Fakulteta za elektrotehniko,
računalništvo in informatiko

Poročilo za projektno nalogo pri predmetu

Principi programskih jezikov

2021/22

Avtorji:

Vid Beranič

Žiga Hace

Urška Dobnik

Mentor:

dr. Peter Kokol, univ. dipl. inž. rač. in inf.

Maribor, 3. 6. 2022

Vsebina

Kazalo slik

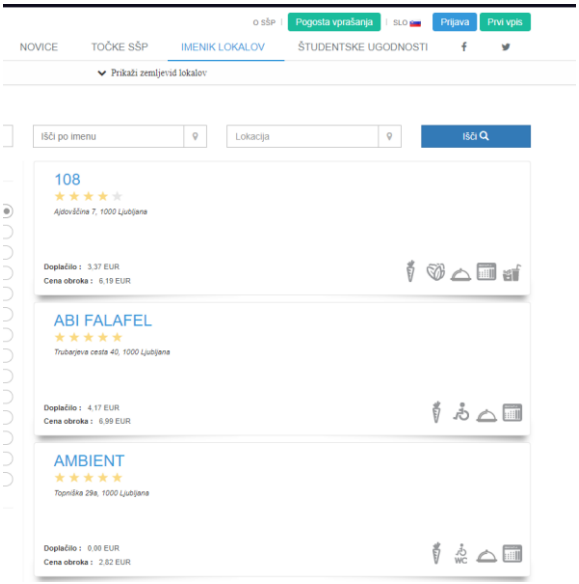
Uvod

Smo skupina Beasty in za naš projekt smo si izbrali prikaz restavracij. Vodja naše skupine je Žiga Hace.

Vir podatkov

Naša aplikacija je predvsem namenjena za študente, saj prikazuje restavracije v katerih nudijo hrano na študentske bone. Za spletno stran smo si izbrali spletno stran Študentska prehrana (Slika 1). Z nje smo pridobili podatke o restavracijah, menijih, tipu hrane, ceni in drugo. Spletna stran nudi seznam restavracij, na katerem so zapisane vse restavracije, ki nudijo menije na študentske bone. (Slika 2)

Slika 1: Domača stran spletne strani Študentska prehrana



Slika 2: Imenik restavracij na Študentska prehrana

Program za razčlenitev podatkov

Izbrana spletna stran ni nudila podatkov v željenem podatku (JSON), zato smo se odločili, da bomo napisali svoj program za razčlembo podatkov.

Odločili smo se, da bomo napisali razčlenjevalnik v programskem jeziku Python. Za Python smo se odločili, saj velja za najboljši programski jezik za »web scraping« oziroma razčlenjevanje podatkov iz spletnih strani, prav tako je enostaven in uporablja knjižnico BeautifulSoup, katera je za razčlenjevanje podatkov iz strani zelo hitra in dokaj enostavna za razumet.

Python program

Pred začetkom pisanja programa smo morali naložit potrebne pakete/knjižnice. Za nalaganje BeautifulSoup knjižnice smo uporabili ukaz »pip install beautifulsoup4«.

na začetku programa smo uvozili BeautifulSoup ter requests, ter prekopirali URL naslov s katerega bomo uporabljali podatke. (Slika 3)

```
from bs4 import BeautifulSoup

import requests

URL = "https://www.studentska-prehrana.si/sl/restaurant"
page = requests.get(URL)
```

Slika 3: Začetek programa

Da lahko iz spletne strani izberemo ustrezne elemente z iskanimi podatki smo stran gledali v njeni HTML obliki, bližnjica na tipkovnici za to je »Ctrl + Shift + C«. na strani je vsaka restavracija zapisana v elementu div z razredom »restaurant-row«, zato smo izbrali vse te elemente z ukazom »restaurants = soup.findAll('div', class_="restaurant-row")«, ter jih shranimo v spremenljivko restaurants, prvi parameter ukaza findAll je iskani element, drugi pa elementov razred. Iz vsakega najdenega elementa rabimo podatke, zato s for zanko gremo čez vse. (Slika 4)

Želimo samo restavracije iz mesta Maribor, zato s stavkom if preverimo, če ima atribut data-city elementa restaurant vrednost MARIBOR, če ima, si z iskanjem po atributih shranimo še vse druge podatke (ime lokala, cena z bonom, cena brez bona, koordinate, naslov).

```
res =[]
restaurants = soup.findAll('div', class_="restaurant-row")

for restaurant in restaurants:
    if(restaurant['data-city'] == 'MARIBOR'):
        links = restaurant.find('a')
        imeLokala = links.get_text('|', strip=True)
        cenaBrezBona = restaurant['data-cena']
        cenaZBonom = restaurant['data-doplacilo']
        latitude = restaurant['data-lat']
        longitude = restaurant['data-lon']
        loc = [float(latitude), float(longitude)]
        location = restaurant['data-naslov'] + ', Maribor'
```

Slika 4: iskanje vseh elementov z razredom, ter pridobivanje osnovnih podatkov

Ti podatki so bili zelo osnovni za pridobit saj so bili zapisani kot atribut v glavi vsakega elementa. Vsaka restavracija ima tudi povezavo na novo stran na kateri so prikazani, link za to sem pridobil z iskanjem elementa »a«, iz dobljenega pa vrednost pod atributom href. Nato sem pridobil stran iz dobljenega vira, podatki za meni, delovne ure in vrsta jedi so pridobljeni iz te nove strani

```
links = restaurant.find('a')
link = links['href']
secondPage = requests.get("https://www.studentska-prehrana.si" + link)
soup = BeautifulSoup(secondPage.content, "html.parser")
```

Slika 6: pridobivanje nove strani za vsako restavracijo posebaj

```
delo = workDay.get_text().strip().replace(" ", "")
delo1 = delo.splitlines()
```

Slika 5: pridobivanje besedila v spremenljivki workDay

Odpiralni čas sem pridobil z iskanjem elementa div z razredom »col-md-12 text-bold« (ukaz na Slika 4) in ga shranil v spremenljivko workDay, nato pa iz njega z ukazom na (Slika 5) besedilo, strip() izbriše »newline« ter »tab« na začetku in koncu besede, replace (" ", "") odstrani presledke, splitlines() pa loči besedo v spremenljivki delo v array, glede na »newline«. Vsako besedo, ki je v spremenljivki dl, ločimo na besedo, ki je pred znakom »:«, ter na besedo za znakom, pred znakom je dan v tednu, za znakom pa delovni čas. Nato v prej določeno polje napolnim z ustreznim časom glede na to kateri dan v tednu je pred znakom »:«.

Naslednji iskani podatek pa je jedilnik restavracije. Vse jedi se nahajajo v elementu div z razredom »shadow-wrapper«, zato uporabimo ukaz findAll, kot smo ga na sliki (Slika 4). Da ne pride do napak v programu preverimo, če restavracija sploh ima vnesene jedi.

```
for dl in delo1:
    if(len(dl) != 0):
        ura = dl.split(":",1)
        if(ura[0] == "Nedelja/Prazniki"):
            delovniCas[6] = ura[1]
        elif(ura[0] == "Sobota"):
            delovniCas[5] = ura[1]
        else:
            delovniCas[0] = delovniCas[1] = delovniCas[2] = ura[1]
            delovniCas[3] = delovniCas[4] = ura[1]
```

Slika 7: pridobivanje delovnih ur restavracije

```
if(len(jedi) == 1 and "Lokal nima vpisanih menijev." in jedi[0].get_text()):
    continue
```

Slika 8: if stavek za ugotavljanje, če ima restavracije vnesene jedi

Čez vse elemente se sprehodimo s for zanko. V vsaki iteraciji zanke imamo jed. ime jedi je shranjeno v elementu strong z razredom »color-blue«, zato ga pridobimo z »jed.find('strong', class_='color-blue')«, priloge pa najdemo z »jed.findAll('li'). Ime hrane ter polje jedDodatek shranim v dvodimenzionalno polje meni.

```
jediNaMeniju = []
jedi = soup.findAll('div', class_="shadow-wrapper")
if(len(jedi) == 1 and "Lokal nima vpisanih menijev." in jedi[0].get_text()):
    continue

for jed in jedi:
    hrana = jed.find('strong', class_='color-blue')
    opis = jed.findAll('li')
    jedDodatek = []
    for opisek in opis:
        jedDodatek.append(opisek.get_text().strip())
    jediNaMeniju.append([hrana.get_text()[2:].strip(), jedDodatek])
```

Slika 9: postopek pridobivanja menija restavracije

Kot zadnji podatek pa nas zanima katero vrsto hrane restavracija nudi, saj se včasih želimo odločit kam želimo it jest glede na kakšno hrano restavracija ponuja ali pa želi vegetarijanec jest in ga zanima katere restavracije imajo vegetarijanske menije. Jedi na jedilniku na Študentska prehrana imajo označeno če je jed solata, pizza, meso, riba, mešano, hitra hrana in ali če je celiakiji prijazen obrok ter vegetarijansko. Ustvarili smo dvodimenzionalno polje ponudbaPoVrsti ter vanj dali te označbe, ter vse nastavili na false (primer: ponudbaPoVrsti = [['Meso', False], ['Vegetarijansko', False], …).

Nato v for zanki kjer gremo čez vsako ponudbo v meniju z »jed.find('img', class_='pull-right')« najdemo img element in si ga shranimo pod spremenljivko vrsta. Img element ima atribut »title« v katerem piše, če je solata, meso, pizza, … Z še eno for zanko gremo čez vse elemente v prej ustvarjenem polju ponudbaPoVrsti, v vsaki iteraciji zanke primerjamo vrednost iz polja na poziciji 0 (['Meso', False], ['Vegetarijansko', False], 'Meso' in 'Vegetarijansko' sta pozicija 0) z vrednostjo v atributu vrsta[»title«], če je vrednost enaka spremenimo vrednost na poziciji 1 iz false v true.

```
if(jed.find('img', class_='pull-right')):
    vrsta = jed.find('img', class_='pull-right')
    for vrstaJedi in ponudbaPoVrsti:
        if(vrsta['title'] == vrstaJedi[0]):
            vrstaJedi[1] = True
```

Slika 10: postopek preverjanja katere vrste jedi ponuja restavracija

S tem smo dobili nekatere vrste jedi, ne pa vseh. Uporabnikom strani še želimo dat opcijo za iskanje restavracij po vrstah jedi kot so burger, špageti, lazanja, burek in podobno, tega pa stran Študentska prehrana ne nudi, zato v for zanki kjer gremo čez vse jedi restavracije preverimo z if in elif stavkom, če naslov jedi vsebuje niz burger za preverjanje če restavracija ima burger v ponudbi, če je niz v naslovu jedi najden spremenimo na pravi poziciji v polju ponudbaPoVrsti iz false na true, enako storimo za iskanje testenin, sendvič, juha …

```
if("burger" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[9][1] = True
elif(("testenine" or 'špageti') in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[8][1] = True
elif("sendvič" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[10][1] = True
elif("juha" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[11][1] = True
elif("burek" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[12][1] = True
elif("raca" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[13][1] = True
elif("pišč" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[14][1] = True
elif("svinj" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[15][1] = True
elif("gove" in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[16][1] = True
elif(("lazanja" or "lasagana") in hrana.get_text()[2:].strip().lower()):
    ponudbaPoVrsti[17][1] = True
```

Slika 11: iskanje še drugih vrst jedi

Pretvorba v JSON format

Vse željene podatke smo iz strani razčlenili, samo v json file še jih moramo pretvoriti. Ustvarimo spremenljivko data, ki je tipa Dictionary, ki vsebuje sete vrednosti in ključa (AskPython, 2022). V data vstavimo vse dobljene podatke kot vrednost in jim damo ključ, ki je ime pod katerim se bo vrednost v JSON datoteki shranila. Na koncu še ustvarjeno spremenljivko data dodamo polju res, katerega smo inicializirali na začetku programa.

```
data = {'ime': imeLokala, 'lokacija': location, 'cenaSStudentskimBonom': cenaZBonom,
 'cenaBrezStudentskegaBona': cenaBrezBona, 'delovniCas': delovniCas, 'loc': loc,
  'ponudbaPoVrstiHrane': ponudbaPoVrsti, 'meni': jediNaMeniju}
res.append(data)
```

Slika 12: shranjevanje dobljenih podatkov v spremenljivko data

Ko pridobimo vse podatke za vse restavracije in jih kot data dodamo v polje res, še to zapišemo v datoteko JSON z ukazom prikazanem na sliki desno. (Slika 13)

```
with open('restavracije.json', 'w', encoding='UTF-8') as f:
    json.dump(res, f, indent=8, ensure_ascii=False)
print("Created Json File")
```

Slika 13: shranjevanje polja z rezultatom v datoteko tipa json

Viri

https://www.studentska-prehrana.si/sl (3. 6. 2022)

https://www.promptcloud.com/blog/best-programming-language-for-web-scraping/ (3. 6. 2022)

https://www.askpython.com/python/dictionary/python-dictionary-dict-tutorial (3. 6. 2022)