti aplikacije

Korištenje Bootstrap alata olakšava izradu sučelja aplikacije tako što se gotov predložak može iskoristiti bez potrebe za dizajniranjem potrebnih elemenata. Na slici 4.7. može se vidjeti HTML kod u kojemu su definirani dijelovi unosa poveznice i odabir opcija za učitavanje.

```
87                      <div class="form-group wow fadeInUp">
88                          <label class="sr-only" for="url">Url</label>
89                          <input type="text" class="form-control" name="url" placeholder="Unesite URL">
90                      </div>
91
92                      <div class="form-group wow fadeInUp" data-wow-delay=".1s">
93                          <label for="h1">< H1 ></label>
94                          <input type="checkbox" id="h1" name="h1" value="h1">
95
96                          <label for="h2">< H2 ></label>
97                          <input type="checkbox" id="h2" name="h2" value="h2">
98
99                          <label for="h3">< H3 ></label>
100                         <input type="checkbox" id="h3" name="h3" value="h3">
101
102                         <label for="a">< a ></label>
103                         <input type="checkbox" id="a" name="a" value="a">
104
105                         <label for="srcImg">< Izvor slike ></label>
106                         <input type="checkbox" id="srcImg" name="srcImg" value="srcImg">
107                     </div>
108
109                     <div class="form-group wow fadeInUp" data-wow-delay=".5s">
110                         <label class="sr-only" for="element">element</label>
111                         <input type="text" id="element" class="form-control" name="element" placeholder="
                            Unesite drugi HTML atribut">
112                     </div>
113                     <div class="form-group wow fadeInUp" data-wow-delay=".5s">
114                         <label for="plaintext">plaintext</label>
115                         <input type="checkbox" id="plaintext" name="plaintext" value="plaintext">
116                         <label for="src">src</label>
117                         <input type="checkbox" id="src" name="src" value="src">
118                     </div>
```

Sl. 4.8. Izrada polja za unos poveznice i odabira opcija za učitavanje

Kao što je prije opisano u izradi aplikacije korištena je biblioteka Simple HTML Dom Parser koja na vrlo jednostavan način omogućava učitavanje podataka ovisno o unešenoj poveznici i odabranim opcijama za učitavanje. Prije nego što se učita nekakav sadržaj aplikacija mora primiti poveznicu (engl. *url)* iz koje će učitati podatke. Nakon unosa poveznice sadržaj stranice se pohranjuje u varijablu u obliku HTML koda.

```php
<?php

if (isset($_POST["url"])) {
    $url = $_POST["url"];
    echo '
    <section class="pfblock pfblock-gray" id="url">

        <div class="container">

            <div class="row skills">

                <div class="row">

                    <div class="col-sm-6 col-sm-offset-3">

                        <div class="pfblock-header wow fadeInUp">
                            <h2 class="pfblock-title">url:</h2>
                            <div class="pfblock-line"></div>
                            <div class="pfblock-subtitle">
                                '. $url .'
                            </div>
                        </div>

                    </div>

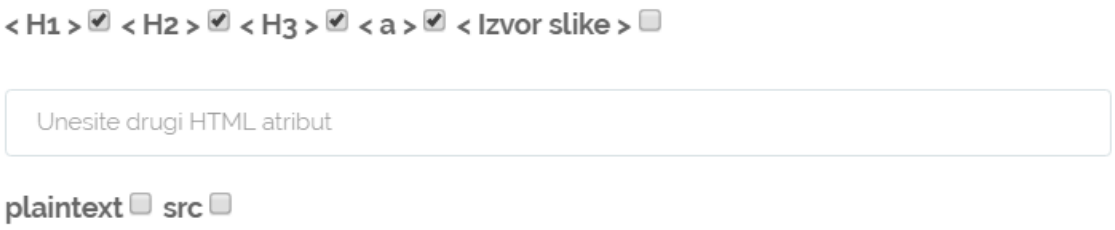
                </div><!-- .row -->
            </div>

        </div>

    </section>
    ';
    // Create DOM from URL or file
    $html = file_get_html($url);
```

Sl. 4.9. Programski dio unosa poveznice i spremanja za daljnju obradu

Nakon što su se podatci učitali izvršava se odabir opcije za učitavanje i ispis podataka koje element sadrži. Programski kod koji se odrađuje u tom trenutku prikazan je na slici 4.11.



Sl. 4.10. Odabir opcija po kojima se podatci učitavaju

```php
if (isset($_POST["h1"])) {
    // Find all H1
    echo '
    <section class="pfblock pfblock-gray" id="h1">

        <div class="container">

            <div class="row skills">

                <div class="row">

                    <div class="col-sm-6 col-sm-offset-3">

                        <div class="pfblock-header wow fadeInUp">
                            <h2 class="pfblock-title">All H1:</h2>
                            <div class="pfblock-line"></div>
                            <div class="pfblock-subtitle">
                                ';

    foreach($html->find('H1') as $element)
           echo $element->plaintext . '<br>';

    echo '
                            </div>
                        </div>

                    </div>

                </div><!-- .row -->
            </div>

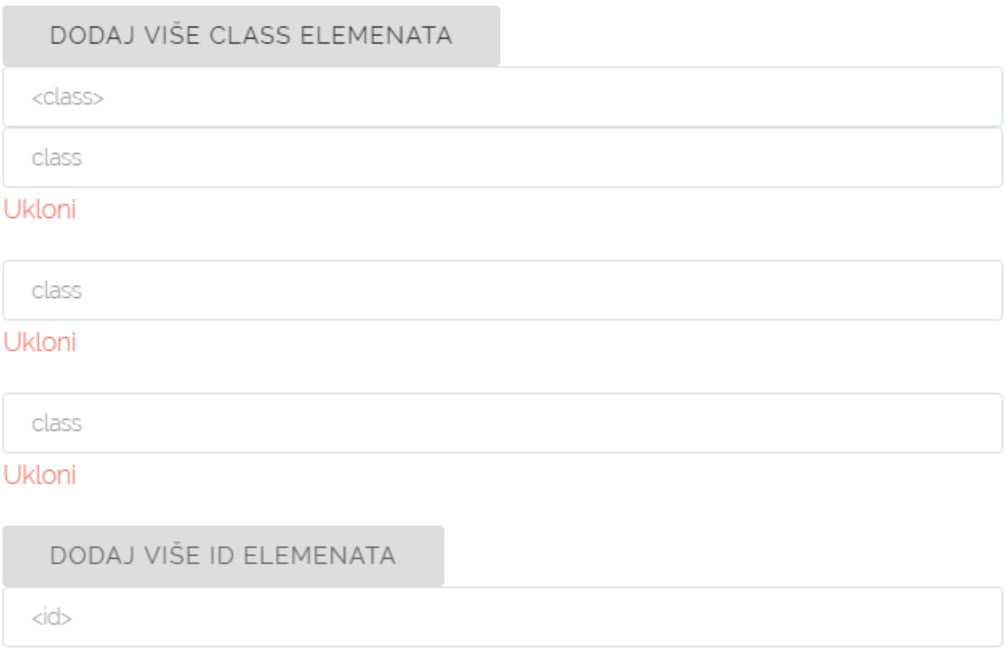
        </div>

    </section>
    ';

```

Sl. 4.11. Programski dio odabira i ispisa odabranog elementa

Za svaki od elementa se izvršava petlja koja ispisuje podatke koje taj element sadrži. Kako bi se omogućio uži izbor i dobili što konkretniji podatci, aplikacija ima mogućnost odabira opcije učitavanja po *class* i *id* selektorima, te kombinaciji istih sa ostalim HTML elementima.



Sl. 4.12. Mogućnost odabira ispisa elementa po jednom ili više *class* selektora

```
491     for ($i=1; $i <= 10; $i++) {
492         if (isset($_POST["id".$i])) {
493             $id = $_POST["id".$i];
494             if (!($id == "")) {
495                 echo '
496                 <section class="pfblock pfblock-gray" id="class">
497 
498                     <div class="container">
499 
500                         <div class="row skills">
501 
502                             <div class="row">
503 
504                                 <div class="col-sm-6 col-sm-offset-3">
505 
506                                     <div class="pfblock-header wow fadeInUp">
507                                         <h2 class="pfblock-title">class: #'.$id.'</h2>
508                                         <div class="pfblock-line"></div>
509                                         <div class="pfblock-subtitle">
510                                             ';
511 
512                 foreach($html->find("#".$id) as $element)
513                        echo $element->plaintext . '<br>';
514 
515                 echo '
516                                         </div>
517                                     </div>
518 
519                                 </div>
520 
521                             </div><!-- .row -->
522                         </div>
523 
524                     </div>
525 
526                 </section>
```

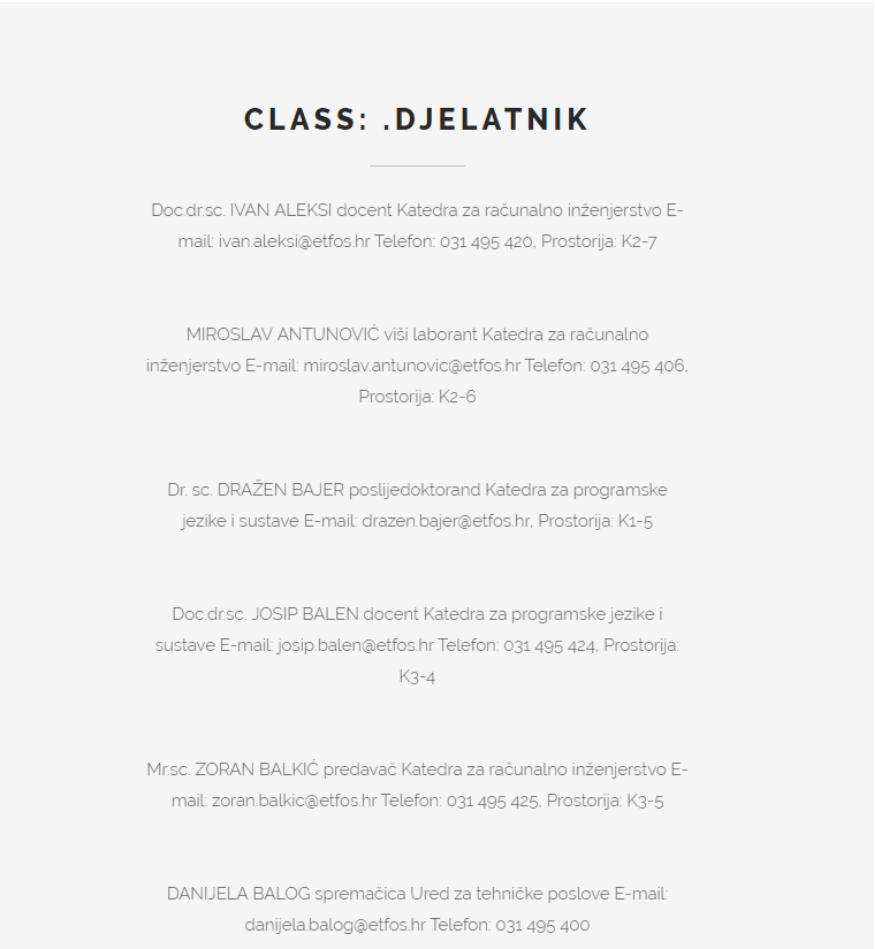
Sl. 4.13. Programski dio ispisa elementa po jednom ili više *id* selektora

4.4. ANALIZA FUNKCIONALNOSTI APLIKACIJE NA STVARNOM PRIMJERU

Aplikaciju je moguće testirati u stvarnom vremenu jer je postavljena na web poslužitelj korisničkih stranica studenta. Proizvoljnim odabirom poveznice i opcija učitavanja dobiju se rezultati nakon par sekundi učitavanja. Na primjeru je odabrana poveznica sa popisom djelatnika Fakulteta elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, te je definirana opcija učitavanja po *class* selektoru koji se vidi u izvoru stranice. Klikom na tipku Scrape nakon par sekundi dobiju se podatci koji zadovoljavaju zadane opcije što je vidljivo na slici 4.14.



Sl. 4.13. Početni korak unosa poveznice i odabira opcije učitavanja



Sl. 4.14. Dobiveni podatci po postavljenoj opciji

4.5. MOGUĆE NADOGRADNJE APLIKACIJE

Ova skripta svakako ima prostora za napredak. Ukoliko će u budućnosti zaživjeti unutar jedne organizacije potrebno je napraviti nekoliko poboljšanja i dodati par funkcionalnosti. Funkcionalnosti koje bi trebalo implementirati:

- Unaprijeđenje sučelja – radi olakšavanja odabira u sučelje se mogu dodati još neke opcije osim osnovnih, iako korisnik ima odabir proizvoljnog elementa, u obliku prijedloga

- Interaktivni prikaz rezultata – dobiveni podatci bi bili označeni/obojani u boju, tablično ili raznihm grafovima kako bi se lakše prepoznali i upotrijebili u daljnjoj proceduri
- Sortiranje po kriterijima – dobiveni rezultati se mogu sortirati po željenim kriterijima u tablicu ili stupce u kojima se na vrhu nalazi definirani uvjet učitavanja
- Izvoz podataka u neki od popularnih formata npr. XML, JSON

5. ZAKLJUČAK

Iako se alati za učitavanje podataka mogu nositi s jednostavnim ili umjerenim zahtjevima za učitavanje podataka, oni nisu preporučeno rješenje ako ste poslovni korisnik koji pokušava steći podatke za istraživanje tržišta ili dobivanje prednosti nad konkurencijom. U modernom poslovnom okruženju proizvodi se velika količina podataka u nestrukturiranom obliku i upravo zbog nedostatka strukture samo mali postotak tih podatka se koristi kao temelj i podrška poslovanju. Kada je zahtjev velik i/ili kompliciran, alati za učitavanje podataka ne ispunjavaju očekivanja. Pomoću gotovih alata mogućnosti prilagodbe su ograničene i automatizacija je gotovo nepostojeća. Alati također dolaze s nedostatkom održavanja, što može biti zastrašujući zadatak. Trenutno dostupne inicijative alata izgledaju prilično primitivno jer uglavnom pružaju funkcionalnosti koje se temelje potpuno ljudskom interakcijom sa preglednikom poput klikanja, pokazivanja i popunjavanja obrazaca. Samostalna izrada je bolja opcija i može biti pravi izbor ako su zahtjevi za podatke ograničeni, a web lokacije koje korisnik želi učitati nisu komplicirane. Ako učitavanje podataka zahtijeva prilagođeni skup postavki, samostalno izrađeni alat to ne može učitati. Koristeći već gotova rješenja korisnik također zahtjeva da vrijeme i učinkovitost budu na visokom nivou. Cijeli ovaj postupak bi mogao imati poteškoća sa ograničenjima jer korisnik, podatke koji sadrže komercijalne informacije, mora prvo dobiti dozvolu od vlasnika podataka jer se u suprotnom krše autorska prava. Stoga je s ovim diplomskim radom postignuta određena automatizacija postupka učitavanja podataka koji se nalaze u nestrukturiranom obliku. U njemu je osmišljen alat koji ima mogućnost učitavanja konkretnih podataka i ispis za daljnje korištenje. Odabirom ponuđenih opcija sužava se izbor i nastoji se filtrirati rezultat. Ono što je cilj ovakvog alata i za što bi ga mogle iskoristiti poslovne organizacije je to da dobiveni podatak bude pretvoren u kvalitetnu informaciju koja je ekonomski iskoristiva. Te informacije ukoliko se iskoriste na pravi način u poslovanju mogu dovesti do stvaranja dodatne vrijednosti te jačanja konkurentske prednosti. Također takva kvalitetna informacija nakon toga može biti pretvorena u znanje te kao vrhunac u mudrost.

LITERATURA

1. Definicija podatka, http://www.enciklopedija.hr/natuknica.aspx?id=48887, pristupljeno 7.6.2017.
2. Definicija informacije, http://www.enciklopedija.hr/natuknica.aspx?id=27405, pristupljeno 7.6.2017.
3. Edgar F. Codd, (1970), *A relationship model of data for large shared data banks,* IBM Research Laboratory
4. UIMA dokumentacija, https://www.ibm.com/developerworks/data/downloads/uima/, pristupljeno 15.6.2017.
5. Shema.org dokumentacija, https://schema.org/docs/documents.html, pristupljeno 15.7.2017.
6. Ryan Mitchell (2015), *Web Scraping with Python: Collecting Data from the modern Web*
7. Screen scrapping definicija, https://www.techopedia.com/definition/16597/screen-scraping, pristupljeno 16.7.2017.
8. OCR definicija, http://www.webopedia.com/TERM/O/optical_character_recognition.html, pristupljeno 4.9.2017.
9. Matthew Turland (2010), *PHP architect's guide to Web Scraping*
10. Report mining definicija, http://ec.europa.eu/research/innovation-union/pdf/TDM-report_from_the_expert_group-042014.pdf, pristupljeno 4.9.2017.
11. Requests for PHP dokumentacija, http://docs.python-requests.org/en/master/, pristupljeno 25.8.2017.
12. GUZZLE dokumentacija, http://docs.guzzlephp.org/en/stable/overview.html, pristupljeno 25.8.2017.
13. HTTPFul dokumentacija, http://phphttpclient.com/docs/, pristupljeno 25.8.2017.
14. Import.io dokumentacija, http://api.docs.import.io/, pristupljeno 10.7.2017.
15. Mozenda dokumentacija, http://www.mozenda.com/api/, pristupljeno 10.7.2017.
16. Octoparse dokumentacija, http://www.octoparse.com/doc-wf/introduction/, pristupljeno 10.7.2017.
17. GREPSR dokumentacija, https://www.grepsr.com/how-it-works/, pristupljeno 10.7.2017.

18. Web Scraping Chrome dokumentacija, http://webscraper.io/documentation, pristupljeno 10.7.2017.
19. HTML dokumentacija, https://developer.mozilla.org/en-US/docs/Learn/HTML, pristupljeno 4.9.2017.
20. CSS dokumentacija, https://developer.mozilla.org/en-US/docs/Learn/CSS, pristupljeno 4.9.2017.
21. M. Otto i T. Jacob, »Twitter Bootstrap, http://www.getbootstrap.com, pristupljeno 4.9.2017.
22. D. Robeli, Osnove programskog jezika PHP, Fakultet strojarstva i brodogradnje, 2002.

SAŽETAK

ALATI ZA UČITAVANJE NESTRUKTURIRANIH PODATAKA

U ovom radu prezentirat će se se teme kao što su tipovi podataka po strukturi, njihovo učitavanje i prikazivanje pomoću gotovih usluga i biblioteka. Osim teorijskog dijela, u radu je prezentiran i konkretan primjer alata za učitavanje podataka iz nekog izvora. Kao rezultat ovakvog alata dobiveni su kvalitetni podatci koji su iskoristivi u praksi. Na temelju uspoređenih usluga, biblioteka i vlastitog alata može se vidjeti razlika u kvaliteti učitavanja nestrukturiranih podataka iz vanjskih izvora te njihovo spremanje u raznim formatima.

Ključne riječi: strukturirani/polustrukturirani podatci, nestrukturirani podatci, UIMA, web scraping alati

ABSTRACT

WEB SCRAPING TOOLS

This paper will present topics such as data types by structure, scraping them and displaying them with completed services and libraries. In addition to the theoretical part, a specific example of a tool for scraping data from a source is presented in the paper. As the result of this kind of tool, quality data was obtained that can be utilized in practice. Based on comparisons of services, libraries and own tools, the difference in the quality of untrusted data can be presented from external sources and their storage in a variety of formats.

Keywords: structured/semistructured data, unstructured data, UIMA, web scraping tools

ŽIVOTOPIS

Slaven Galić je rođen 11. veljače 1991. u Osijeku, Republika Hrvatska. Odrastao i živi u Čepinu sa svojim roditeljima. U Čepinu je završio osnovnu školu Miroslav Krleža te nakon završetka uspisao 2. Gimnaziju u Osijeku. Godine 2009. upisao je preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku, današnji Fakultet elektrotehnike, računarstva i informacijskih tehnologija. Završetkom preddiplomskog studija 2013. godine stekao je titulu diplomiranog pripravnika inženjera računarstva, univ.bacc.ing.comp. Pored fakulteta konstantno je radio na raznim poslovima vezanim za struku i van nje. Aktivno se služi engleski jezikom te poznaje njemački i švedski.

Potpis

_____

Slaven Galić

PRILOZI

I. Prilozi na CD-u

Na pripadajućem CD-u nalaze se sljedeći prilozi:

- Diplomski rad u digitalnom formatu (.docx i .pdf),
- Izvorni kod Web aplikacije