**Universitatea POLITEHNICA din Bucureşti**
**Facultatea de Inginerie Medicală**
Strada Gheorghe Polizu, nr. 1-7, sector 1, București
Tel.: +40 21 4029039, Fax: +40 21 4029038, corpul F

# PROIECT DE SEMESTRU

## Formarea unei baze de date de medici

**Autor:** Diana BEJINARU
Vladislav GHIDIRIM

**Cadru didactic îndrumător:** Anamaria DUMITRESCU

**București**
**Mai 2023**

# INTRODUCERE

În lumea digitală de astăzi, datele devin din ce în ce mai valoroase. Dacă ne uităm în jur, vom vedea un număr mare de servicii diferite care încearcă să ne îmbunătățească viața. De la giganți precum Google până la startup-uri și mici proiecte pilot, toate aceste servicii funcționează cu date. În centrul oricărei sarcini pe care o mașină sau o persoană trebuie să o rezolve astăzi se află datele.

Pe vremuri, internetul era o mică rețea americană pentru câteva sute de oameni, unde aproape toată lumea se cunoștea. Acum este o structură informațională gigantică. Este practic imposibil să controlezi fluxul de informații aici. Într-un minut pe internet se fac, de exemplu, peste 3 milioane de postări noi pe Facebook, peste 300 de mii de mesaje pe Twitter, peste 30 de mii de recenzii de cărți și achiziții.

Toată această varietate de informații este acum utilizată în mod activ nu numai de oameni, ci și de organizații pentru a îmbunătăți eficiența activităților lor. Viața oamenilor se schimbă atât de rapid acum, încât datele cu caracter personal tradiționale pur și simplu nu sunt suficiente pentru a evalua comportamentul unui împrumutat (dacă este o bancă). Într-o lume în schimbare rapidă, ele sunt pur și simplu depășite și apare necesitatea apelării la datele „externe”. Este important ca băncile să analizeze ce scriu utilizatorii în rețelele sociale și site-urile web. Evaluarea nivelului de alfabetizare, vocabularul, subiectele mesajelor publicate.

Cu toate acestea, pentru a lucra eficient cu aceste informații și pentru a implementa sarcinile cerute de companii și oameni, datele trebuie extrase, procesate și structurate. O persoană, după ce s-a uitat la o pagină web, determină imediat cu ușurință secțiunea necesară și semnificativă, dar pentru un computer, înțelegerea ce tip de text trebuie procesat, cum să se separe acest text de publicitate, titluri inutile, link-uri este o sarcină destul de dificilă.

Pe măsură ce fluxul de informații crește, se dezvoltă posibilitățile de utilizare a acestor informații în sarcini aplicate, se dezvoltă abordări tehnice, unite prin termenul general de „**web crawling**” sau „**web scraping**”. Acestea sunt concepute pentru a colecta informații de pe Internet și a le pregăti pentru procesarea automată [1].

Datorită numărului mare de instrumente și biblioteci accesibile care oferă implementări eficiente ale multor funcționalități necesare, web scraping este un proces destul de simplu în general. Abilitatea de a trimite solicitări HTTP personalizate cu diferite antete și încărcări utile este o caracteristică standard a majorității programelor de scraping web.

Web scraping este o tehnică de conversie a datelor web nestructurate în date structurate care pot fi stocate și analizate într-o bază de date centrală sau foaie de calcul (Sirisuriya, 2015). Această metodă este utilizată în diferite industrii, cum ar fi securitatea cibernetică, poate fi utilizată pentru a determina prețurile și costurile de producție prin evaluarea datelor care au fost colectate. Reclamele pot fi create pentru a face publicitate produselor către diferiți utilizatori, în funcție de datele acestora, cum ar fi locația și setările cookie-urilor.

Metoda este menită să înlocuiască metodele învechite de colectare a datelor, deoarece mai multe companii caută să recupereze noile tendințe care sunt urmate de consumatori (Hillen,2019) [2].

Procesul de răzuire a datelor de pe Internet poate fi împărțit în doi pași succesivi; achiziționarea de resurse web și apoi extragerea informațiilor dorite din datele achiziționate. Mai exact, un program de web scraping începe prin alcătuirea unei cereri HTTP pentru a achiziționa resurse de la un site web vizat. Această solicitare poate fi formatată fie într-o adresă URL care conține o interogare GET, fie într-o bucată de mesaj HTTP care conține o interogare POST [3].

O ilustrație foarte simplă pentru răzuirea web este prezentată în Fig.1.

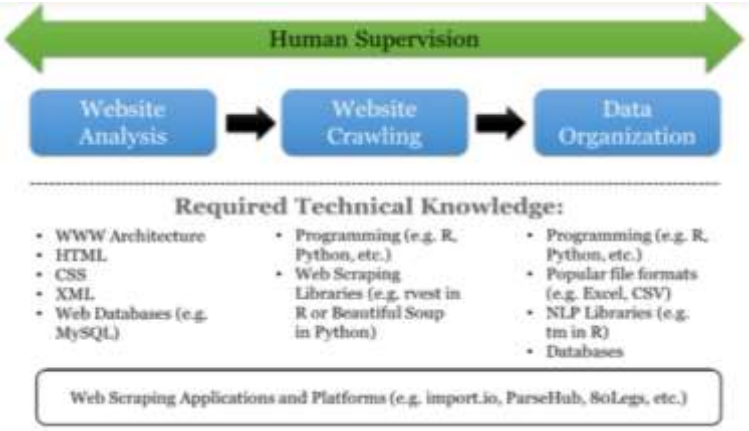



**Fig.1.** Web Scraping (Krotov și Tennyson 2018)

După procesul de răzuire urmează etapa de transformare: Acum că rămân doar datele relevante, acestea pot fi convertite într-un format structurat pentru prezentare sau stocare. Folosind datele stocate, putem aduna informații care pot ajuta business intelligence să ia o decizie mai bună și multe altele [2].

# CAPITOLUL 2. PYTHON PENTRU WEB SCRAPPING

Unul dintre motivele pentru care python este o alegere bună pentru web scraping este că python este unul dintre cele mai populare limbaje, având o comunitate imensă care vă poate ajuta în cazul în care vă confruntați cu orice problem, în timpul programării, este destul de probabil ca altcineva să fi avut aceeași problemă și să fi rezolvat.

Un alt motiv pentru care python este o alegere fantastică pentru web scraping (sau pentru a învăța orice abilitate de programare) este că codul Python este foarte simplu de citit.

Java:

```
class HelloWorldApp {
  public static void main(String[] args) {
    System.out.println("Hello World!");
  }
}
```

C++:

```
#include <iostream>

int main()
{
  std::cout << "Hello, world!";
}
```

Python:

```
Print "Hello, world!";
```

**Fig.2.1.** Exemplu de cod în diferite limbi

## 2.1.Utilizarea și Aplicarea Web Scraping

Web Scraping implică crearea și implementarea a două programe software: un crawler și un scraper. Crawler-ul descarcă datele de pe Internet într-o manieră sistematică; scraperul extrage apoi informații importante în forma sa brută din datele descărcate, le codifică și le stochează într-o bază de date sau într-un fișier conform unei structuri și format definite de utilizator. Acest nou fișier este apoi evaluat în moduri, pe care prezentarea inițială a datelor pe internet nu le permite.

Web Scraping este utilizat pe scară largă în multe domenii diferite ale informaticii cum ar fi: Business Intelligence, AI, Data Science, Big Data, Cloud Computing și Cyber Security.

**Business Intelligence**

1. Evaluarea concurenței de afaceri.
2. Popularitatea site-ului web în ceea ce privește traficul sau interacțiunea utilizatorului.

3. Lucrurile care sunt create/schimbate sunt unice sau esențiale .

4. Defecte de securitate existente în site-uri web

**AI**

1. Tehnicile moderne sunt capabile să proceseze cantități masive de informații într-un timp scurt.

2. Internetul oferă acces la un volum mare de material nestructurat sau neetichetat, care este greu de obținut de către oameni din cauza lipsei de expertiză a surselor de informații disponibile.

3. Utilizând algoritmi de inteligență artificială, sistemul este capabil să diferențieze informațiile mai multor persoane cu același nume .

4. Un Web-Crawler este un tip de bot care se accesează cu crawlere printr-o colecție de site-uri web sau pe întregul internet. Web-crawlerele sunt denumite și Webspiders. În timp ce accesarea cu crawlere a întregului internet este prea mult pentru un asistent personal, un bot poate accesa cu crawlere câteva site-uri și aduna informații.

**Data Science**

1. Preluarea unui tabel interesant de pe o pagină Wikipedia pentru a efectua o analiză statistică.

2. Obținerea unei liste de proprietăți de pe un site web imobiliar pentru a crea o vizualizare geografică atrăgătoare.

3. Culegerea funcțiilor suplimentare pentru a îmbunătăți setul de date folosind date culese din pagini web,cum ar fi datele meteorologice care prezic vânzările de băuturi răcoritoare.

**Big Data**

Termenul de „date mari” discută setul divers de tehnici de colectare și analiză a datelor în metode care anterior erau de neimaginat înainte de apariția computerelor personale contemporane. Datorită colectării automate a informațiilor din paginile web, web scraping este o metodă pe care o examinează specialiștii în big data. Pot exista zeci de mii de variabile într-un set de date web scraping, dar pot exista și zeci de mii de exemple într-un set de date mai mic și mai ușor de gestionat în doar câteva ore folosind web scraping.

Datorită cantității masive de date diverse generate zilnic pe WWW, web scraping este recunoscută pe scară largă ca o tehnică eficientă și puternică pentru colectarea unor cantități mari de date. Pentru a se adapta la o varietate de scenarii, tehnicile moderne de scraping online au evoluat de la operațiuni manuale, ad-hoc, la utilizarea unor sisteme complet automatizate capabile să transforme pagini web întregi în seturi de date bine organizate.

**Cloud Computing**

O tehnologie sofisticată, cum ar fi cloud computing, permite unui web scraper să funcționeze mai eficient. Razuitorul web propus, care este prezentat în **Fig.2.2.**, extrage datele de pe web într-o manieră de încredere. Cloud computing permite accesul la cerere la resursele de stocare și calculul elastic într-un mod dispersat [2].

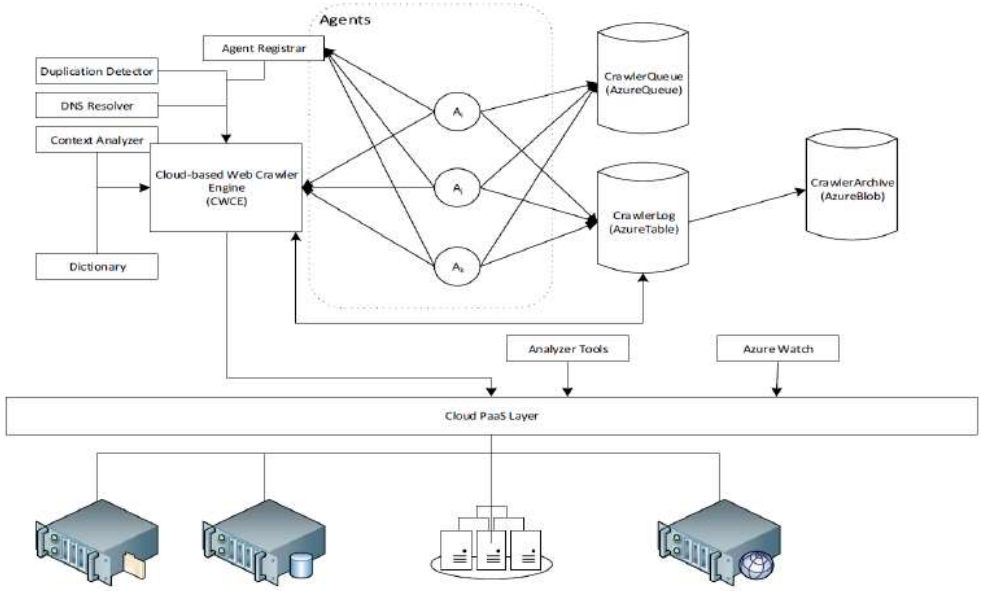



**Fig 2.2.** O arhitectură a crawler-ului web bazat pe cloud propus

## 2.2.Viitorul Web Scraping

Pe măsură ce computerele devin din ce în ce mai puternice în medie, potențialul pentru algoritmi de scraping web mai complecși și inteligenți continuă să crească.

Un aspect important va fi utilizarea sporită a inteligenței artificiale și a învățării automate în tehnicile de web scraping. Utilizarea învățării automate ar putea duce la tehnici de scraping care „învață” automat din experiență și se pot adapta la o varietate mai largă de site-uri.

În cazul în care web scraping continuă să crească ca industrie, va fi important ca orice afacere care nu a prins să se alăture în curând, atât în ceea ce privește protejarea propriilor date pe care nu doresc să le acceseze, cât și în investiția în propriile activități de scraping web. Companiile care nu reușesc să se ridice la standardele moderne așteptate vor deveni adesea învechite și vor fi respinse de consumatori.

## 2.3.Concluzii

Extragerea datelor web ascunse este o provocare majoră în zilele noastre, din cauza naturii autonome și eterogene a conținutului web ascuns, motorul tradițional de stres a devenit acum o modalitate ineficientă de a căuta acest tip de date [4].

Această metodă este un instrument extrem de utilă în era informațională și una esențială în diferite domenii pentru orice companie care dorește să își mențină prezența online, ceea ce în sine este foarte necesar pentru a supraviețui pe piața actuală. Una dintre cele mai bune limbaje pentru implementarea acestuia ar fi Python, datorită curbei sale rapide de învățare și sintaxei puternice [2].

# CAPITOLUL 3. DESCRIEREA BAZEI DE DATE

Datele folosite în această lucrare aparțin bazei de medici al clinicii Medicover din Iași, România. Aceasta conține o listă cu 52 de medici, aparținând mai multor specializări: Alergologie pediatrică, Alergologie și imunologie clinică, Cardiologie, CT, Dermatologie, Diabetologie, Ecografie, Endocrinologie, Epidemiologie, Gastroenterologie, Ginecologie, Medicină generală. Medicina muncii, Neurologie, ORL, Ortopedie, Ortopedie pediatrică, Pediatrie, Pneumologie, Psihologie, RMN și Urologie.

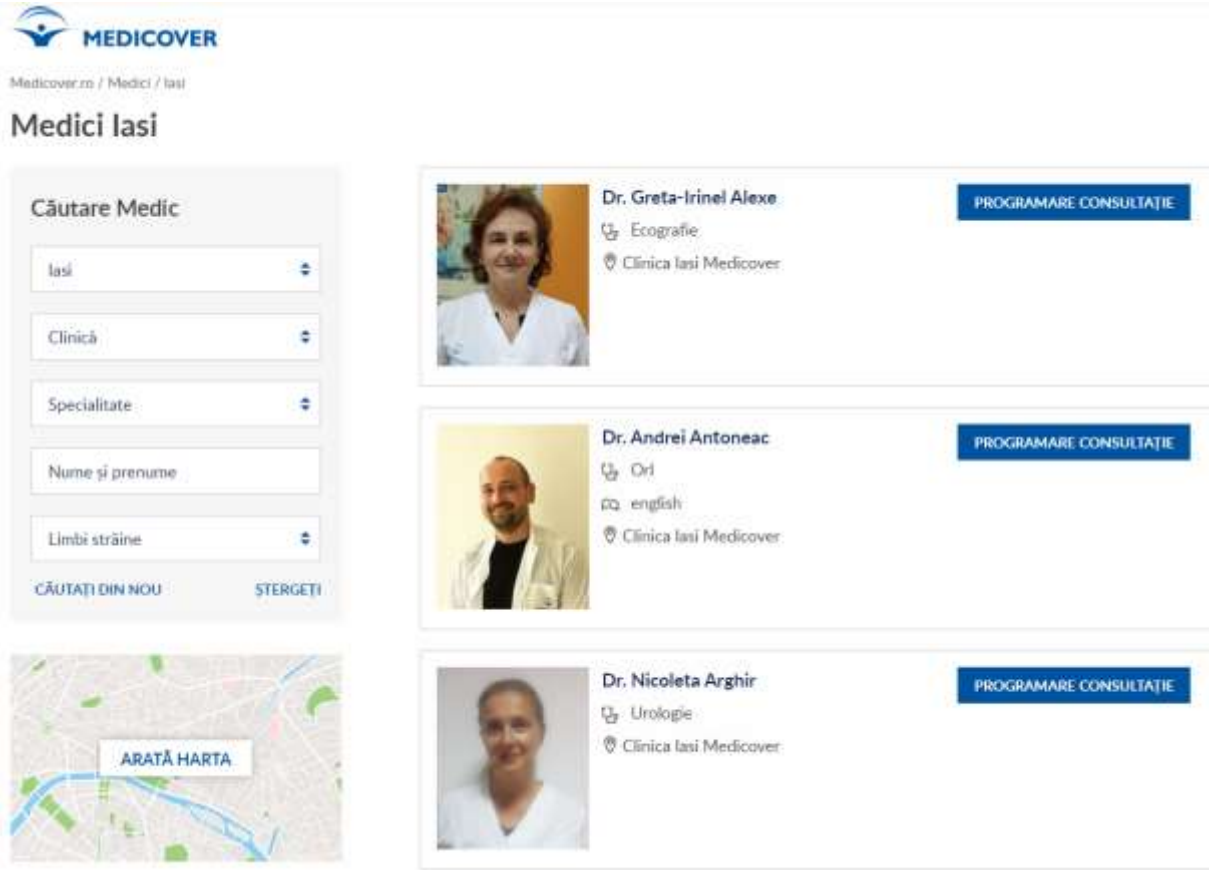



**Fig.3.** Site-ul clinicii Medicover cu lista medicilor din Iași

# CAPITOLUL 4. DESCRIEREA METODEI PROPUSE

Obiectivul acestei lucrări este de a extrage automat lista de medici din județul Iași de pe site-ul web al clinicii Medicover și de a o insera într-un fișier Excel. Pentru aceasta s-a utilizat limbajul de programare Python.

Primul pas este instalarea bibliotecii Beautiful Soup care permite analizarea documentelor de tip HTML și XML. Aceasta conține funcții intuitive care ajută la explorarea unui fișier HTML.

De asemenea, este necesară și instalarea bibliotecii Pandas pentru manipularea și analiza de date. Biblioteca aceasta pune la dispoziție funcții pentru prelucrarea tabelelor numerice și grafice.

Următorul pas este crearea unei variabile care conține adresa web (link-ul) site-ului Medicover. Pe lângă aceasta, au fost create alte două variabile de tip listă, în care urmează să fie introduse datele corespunzătoare numelor medicilor, respectiv specializarea lor.

Pentru a avea acces la numele tuturor numele doctorilor, trebuie inspectată sursa paginii web a site-ului. Deoarece numele medicilor aparținând clinicii din Iași sunt împărțite în 4 subpagini, adresa URL a fiecăreia dintre ele va avea o terminație diferită.

Astfel, se parcurge fiecare subpagină și se creează un obiect de tip BeautifulSoup cu adresele fiecăreia dintre ele. Apoi se caută cu ajutorul funcției *soup.find* etichetele de tip *div* din pagina HTML a fiecărei subpagini care conțin toate numele cu specializarea medicilor, după care sunt introduse într-o variabilă nouă cu rezultate. De asemenea, din aceste secțiuni ale etichelor *div*, se caută alte etichete incluse în cele găsite anterior, care să conțină doar numele și specializarea pentru fiecare în parte. Aceste etichete identificate au o clasă numită “result-text”.

În **Fig.4.** se poate observa sursa paginii web a site-ului unde este identificată această etichetă *div* cu clasa “result-text”.

```
<div class="doctors-box" itemscope itemtype="https://schema.org/Physician">

    <div class="image show-for-medium  " data-clause="Medicul aplică clauza de conștiință">
       <a href="/medici/greta-irinel-alexe,258,d,256" title="Greta-Irinel Alexe" rel=""><img alt

    </div>
    <div class="doctors-content">
        <div class="result-text">
            <span class="show-for-medium button-center">

                <button class="doctor-visit-button btn btn-darker-blue" onclick="openWithParamet
            </span>
            <h2 class="result-title" itemprop="name">
                <a itemprop="url" href="/medici/greta-irinel-alexe,258,d,256">
                    Dr.
                    Greta-Irinel Alexe
                </a>
            </h2>
            <div class="doctor-rating">

            </div>

                <div class="doctor-specs">
                    <span class="icon icon-menuicon-01"></span>
                    <span class="values" style="text-transform:none;">
                        Ecografie
                    </span>
                </div>

            <div itemprop="hasPOS">
                <div itemtype="Place" class="doctor-facility">
                    <span class="icon icon-menuicon-11"></span>
                    Clinica Iasi Medicover

                </div>
            </div>
```

**Fig.4.** Sursa paginii web cu etichetele div corespunzătoare numelor și specializărilor medicilor

Ulterior se elimină spațiile libere nefolositoare din textul extras ce conține datele de interes cu ajutorul unor funcții de modificare a șirurilor de caractere.

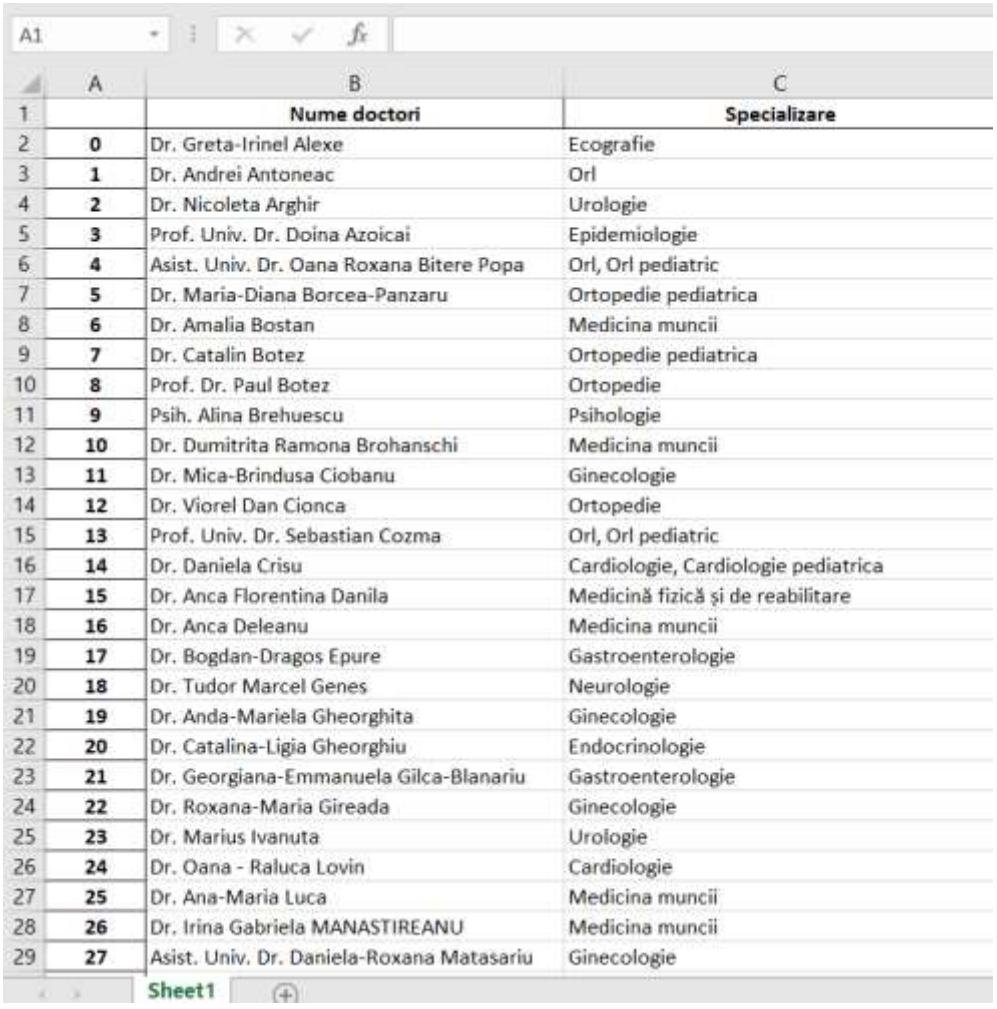
Apoi, textele găsite cu numele și specializarea medicilor se atașează variabilelor create anterior de tip listă.

Pentru a prelucra aceste date se utilizeaza biblioteca Pandas și se creează o variabilă de tip DataFrame, în care sunt introduse datele sub forma unui tabel cu două coloane. Apoi, cu ajutorul funcției *dataframe.to_excel* sunt exportate aceste date într-un fișier extern în format Excel [5].

# CAPITOLUL 5. REZULTATE ȘI CONCLUZII

Cu ajutorul funcțiilor din bibliotecile BeautifulSoup și Pandas din Python au fost identificate și exportate cu succes textele cu numele și specializările medicilor din clinica Medicover Iași.

Astfel, s-a demonstrat că această tehnică de Web Scraping este eficientă în manipularea și identificarea datelor din fișierele de tip HTML.



| | A | B | C |
|---|---|---|---|
| 1 | | **Nume doctori** | **Specializare** |
| 2 | **0** | Dr. Greta-Irinel Alexe | Ecografie |
| 3 | **1** | Dr. Andrei Antoneac | Orl |
| 4 | **2** | Dr. Nicoleta Arghir | Urologie |
| 5 | **3** | Prof. Univ. Dr. Doina Azoicai | Epidemiologie |
| 6 | **4** | Asist. Univ. Dr. Oana Roxana Bitere Popa | Orl, Orl pediatric |
| 7 | **5** | Dr. Maria-Diana Borcea-Panzaru | Ortopedie pediatrica |
| 8 | **6** | Dr. Amalia Bostan | Medicina muncii |
| 9 | **7** | Dr. Catalin Botez | Ortopedie pediatrica |
| 10 | **8** | Prof. Dr. Paul Botez | Ortopedie |
| 11 | **9** | Psih. Alina Brehuescu | Psihologie |
| 12 | **10** | Dr. Dumitrita Ramona Brohanschi | Medicina muncii |
| 13 | **11** | Dr. Mica-Brindusa Ciobanu | Ginecologie |
| 14 | **12** | Dr. Viorel Dan Cionca | Ortopedie |
| 15 | **13** | Prof. Univ. Dr. Sebastian Cozma | Orl, Orl pediatric |
| 16 | **14** | Dr. Daniela Crisu | Cardiologie, Cardiologie pediatrica |
| 17 | **15** | Dr. Anca Florentina Danila | Medicină fizică și de reabilitare |
| 18 | **16** | Dr. Anca Deleanu | Medicina muncii |
| 19 | **17** | Dr. Bogdan-Dragos Epure | Gastroenterologie |
| 20 | **18** | Dr. Tudor Marcel Genes | Neurologie |
| 21 | **19** | Dr. Anda-Mariela Gheorghita | Ginecologie |
| 22 | **20** | Dr. Catalina-Ligia Gheorghiu | Endocrinologie |
| 23 | **21** | Dr. Georgiana-Emmanuela Gilca-Blanariu | Gastroenterologie |
| 24 | **22** | Dr. Roxana-Maria Gireada | Ginecologie |
| 25 | **23** | Dr. Marius Ivanuta | Urologie |
| 26 | **24** | Dr. Oana - Raluca Lovin | Cardiologie |
| 27 | **25** | Dr. Ana-Maria Luca | Medicina muncii |
| 28 | **26** | Dr. Irina Gabriela MANASTIREANU | Medicina muncii |
| 29 | **27** | Asist. Univ. Dr. Daniela-Roxana Matasariu | Ginecologie |

**Fig.5.** Tabelul din fișierul Excel exportat cu numele și specializările medicilor

## Bibliografie

[1] “Web Scraping with Python”, 2nd Edition Ryan Mitchell

[2] “Web Scraping or Web Crawling : State of Art, Techniques, Approaches and Application” Moaiad Ahmad Khder

[3] “Web Scraping” Bo Zhao University of Washington Seattle

[4] “Data Analysis by Web Scraping using Python” David Mathew Thomas, Sandeep Mathur

[5] “A Practical Introduction to Web Scraping in Python”, realpython.com

## ANEXĂ

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

URL='https://www.medicover.ro/medici/iasi,l,'

data1=[]
data2=[]

for page in range (1,5):
    pagina=requests.get(URL + str(page) + ',s')
    soup=BeautifulSoup(pagina.content, 'html.parser')
    results=soup.find('div', class_='doctors-list')
    lista_doctori=results.find_all('div', class_="result-text")

    for result in lista_doctori:
        nume_doctor=result.find("h2", class_="result-title")
        nume_doctor_stripped=" ".join(nume_doctor.text.split())
        specializare=result.find("span", class_="values")
        specializare_stripped=specializare.text.strip()
        data1.append(nume_doctor_stripped)
        data2.append(specializare_stripped)
        print(nume_doctor_stripped)
        print(specializare_stripped)
        print()

df=pd.DataFrame()
df['Nume doctori']=data1
df['Specializare']=data2
df.to_excel('Tabel_doctori_Medicover.xlsx', index=True)
```