# Uvod

# Prikupljanje podataka

Stalno stvaranje novog sadržaja na internetu čini ga neiscrpnim izvorom podataka. U kontekstu ovog rada prikupljanje podataka odnosi se na dohvaćanje prirodnog jezika sa web mjesta. Društvene mreže poput Facebook-a imaju dokumentiran i otvoren sustav za dohvaćanje i objavljivanje sadržaja što olakšava pristup podacima zanimljivim za obradu. Ako web mjesto nema otvoren programatski pristup svojim podacima onda se koristi web struganje (eng. web scraping). Za izradu programa za prikupljanje podataka korišten je programski jezik Python 2.7. Python je visoki programski jezik opće namjene a glavni naglasak je na čitljivosti koda i jednostavnosti sintakse. Važna značajka Python-a je interaktivno izvođenje programskog koda što je posebno korisno u izradi programa za prikupljanje podataka. Za izradu programa za web struganje koristit će se razvojna cjelina Scrapy.

## Facebook Graph API

Aplikacijsko programsko sučelje (eng. Application Programming Interface, API) je skup određenih pravila i specifikacija koje programeri slijede tako da se mogu služiti uslugama ili resursima nekog složenog programa. Graph API je osnovni način za dohvat podataka iz Facebook platforme. API je baziran na HTTP-u (hypertext transfer protocol) koji se koristi za izradu upita prema podacima kao što su objave, reklame, fotografije i niz drugih zadaća ovisno o potrebi aplikacije. Graph API predstavlja podatke na sljedeći način:

* nodes – objekti kao što su korisnik, fotografija, komentar, stranica
* edges – veze između objekata, fotografije stranice, komentari fotografije
* fields – informacije o objektima kao rođendan korisnika, ime stranice ili fotografije

Budući da je API baziran na HTTP-u moguće je koristiti bilo koji jezik koji sadrži HTTP biblioteku. Za pristup API-ju potreban je dokaz za pristup (eng. access token). Access token je niz znakova koji identificira korisnika, aplikaciju ili stranicu koja pristupa API-ju te ograničava prava pristupa podacima.

### Implementacija u Pythonu

Za dohvaćanje podataka sa API-ja u ovome radu koristiti će se programski jezik Python i HTTP biblioteka urllib2. Osnova korištenja API-ja baziranim na HTTP-u je slanje GET requestova na određeni URL.



Slika 1 Funkcija request\_until\_suceed

Funkcija prima parametar URL (eng. Uniform Resource Locator) te šalje GET zahtjev na taj URL. Ukoliko je odgovor servera 200 (što znači da je zahtjev uspješan) funkcija vraća podatke poslane od strane servera. Ako dođe do greške program se zaustavlja na 5 sekundi prije ponovnog zahtjeva te obavještava korisnika koji URL zahtjev nije uspio i vrijeme greške. Ova osnovna funkcija koristiti će se za sve zahtjeve prema API-ju.

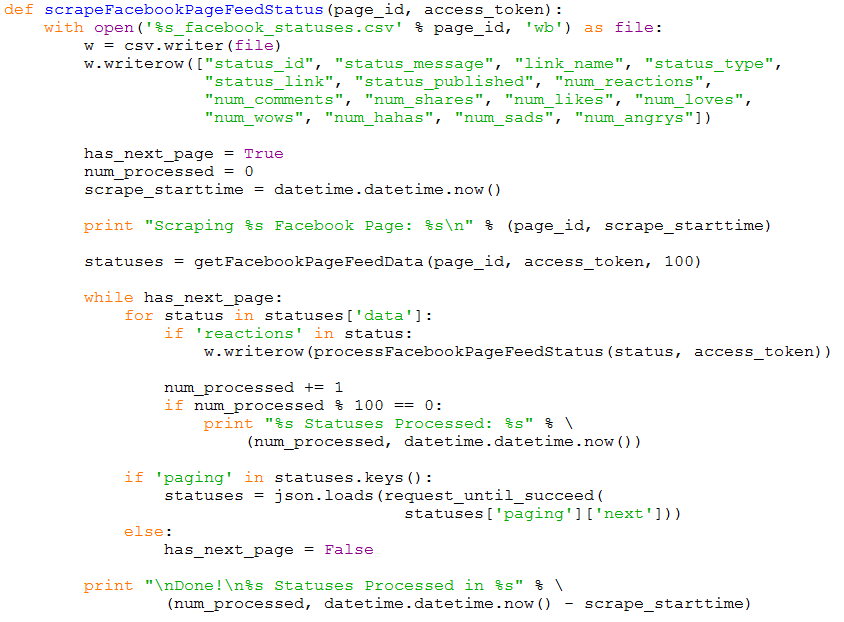
Podaci koje gore prikazana funkcija dohvaća ovise o datom parametru URL. Za dohvaćanje objava i meta-podataka sa neke stranice implementirana je funkcija koja određuje potrebne dijelove URL-a i prosljeđuje niz znakova koji predstavljaju URL funkciji [request\_until\_succeed]. API vraća podatke u JSON (eng. JavaScript Object Notation) obliku. JSON je format za razmjenu podataka, lako se čita i piše te je neovisan o izboru programskog jezika. Koristi konvencije koje su poznate programskim jezicima iz C obitelji što ga čini idealnim izborom za izmjenu podataka. Za manipulaciju JSON tipom podatka koristi se Python-ov modul *json* koji je dio standardne biblioteke.



Slika 2 Funkcija getFacebookPageFeedData

Funkcija [getFacebookPageFeedData] vraća dohvaćene podatke u JSON obliku za daljnju obradu. U varijabli [parameters] se sprema broj koji određuje koliko će se objava dohvatiti i [access\_token] – niz znakova koji služi za autorizaciju i autentifikaciju programa koji šalje zahtjeve prema API-ju. Kada se dohvaća veći broj objava API podatke razdvaja po stranicama. Za "listanje" stranica potrebno je slati zaseban zahtjev za iduću ili prethodnu stranu. URL za navigaciju po stranicama predstavljen je u JSON-u u objektu [paging] koji sadrži [next i previous] ključeve sa odgovarajućim vrijednostima u obliku niza znakova.

Prikupljeni podaci spremaju se u tekstualnu datoteku za daljnju obradu. Odabrani formati je CSV (eng. Comma Seperated Values), podaci se spremaju u polja koja su razdvojena zerezima ili drugim arbitrarnim znakom. Razlog odabira ovog formata je podrška u raznim programima i programskim jezicima te čitljivost i podrška za posebne znakove. Format je također služi za uvoz i izvoz podataka iz baza podataka što može biti korisno kada se veličina datoteka poveća ili se podaci povezuju sa raznim servisima i aplikacijama. Za rukovanje CSV datotekama u implementaciji za dohvat podataka koristi se modul iz standardne Python-ove biblioteke *csv,* prije unosa dohvaćenih podataka program upisuje zaglavlja koja označuju o kojem meta-podatku se radi.



Slika 3 scrapeFacebookPageFeedStatus

Funkcija [scrapeFacebookPageFeedStatus] koristi navedene funkcije za interakciju sa API-jem i procesuiranjem dohvaćenih podataka, obavještava korisnika o vremenu i broju dohvaćenih podataka te ih sprema u CSV datoteku za daljnju obradu.

Parametar *page\_id* je jedinstveni identifikator stranice sa koje želimo dohvatiti objave. Identifikator se lako može saznati tako da iz internetskog pretraživača posjetimo stranicu te iz URL-a isčitamo identifikator.



Slika 4 Jedinstveni identifikator stranice ili grupe

Objave koje se dohvaćaju obično budu vezane za određenu temu a mogu biti objavljivane od strane korisnika učlanjenih na stranicu ili kao u slučaju portala koji imaju službene Facebook stranice objave dolaze jedino od vlasnika stranice dok korisnici ostavljaju komentare na objave. Prije prikupljanja podataka korisno je "ručno" analizirati potencijalnu stranicu da bi se utvrdile neke osnovne karakteristike kao što je učestalost objava, tematika objava i tip objava (primjerice stranice gdje korisnici postavljaju pitanja na određenu temu).

## Web scraping

Ako podaci koje želimo dohvatiti nisu dostupni na druge načine poput API-ja onda se koristi web scraping. Prilikom prikupljanja podataka na taj način koriste se programi koji na temelju obrasca pretražuju određenu web stranicu i dohvaćaju podatke koji odgovaraju obrascu. Programske implementacije web scrapera automatizirano pretražuju web stranice i dohvaćaju sadržaj koji odgovara kodiranom obrascu. Obrazac po kojem se prikupljaju podaci najčešće se određuje prema HTML strukturi mjesta u kombinaciji sa regularnim izrazima (eng. regular expressions). Loša ili promjenjiva struktura HTML-a i sve više dinamičkih web stranica i sadržaja, otežavaju a ponekad i onemogućuju prikupljanje podataka. Program koji se koristi za ovakvu vrstu prikupljanja podataka naziva se web robot (eng. web crawler), kada su dohvaćeni podaci poveznice na druga web mjesta ili stranice koje program prati prema nekim pravilima onda se takav program naziva web pauk (eng. web spider). Web robote i web pauke koriste internetske tražilice da bi indeksirale web i omogućile pretraživanje kakvo svakodnevno koristimo.

Izrada programa za web scraping može biti jednostavan zadatak ukoliko se radi o dohvaćanju manje količine podataka sa malog broja web stranica. Kada se radi o većem broju stranica sa kompliciranijim obrascima za dohvaćanje podataka te potrebom da se web stranice provjeravaju za novi sadržaj ili pojavu novih web stranica onda se potrebno pobrinuti da programski kod bude održiv i čitljiv. Scrapy je razvojna cjelina za izradu web robota i izvlačenje strukturiranih podataka, napisan je u Python programskom jeziku. Može se koristiti za niz aplikacija kao što je rudarenje podataka, arhiviranje i procesiranje podataka, automatski monitoring i testiranje. Dizajn razvojne cjeline omogućuje da koncentracija razvoja bude na izradi pravila po kojima se dohvaćaju podaci. Sustav je modularan tako da je moguće koristiti dodatne ekstenzije i module na dohvaćenim podacima. Službeni održavatelj i developer je Scrapinghub Ltd. koji licencira Scrapy pod BSD licencom, na razvoju sudjeluje 221 developer[Github repozitorij]. Uz sve prednosti koje pruža korištenje razvojne cjeline, Scrapy također ima mogućnost izvođenja u oblaku. Kada se radi web scraping na velikim web stranicama koje sadrže puno podataka, robot šalje veliki broj zahtjeva prema poslužiteljima što može postati nestabilno kada se koristi osobna internetska usluga. Resursi osobnog računala općenito postaju nedovoljni kada proces prikupljanja podataka zahtjeva veće količine memorije i duže vrijeme izvođenja. Scrapy Cloud omogućuje implementaciju zahtjevnijih web robota bez potrebe održavanja infrastrukture koju takvi web roboti zahtjevaju. Prikupljeni podaci se spremaju u high-availability baze podataka i dostupni su u JSON, XML i CSV oblicima.

### Arhitektura sustava

Svaka komponenta zadužena je za određeni dio toka podataka unutar sustava.

* Engine

Centralna komponenta Scrapy-a, zadužena za kontrolu toka podataka kroz ostale komponente. Postavlja okidače za određene događaje unutar sustava.

* Scheduler

Zaprima zahtjeve (eng. requests) od engina i određuje redosljed izvođenja te ih vraća enginu.

* Downloader

Komponenta zadužena za dohvat web stranica, nakon dohvata vraća ih enginu da bi bili prosljeđeni paucima.

* Spiders

Pauci su klase izrađene od strane korisnika, zadužene za ekstrakciju podataka ili postavljanje novih requestova.

* Item pipeline

Komponenta zadužena za daljnje procesiranje podataka (itema) koje parsiraju spideri. Uobičajena namjena im je validacija, čišćenje i pohranjivanje podataka (primjerice u bazu podataka).

Osim navedenih komponenti moguće su implementacije eng. Downloader middlewares i Spider middlewares. Downloader middlewares su specifične kuke (eng. hooks) koje se nalaze između Engine i Downloader komponenti i procesiraju zahtjeve (eng. requests) od Engine-a prema Downloader-u i odgovore (eng. responses) koje idu od Downloader-a prema Engine-u. Ove funkcionalnosti su korisne kada je potrebno izmjeniti response prije nego što je poslan Spider komponenti ili zabranjivanje slanja određenih requestova.

### Spiders

Pauci (eng. Spiders) su klase koje definiraju kako će podaci sa određenih web stranica biti dohvaćeni, uključujući i kako će poveznice unutar tih stranica biti praćene. Ciklus dohvaćanja počinje sa inicijalnim zahtjevima na (eng. requests) zadane URL-ove, te određivanje callback funkcije koja će biti pozvana kada spider primi odgovor (eng. response). Unutar callback funkcije parsira se vraćeni odgovor te vraća parsirane podatke u obliku Item objekta, Request objekta ili iterabilnog tipa koji sadrži neke od navedenih objekata. U callback funkciji određujemo elemente web-a koji želimo dohvatiti. Za pronalazak željenih elemenata moguće je koristiti CSS selektore, Xpath (eng. XML Path Language) ili vanjske biblioteke za parsiranje.

# Obrada prirodnog jezika