**MINISTERUL EDUCAŢIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Programul de studii: Tehnologia informației**

RAPORT

# LUCRARE DE LABORATOR NR. 5

# la Programarea Declarativă

**Tema: Colectarea automată a datelor**

A efectuat:

st. gr. TI-211 Popa Cătălin

A verificat: lect.dr. Mariana Rusu

UTM, Chișinău 2023

**Tema**

Colectarea automată a datelor.

**Exercițiul 1**

Alegeți un subiect ce vă interesează pe wikipedia.org și îndepliniți următoarele sarcini:

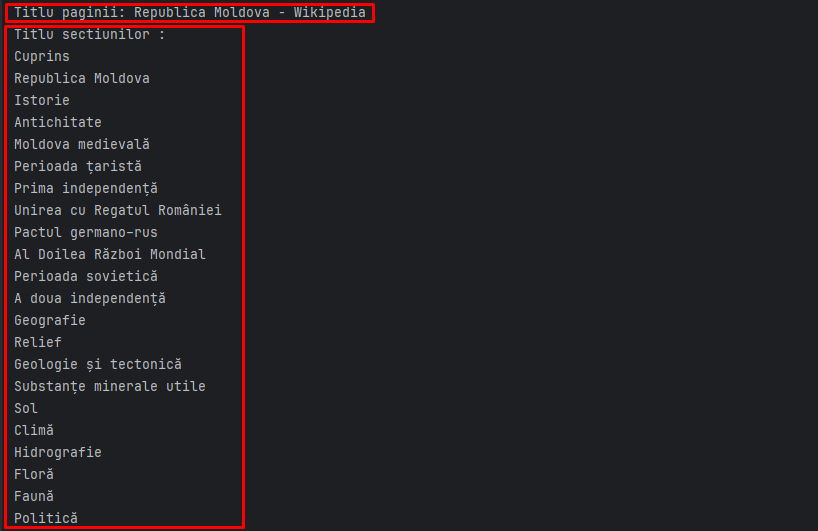
• capturați titlul paginii;

• capturați toate titlurile secțiunilor;

• obțineți minim o imagine de pe acel site

import os  
import re  
import requests  
from bs4 import BeautifulSoup  
from urllib.parse import urljoin  
url = 'https://ro.wikipedia.org/wiki/Republica\_Moldova'  
response = requests.get(url)  
  
soup = BeautifulSoup(response.content, 'html.parser')  
title = soup.title.string  
print(f"Titlu paginii: {title}")  
  
titluri\_sectiuni = soup.find\_all(re.compile('^h[1-6]'))  
print("Titlu sectiunilor : ")  
for header in titluri\_sectiuni:  
 print(header.text)  
  
desktop\_path = os.path.join(os.path.expanduser("~"), "Desktop")  
save\_path = os.path.join(desktop\_path, 'imagini')  
if not os.path.exists(save\_path):  
 os.makedirs(save\_path)  
  
# Parcurgem toate tag-urile de imagine din pagina web  
downloaded\_images = 0  
for item in soup.find\_all('img'):  
 if downloaded\_images >= 30:  
 break  
 # Obținem URL-ul imaginii  
 img\_url = urljoin(url, item['src'])  
 try:  
 # Descarcăm imaginea  
 img\_data = requests.get(img\_url).content  
 # Obținem numele fișierului din URL-ul imaginii  
 img\_name = os.path.join(save\_path, os.path.basename(img\_url))  
 # Salvam imaginea în directorul creat  
 with open(img\_name, 'wb') as img\_file:  
 img\_file.write(img\_data)  
 print(f'Imagine descărcată și salvată: {img\_name}')  
 downloaded\_images += 1  
 except Exception as e:  
 print(f'Eroare la descărcarea imaginii de la {img\_url}: {e}')

wb înseamnă "write binary" sau "scrie în mod binar"



**Figura 1 – Afișare titlu pagina și titlu secțiunilor.**



**Figura 2 – Afișarea imaginilor salvate.**

**Exercițiul 2**

Accesați site-ul web: http://books.toscrape.com/index.html care este conceput special pentru testarea web scraping. Obțineți titlul fiecărei cărți care are o evaluare de 2 stele și, la sfârșit, să aveți doar o listă Python cu toate titlurile lor.

• găsiți structura URL-ului pentru a parcurge fiecare pagină ;

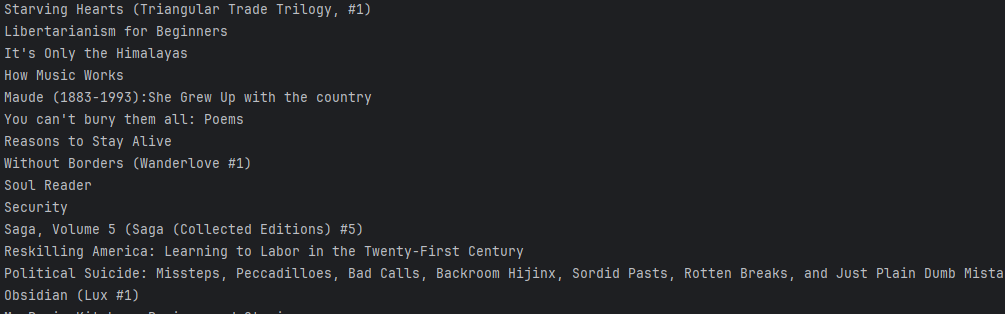
• parsați fiecare pagină din catalog;

• găsiți ce etichetă/clasă reprezintă evaluarea cu stele ;

• filtrați cu if evaluarea cu stele;

• stocați rezultatele într-o listă.

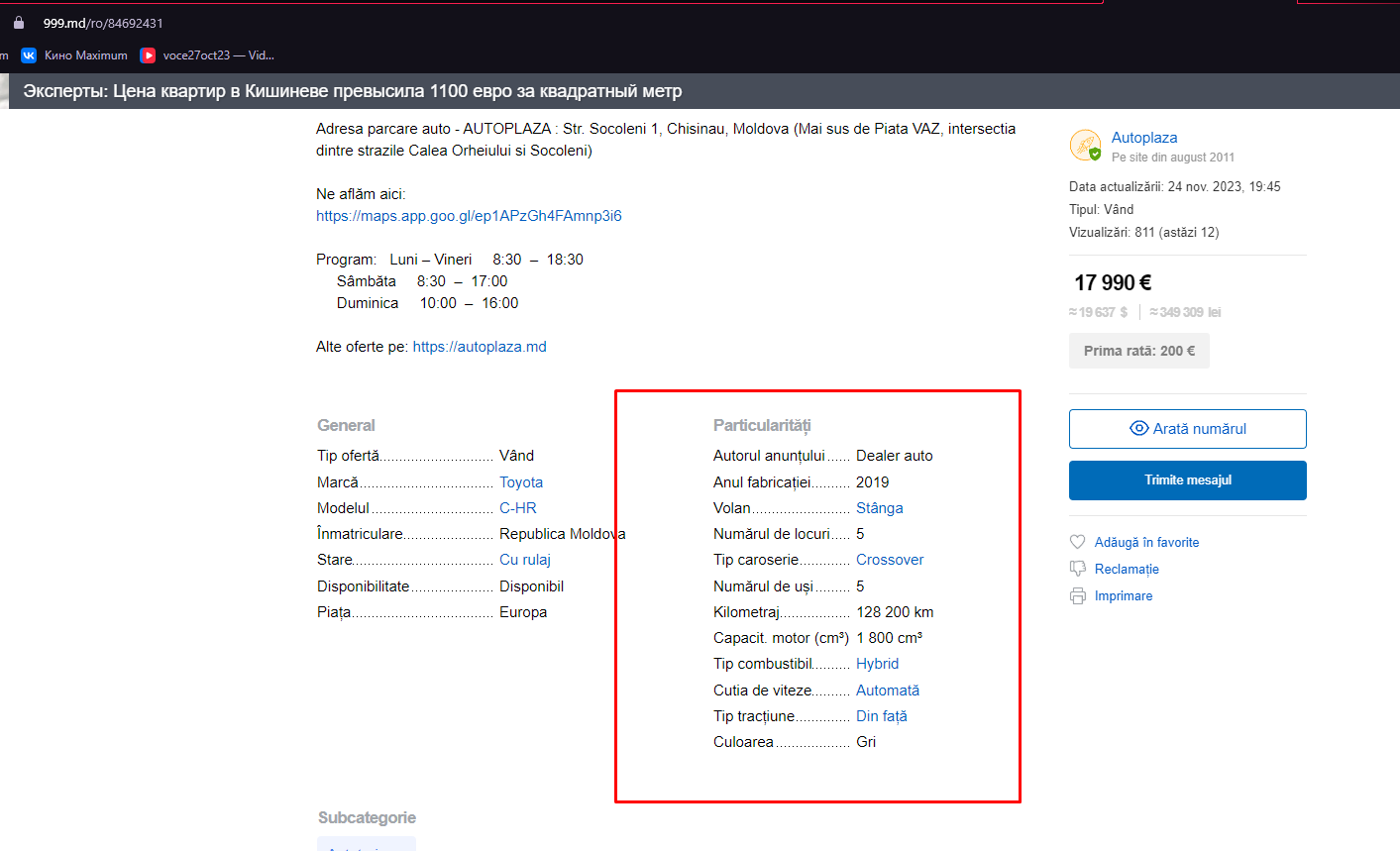
import requests  
from bs4 import BeautifulSoup  
def carti\_2\_stele(url):  
 titles = []  
 while url:  
 response = requests.get(url)  
 soup = BeautifulSoup(response.text, 'html.parser')  
  
 # Obținem titlurile cărților cu evaluare de 2 stele  
 for book in soup.find\_all('article', class\_='product\_pod'):  
 rating = book.find('p', class\_='star-rating')['class'][1]  
 if rating == 'Two':  
 title = book.h3.a['title']  
 titles.append(title)  
  
 # Găsim următoarea pagină  
 next\_page = soup.find('li', class\_='next')  
 if next\_page:  
 url = url.rsplit('/', 1)[0] + '/' + next\_page.a['href']  
 else:  
 url = None  
 return titles  
  
url = 'http://books.toscrape.com/index.html'  
result = carti\_2\_stele(url)  
for title in result:  
 print(title)



**Figura 4 – Afișare titlu cărți cu 2 stele.**

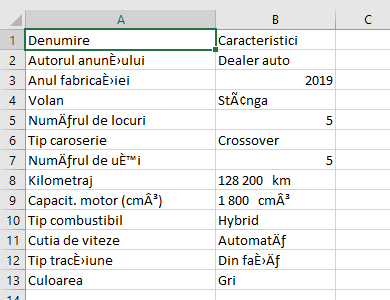
**Exercițiul 3**

Faceți cereri către minim 3 site-uri pentru a obține informația dorită (la alegere: date meteo, curs valutar, preț produse, rating hotele, restaurante etc.). Ca rezultat trebuie să aveți 3 fișiere .csv cu date similare, de ex. date meteo pe luna octombrie de la 3 site-uri de prognoză meteo, cursul valutar pentru o lună de la 3 bănci diferite, ratingul la o listă de hotele de la 3 site-uri diferite etc. Notă: asigurați-vă că site-urile permit web scraping.

**999**

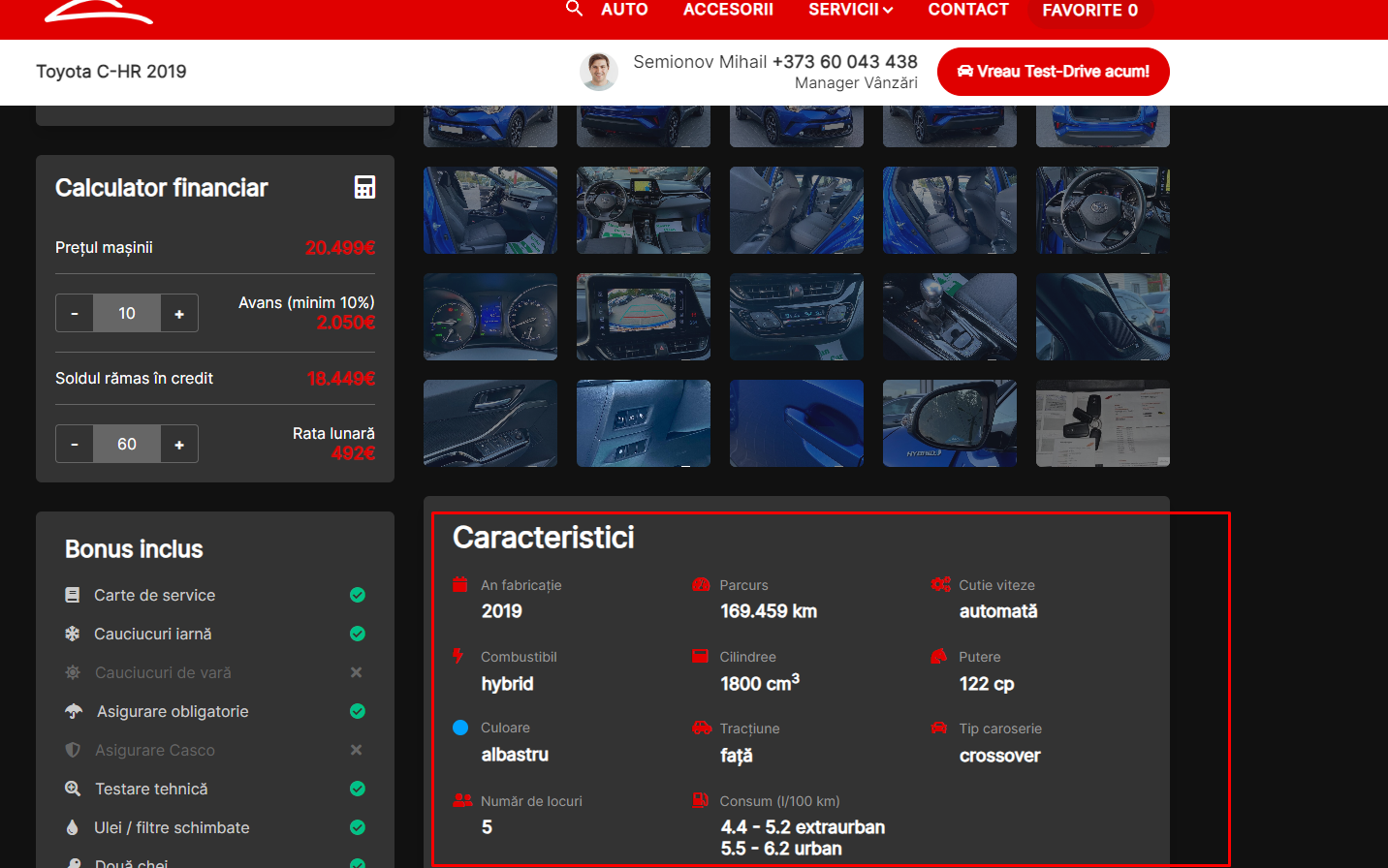
**Figura 5 – Afișare proprietăți pe site.**

import requests  
from bs4 import BeautifulSoup  
from csv import writer  
url = "https://999.md/ro/84692431"  
page = requests.get(url)  
soup = BeautifulSoup(page.content, 'html.parser')  
results = soup.find\_all('div', class\_='adPage\_\_content\_\_features\_\_col grid\_7 suffix\_1')  
  
with open('999.csv', 'w', newline='', encoding='utf8') as f:  
 thewriter = writer(f)  
 header = ['Denumire', 'Caracteristici']  
 thewriter.writerow(header)  
  
 for result in results:  
 uls = result.find\_all('ul')  
 for div in uls:  
 key\_spans = div.find\_all('span', class\_='adPage\_\_content\_\_features\_\_key')  
 value\_spans = div.find\_all('span', class\_='adPage\_\_content\_\_features\_\_value')  
  
 key\_texts = [span.text.strip() for span in key\_spans]  
 value\_texts = [span.text.strip() for span in value\_spans]  
  
 for key, value in zip(key\_texts, value\_texts):  
 jobinfo = [key, value]  
 thewriter.writerow(jobinfo)



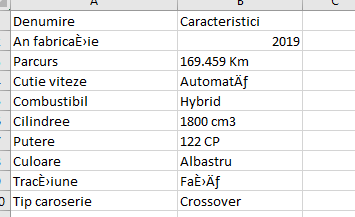
**Figura 6 – Afișare proprietăți în .csv.**

**Interauto**



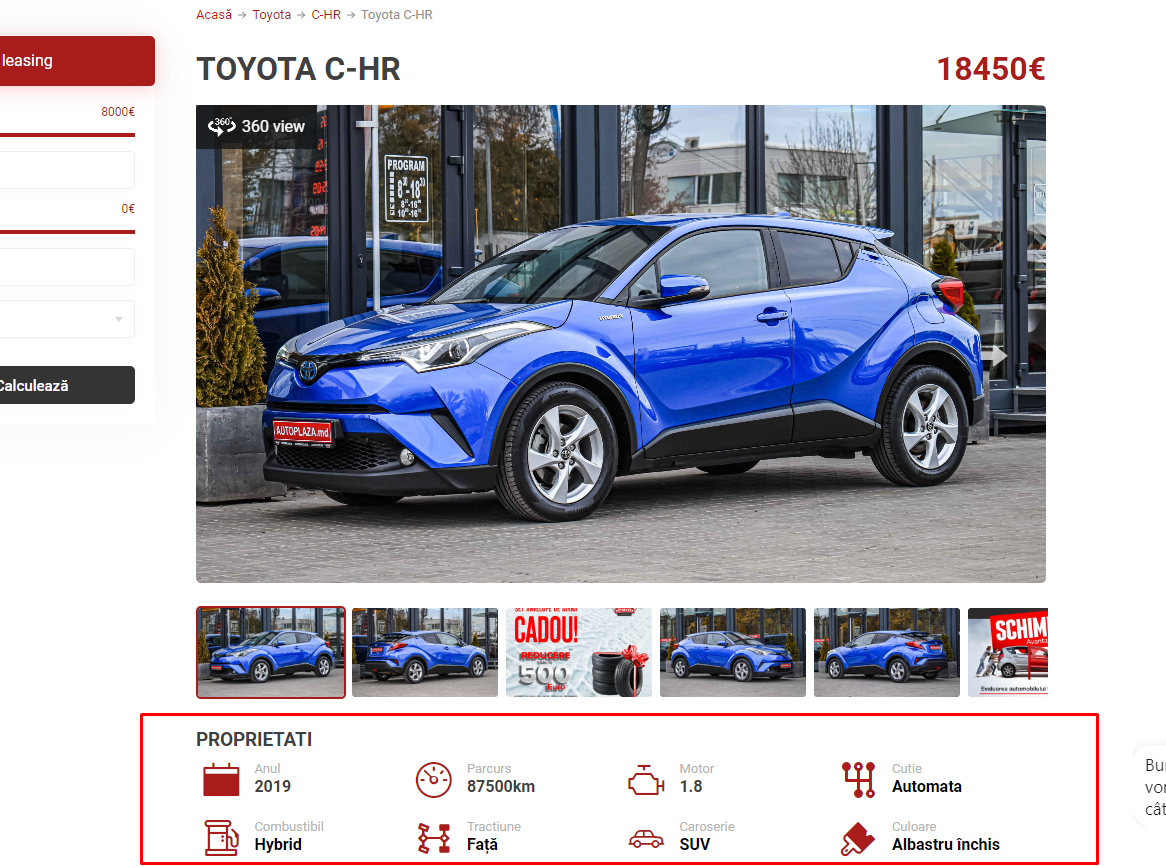
**Figura 7 – Afișare proprietăți pe site.**

import requests  
from bs4 import BeautifulSoup  
from csv import writer  
  
url = "https://interauto.md/vehicle/toyota-c-hr-a63880/"  
page = requests.get(url)  
soup = BeautifulSoup(page.content, 'html.parser')  
results = soup.find\_all('div', class\_='rounded shadow bg-dark position-relative single\_car\_\_specifications')  
#print(results)  
with open('interauto.csv', 'w', newline='', encoding='utf8') as f:  
 thewriter = writer(f)  
 header = ['Denumire', 'Caracteristici']  
 thewriter.writerow(header)  
  
 for result in results:  
 divs = result.find\_all('div', class\_='col-6 col-md-4 item')  
 for div in divs:  
 h4\_text = div.h4.text.strip()  
 h6\_text = div.h6.text.strip()  
 print(h4\_text)  
 print(h6\_text)  
 jobinfo = [h6\_text, h4\_text]  
 thewriter.writerow(jobinfo)



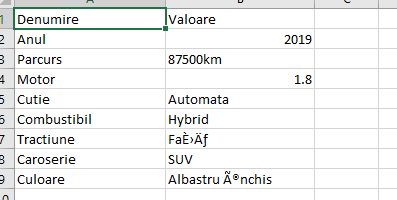
**Figura 8 – Afișare proprietăți în .csv.**

**Autoplaza**



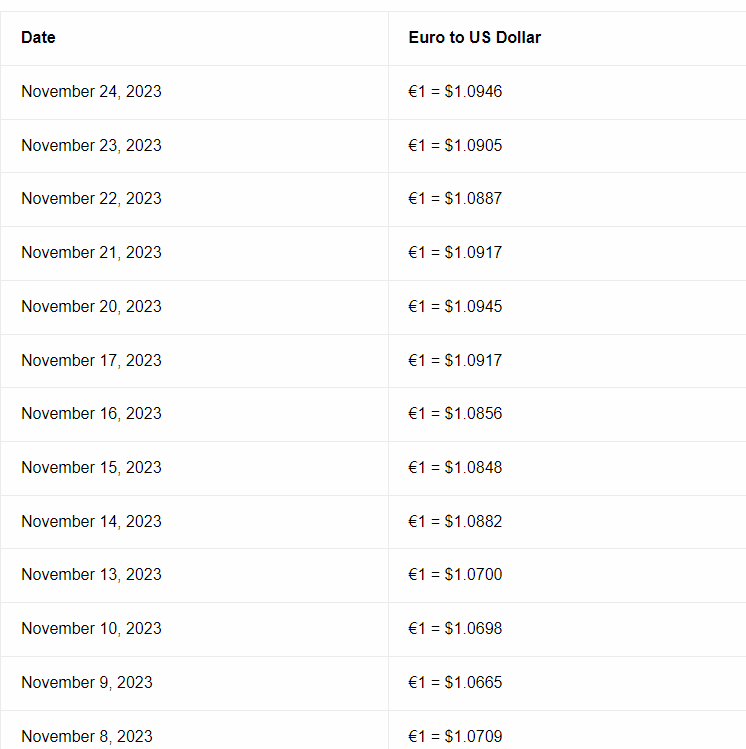
**Figura 9 – Afișare proprietăți pe site.**

import requests  
from bs4 import BeautifulSoup  
from csv import writer  
url = "https://autoplaza.md/cars/171674/"  
page = requests.get(url)  
soup = BeautifulSoup(page.content, 'html.parser')  
car\_properties\_div = soup.find('div', class\_='car-proprieties')  
print(car\_properties\_div)  
with open('autoplaza.csv', 'w', newline='', encoding='utf8') as f:  
 thewriter = writer(f)  
 header = ['Denumire', 'Valoare']  
 thewriter.writerow(header)  
  
 # Iterează prin fiecare div cu clasa "car-prop" în cadrul div-ului "car-proprieties"  
 for car\_prop\_div in car\_properties\_div.find\_all('div', class\_='car-prop'):  
 # Extrage denumirea proprietății și valoarea corespunzătoare  
 property\_name = car\_prop\_div.find('span').get\_text(strip=True)  
 property\_value = car\_prop\_div.find('strong') or car\_prop\_div.find('a')  
 property\_value = property\_value.get\_text(strip=True) if property\_value else ""  
  
 # Adaugă perechea de denumire și valoare în fișierul CSV  
 jobinfo = [property\_name, property\_value]  
 thewriter.writerow(jobinfo)



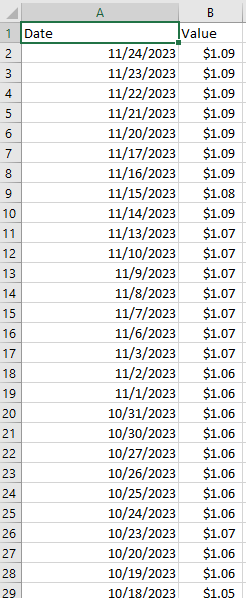
**Figura 10 – Afișare proprietăți în .csv.**

**Curs valutar 1**



**Figura 11 – Curs valutar 1.**

import requests  
from bs4 import BeautifulSoup  
import csv  
  
url = "https://www.exchange-rates.org/exchange-rate-history/eur-usd"  
page = requests.get(url)  
soup = BeautifulSoup(page.content, 'html.parser')  
table = soup.find('table', class\_='history-rates-data')  
  
if table:  
 with open('curs1.csv', 'w', newline='') as csvfile:  
 csv\_writer = csv.writer(csvfile)  
 csv\_writer.writerow(['Date', 'Value'])  
  
 rows = table.find\_all('tr')  
 for row in rows[1:]: # sarim peste header  
 columns = row.find\_all('td')  
  
 if len(columns) >= 2:  
 date = columns[0].find('a', class\_='n').text.strip()  
 value1 = columns[1].find('span', class\_='n').text.strip()  
 value = value1.split('=')[1].strip().split()[0]  
 print(f'Date: {date}, Value: {value}')  
 csv\_writer.writerow([date, value])  
else:  
 print("Table not found on the page.")



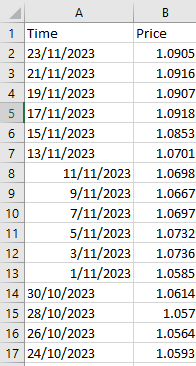
**Figura 12 – Afisarea in csv.**

**Curs valutar 2**



**Figura 13 – Curs valutar 2.**

import requests  
from bs4 import BeautifulSoup  
import csv  
from datetime import datetime  
  
url = "https://www.exchangerates.org.uk/EUR-USD-exchange-rate-history.html"  
page = requests.get(url)  
soup = BeautifulSoup(page.content, 'html.parser')  
  
table = soup.find('table')  
if table:  
 with open('curs2.csv', 'w', newline='') as csvfile:  
 csv\_writer = csv.writer(csvfile)  
 csv\_writer.writerow(['Time', 'Price'])  
  
 rows = table.find\_all('tr', class\_='colone')  
 for row in rows[1:]: # sarim peste header  
 columns = row.find\_all('td')  
  
 raw\_time = columns[0].text.strip()  
 time = datetime.strptime(raw\_time, '%A %d %B %Y').strftime('%d/%m/%Y')  
  
 raw\_price = columns[1].text.strip()  
 price = raw\_price.split('=')[1].strip().split()[0]  
 print(f'Time: {time}, Price: {price}')  
 csv\_writer.writerow([time, price])  
else:  
 print("Table not found on the page.")



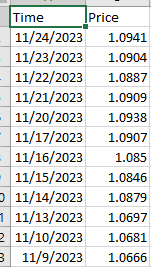
**Figura 14 – Afisarea in csv.**

**Curs valutar 3**



**Figura 15 – Curs valutar 3.**

import requests  
from bs4 import BeautifulSoup  
import csv  
  
url = "https://www.investing.com/currencies/eur-usd-historical-data"  
page = requests.get(url)  
soup = BeautifulSoup(page.content, 'html.parser')  
  
table = soup.find('table', class\_='datatable\_table\_\_DE\_1\_ datatable\_table--border\_\_XOKr2 datatable\_table--mobile-basic\_\_rzXxT datatable\_table--freeze-column\_\_XKTDf')  
if table:  
 with open('curs3.csv', 'w', newline='') as csvfile:  
 csv\_writer = csv.writer(csvfile)  
 csv\_writer.writerow(['Time', 'Price'])  
  
 rows = table.find\_all('tr')  
 for row in rows[1:]:   
 columns = row.find\_all('td')  
 time = columns[0].text.strip()  
 price = columns[1].text.strip()  
 print(f'Time: {time}, Price: {price}')  
 csv\_writer.writerow([time, price])  
else:  
 print("Table not found on the page.")



**Figura 16 – Afisarea in csv.**

**Concluzie:**

În concluzie, realizarea acestui set de sarcini a evidențiat eficiența tehnicii de web scraping în extragerea informațiilor de pe diverse site-uri web. Procesul a implicat capturarea de date de pe Wikipedia, parsarea catalogului de cărți de pe un site specializat, și obținerea informațiilor de la trei site-uri distincte referitoare la date meteorologice, curs valutar și ratinguri de hoteluri etc. Respectarea eticii și a regulilor legale a fost prioritară pentru a asigura o colectare de date responsabilă și în conformitate cu termenii și condițiile site-urilor respective. Web scraping-ul a demonstrat utilitatea în extragerea eficientă a informațiilor variate de pe internet.