Proiect pentru examenul de atestare a competențelor profesionale de informatică

Web Scraping Bot

Elev: Crăciun Alexandru – Emilian

Clasa a XII-a D

Colegiul Național „Vasile Alecsandri”, Galați

Prof. Coordonator: Voinea Mirela

Mai 2022

Cuprins

Considerații generale despre Python

Python este un limbaj de programare dinamic multi-paradigmă, creat în 1989 de programatorul olandez Guido van Rossum. Implementarea de referință a Python este scrisă în *C* și poartă deci numele de *CPython*

Python este un limbaj multifuncțional folosit, de exemplu, de către companii precum Google, Meta Platforms, Spotify sau Netflix pentru dezvoltarea aplicațiilor *web*, însă există și o serie de aplicații științifice sau de divertisment programate parțial sau în întregime în Python.

Popularitatea în creștere, dar și puterea limbajului de programare Python au dus la adoptarea sa ca limbaj principal de dezvoltare de către programatori specializați și chiar și la predarea limbajului în unele medii universitare. Din aceleași motive, multe sisteme bazate pe Unix, inclusiv Linux, BSD și Mac OS X includ din start interpretatorul *CPython.*

Python pune accentul pe curățenia și simplitatea codului, iar sintaxa sa le permite dezvoltatorilor să exprime unele idei programatice într-o manieră mai clară și mai concisă decât în alte limbaje de programare ca C. Un alt avantaj al limbajului este existența unei ample biblioteci standard de metode.

În ceea ce privește paradigma de programare, Python poate servi ca limbaj pentru software de tipul *object-oriented*, dar permite și programarea imperativă, funcțională sau procedurală. Sistemul de tipizare este dinamic iar administrarea memoriei decurge automat prin intermediul unui serviciu „gunoier” (*garbage collector*).

Introducere

Trăim în epoca informațiilor, dar acest fapt nu înseamnă că tot ceea ce putem găsi pe Internet este și folositor, în special când vine vorba despre informațiile profesionale.

În decursul ultimilor ani, pentru a rămâne la curent cu evoluția tehnologiei și aplicațiile ei în diverse industrii, am revenit la articolele publicate pe *https://news.ycombinator.com/*

Cu toate acestea, am descoperit că, în general, doar articolele cu peste 100 de voturi din partea comunității sunt cu adevărat valoroase. Problema este că *Hacker News* nu îți permite să filtrezi articolele publicate la ei pe website.

Astfel, am decis să construiesc un program care să extragă doar titlurile și linkurile articolelor ce au peste 100 de voturi / puncte.

Instalare

Pentru a porni programul se deschide "*WebScraping"*, o scurtătură ce va lansa aplicația în terminal, afișând titlurile articolelor, voturile și linkurile pentru fiecare dintre acestea.

Prezentarea aplicației

Proiectul folosește modulele *requests*, pentru a extrage codul sursă al site-ului web și *BeautifulSoup*, pentru a prelucra codul sursă HTML extras într-o formă ușor de citit.

Codul este alcătuit din 3 mecanisme: extragerea codului sursă HTML, prelucrarea codului sursă pentru a afișa doar articolele cu peste 100 voturi și afișarea în ordine a articolelor cu cele mai multe voturi.

Primul mecanism folosește funcții din modulele menționate anterior pentru a extrage codul sursă de tip HTML, pe fiecare dintre primele 3 pagini ale site-ului și a-l reține într-un obiect de tip ***BeautifulSoup***:

```
21  for page in range(1,4):
22      res = requests.get(f'https://news.ycombinator.com/news?p={page}')
23      soup = BeautifulSoup(res.text, 'html.parser')
```

În continuare, se extrag titlurile și sub-textul de la fiecare dintre articole, ce conține și numărul de voturi:

```
25      links = soup.select('.titlelink')
26      subtext = soup.select('.subtext')
```

1. ▲ The Game: A continually-run D&D campaign, since 1982 (thegamednd.com)
126 points by gaws 2 hours ago | hide | 20 comments

Al doilea mecanism este reprezentat de funcția ***create_custom_news()***, ce primește ca parametri ***links*** și ***subtext***.

```
4   def create_custom_news(links, subtext):
5       news = []
6       for indx, item in enumerate(links):
7           title = links[indx].getText()
8           href = links[indx].get('href', None)
9           vote = subtext[indx].select('.score')
10
11          if len(vote):
12              points = int(vote[0].getText().replace(" points", ''))
13
14              if points > 99:
15                  news.append({'title': title, 'link': href, 'votes': points})
16
17      return news
```

Articolele urmează a fi returnate prin variabila ***news***, o listă de dicționare. Fiecare dicționar din listă reprezintă un articol, având în componența sa titlul articolului, linkul și numărul de voturi.

```
6       for indx, item in enumerate(links):
7           title = links[indx].getText()
8           href = links[indx].get('href', None)
9           vote = subtext[indx].select('.score')
```

Respectivele componente ale unui articol sunt extrase pe baza tagurilor prezente în codul sursă HTML, titlul fiind textul simplu, linkul, având tagul ***href*** în codul sursă, iar voturile fiind din clasa ***score***.

```
▼ <td class="title">
    <a href="https://thegamednd.com/the-game-out-of-game/" class="titlelink">The Game: A
    continually-run D&D campaign, since 1982</a> == $0
```

```
▼ <td class="subtext">
    <span class="score" id="score_31230967">126 points</span> == $0
```

Dar, se observă că unele articole nu au niciun vot.

```
22.  BuildZoom (better way to build custom homes) Is hiring a Growth Associate (lever.co)
     8 hours ago | hide
```

Doar pentru cele care au un număr de voturi vom transforma variabila de tip string (e.g. “126 points”) într-un integer, folosind metoda ***replace().***

```
11          if len(vote):
12              points = int(vote[0].getText().replace(" points", ''))
```

Acum că variabila ***points*** este de tip integer putem verifica dacă numărul de voturi al articolului curent este cel puțin egal cu 100. Dacă acesta este cazul, vom adăuga articolul sub forma unui dicționar în lista articolelor din pagina curentă, urmând să returnăm lista prin variabila ***news***.

```
14              if points > 99:
15                  news.append({'title': title, 'link': href, 'votes': points})
16
17      return news
```

În final, adăugăm rezultatul funcției ***create_custom_news() –*** *articolele cu peste 100 de voturi din pagina curentă,* la lista cu toate articolele extrase până atunci, ***all_news***.

```
28      all_news.extend(create_custom_news(links, subtext))
```

Acest proces se repetă pentru fiecare dintre primele 3 pagini ale site-ului.

```
19  all_news = []
20
21  for page in range(1,4):
22      res = requests.get(f'https://news.ycombinator.com/news?p={page}')
23      soup = BeautifulSoup(res.text, 'html.parser')
24
25      links = soup.select('.titlelink')
26      subtext = soup.select('.subtext')
27
28      all_news.extend(create_custom_news(links, subtext))
```

Ultimul mecanism este cel de sortare și afișare a articolelor pentru ca acestea să pot fi citite ușor. Folosim metoda ***sort()***, ordonând lista în funcție de numărul de voturi a fiecărui articol, în ordine descrescătoare.

În cele din urmă, afișăm articolele în consolă într-un format ușor de citit:

```
30  all_news.sort(key = lambda a: a['votes'], reverse = True)
31
32  print()
33  for item in all_news:
34      print(item['votes'], 'votes')
35      print(item['title'])
36      print(item['link'], end = '\n\n')
```

Codul Sursă

scrape.py

```python
import requests
from bs4 import BeautifulSoup

def create_custom_news(links, subtext):
    news = []
    for indx, item in enumerate(links):
        title = links[indx].getText()
        href = links[indx].get('href', None)
        vote = subtext[indx].select('.score')

        if len(vote):
            points = int(vote[0].getText().replace(" points", ''))

            if points > 99:
                news.append({'title': title, 'link': href, 'votes': points})

    return news

all_news = []

for page in range(1,4):
    res = requests.get(f'https://news.ycombinator.com/news?p={page}')
    soup = BeautifulSoup(res.text, 'html.parser')

    links = soup.select('.titlelink')
    subtext = soup.select('.subtext')

    all_news.extend(create_custom_news(links, subtext))

all_news.sort(key = lambda a: a['votes'], reverse = True)

print()
for item in all_news:
    print(item['votes'], 'votes')
    print(item['title'])
    print(item['link'], end = '\n\n')
```

Posibilități de a îmbunătăți aplicația

Aplicația poate fi îmbunătățită prin integrarea unei interfețe vizuale sau utilizarea unui modul pentru a trimite articolele rezultate pe adresa de e-mail a utilizatorului.

De asemenea, poate fi mărit numărul de pagini indexate și, totodată, poate fi utilizat un framework precum Scrapy pentru a facilita procesul de analiză a numărului mai mare de date.

Bibliografie

1. *https://ro.wikipedia.org/wiki/Python/*
2. *https://www.crummy.com/software/BeautifulSoup/bs4/doc/*
3. *https://docs.python-requests.org/en/latest/*
4. *https://news.ycombinator.com/*
5. *https://www.w3schools.com/cssref/css_selectors.asp*