In [ ]:
```
# Web Scraping Assignment 3
```

In [3]:
```
 # Let's first install the selenium library
! pip install selenium
```

```
Requirement already satisfied: selenium in c:\users\gaura\anaconda3\lib\site-packages
(4.17.2)
Requirement already satisfied: urllib3[socks]<3,>=1.26 in c:\users\gaura\anaconda3\li
b\site-packages (from selenium) (1.26.16)
Requirement already satisfied: trio~=0.17 in c:\users\gaura\anaconda3\lib\site-packag
es (from selenium) (0.24.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\gaura\anaconda3\lib\si
te-packages (from selenium) (0.11.1)
Requirement already satisfied: certifi>=2021.10.8 in c:\users\gaura\anaconda3\lib\sit
e-packages (from selenium) (2023.7.22)
Requirement already satisfied: typing_extensions>=4.9.0 in c:\users\gaura\anaconda3\l
ib\site-packages (from selenium) (4.9.0)
Requirement already satisfied: attrs>=20.1.0 in c:\users\gaura\anaconda3\lib\site-pac
kages (from trio~=0.17->selenium) (22.1.0)
Requirement already satisfied: sortedcontainers in c:\users\gaura\anaconda3\lib\site-
packages (from trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: idna in c:\users\gaura\anaconda3\lib\site-packages (fr
om trio~=0.17->selenium) (3.4)
Requirement already satisfied: outcome in c:\users\gaura\anaconda3\lib\site-packages
(from trio~=0.17->selenium) (1.3.0.post0)
Requirement already satisfied: sniffio>=1.3.0 in c:\users\gaura\anaconda3\lib\site-pa
ckages (from trio~=0.17->selenium) (1.3.0)
Requirement already satisfied: cffi>=1.14 in c:\users\gaura\anaconda3\lib\site-packag
es (from trio~=0.17->selenium) (1.15.1)
Requirement already satisfied: wsproto>=0.14 in c:\users\gaura\anaconda3\lib\site-pac
kages (from trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: PySocks!=1.5.7,<2.0,>=1.5.6 in c:\users\gaura\anaconda
3\lib\site-packages (from urllib3[socks]<3,>=1.26->selenium) (1.7.1)
Requirement already satisfied: pycparser in c:\users\gaura\anaconda3\lib\site-package
s (from cffi>=1.14->trio~=0.17->selenium) (2.21)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\gaura\anaconda3\lib\site-pac
kages (from wsproto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
```

In [2]:
```
#importing all the required libraries
import pandas as pd
import selenium
from selenium import webdriver
import time
from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementEx
```

In [ ]:
```
# let's first connect to the web driver
driver = webdriver.Chrome(r"C:\Users\Neha\Downloads\chromedriver_win32\chromedriver.ex
```

In [ ]:
```
1 Write a python program which searches all the product under a particular product fro
```

In [ ]:
```
url="https://www.amazon.in/"
driver.get(url)

time.sleep(2)
search_bar=driver.find_element_by_xpath("//input[@type='text']")
search_bar
```

```
search_bar.send_keys('shoes')

time.sleep(2)
# Locating the button and clicking it to search for shoes
button=driver.find_element_by_xpath("//input[@id='nav-search-submit-button']")
button.click()
```

In [ ]:
```
2 In the above question, now scrape the following details of each product listed in fi
```

In [ ]:
```
# creating empty list
brand=[]
name_pr=[]
rating=[]
no_rating=[]
price=[]
re_ex=[]
exp_del=[]
avail=[]
other_detail=[]


for page in range(0,3):

    brands=driver.find_elements_by_xpath("//span[@class='a-size-base-plus a-color-base
    for i in brands:
        brand.append(i.text)#appending the text in Brand list


    name_product=driver.find_elements_by_xpath('//span[@class="a-size-base-plus a-colo
    for j in name_product:
        name_pr.append(j.text)# appending the text in name_pr

    prices=driver.find_elements_by_xpath('//span[@class="a-price-whole"]')
    for k in prices:
        price.append(k.text)# appending the text in price list
```

In [ ]:
```
len(brand),len(price),len(name_pr)
```

In [ ]:
```
page_urls=[]
for page in range(0,3):
    url1=driver.find_elements_by_xpath("//a[@class='a-link-normal a-text-normal']")
    for t in url1:
        page_urls.append(t.get_attribute('href'))
    page_urls

    time.sleep(3)
  #  for scraping no_rating
    no_ratings=driver.find_elements_by_xpath('//a[@id="acrCustomerReviewLink"]')
    for l in no_ratings:
        if l.text is None:
            no_rating.append("--")
        else:
            no_rating.append(l.text)

    time.sleep(3)
  # for scraping rating
    ratings=driver.find_elements_by_xpath('//span[@data-hook="acr-average-stars-rating
```

```
    for m in ratings:
        rating.append(m.text)

    time.sleep(2)
  # for scraping return/exchange
    return_ex= driver.find_elements_by_xpath('//a[@class="a-size-small a-link-normal a
    for n in return_ex:
        re_ex.append(n.text)

        time.sleep(2)
  # for scraping expected delivery
    expec_del=driver.find_elements_by_xpath('//div[@id="ddmDeliveryMessage"]')
    for o in expec_del:
        exp_del.append(o.text)

    time.sleep(2)
  # for scraing other detail
    pr_detail=driver.find_elements_by_xpath('//hr[@aria-hidden="true"]')
    for p in pr_detail:
        other_detail.append(p.text)

    time.sleep(2)
  # for scraping availability
    pr_avail=driver.find_elements_by_xpath('//div[@id="availability"]')
    for q in pr_avail:
        avail.append(q.text)

    time.sleep(2)
  # for scraping product url
 #   product_url=driver.find_elements_by_xpath('//a[@class="a-link-normal a-text-norma
   # for r in product_url:
    #    pr_url.append(r.text)
```

In [ ]:
```
len(no_rating)
```

In [ ]:
```
#combining all lists in to a single dataframe
df_shoes=pd.DataFrame({'Brand_name':brand,
                       'Product_name':name_pr,
                       'Ratings':rating,
                       'No_ratings':no_rating,
                       'Price':price,
                       'Return/Exchange':re_ex,
                       'Expected_del':exp_del,
                       'Availability':avail,
                       'Other_detail':other_detail,
                       'Product_URL':pr_url})
df_shoes
```

In [ ]:
```
3. Write a python program to access the search bar and search button on images.google.
images each for keywords ‘fruits’, ‘cars’ and ‘Machine Learning’, ‘Guitar’, ‘Cakes’.
```

In [ ]:
```
4. Write a python program to search for a smartphone(e.g.: Oneplus Nord, pixel 4A, etc
and scrape following details for all the search results displayed on 1st page. Details
Name”, “Smartphone name”, “Colour”, “RAM”, “Storage(ROM)”, “Primary Camera”,
“Secondary Camera”, “Display Size”, “Battery Capacity”, “Price”, “Product URL”. Incase
details is missing then replace it by “- “. Save your results in a dataframe and CSV.
```

In [ ]:
```
# let's first connect to the web driver
driver = webdriver.Chrome(r"C:\Users\Neha\Downloads\chromedriver_win32\chromedriver.ex
```

In [ ]:
```
url = "https://www.flipkart.com"
driver.get(url)
```

In [ ]:
```
# finding element for job search bar
search_ph = driver.find_element_by_xpath("//input[@type='text']")
search_ph
```

In [ ]:
```
# write on search bar
search_ph.send_keys("Samsung Galaxy M31")
```

In [ ]:
```
# clicking the search button
search_button=driver.find_element_by_xpath("//button[@class='L0Z3Pu']")
search_button.click()
```

In [ ]:
```
# creating empty lists for scraping data
b_name=[]
ph_name=[]
color=[]
ram=[]
s_rom=[]
=[]
```

In [ ]:
```
5. Write a program to scrap geospatial coordinates (latitude, longitude) of a city sea
```

In [ ]:
```
import requests
from bs4 import BeautifulSoup

def get_coordinates(city):
  # Format the city name for the URL
  formatted_city = city.replace(" ", "+")
```

In [ ]:
```
# Send a GET request to Google Maps with the formatted city name
  url = f"https://www.google.com/maps/search/{formatted_city}"
  response = requests.get(url)
```

In [ ]:
```
# Parse the HTML response using BeautifulSoup
  soup = BeautifulSoup(response.text, "html.parser")
```

In [ ]:
```
 # Find the element containing the coordinates
  coordinates_element = soup.find("meta", itemprop="image")
```

In [ ]:
```
# Extract the coordinates from the element's content attribute
  coordinates = coordinates_element["content"].split(";")[1].strip().split(",")
```

In [ ]:
```
# Return the latitude and longitude as a tuple
  return float(coordinates[0]), float(coordinates[1])
```

In [ ]:
```
# Example usage
city = input("Enter a city name: ")
latitude, longitude = get_coordinates(city)
print(f"The coordinates of {city} are: Latitude: {latitude}, Longitude: {longitude}")
```

In [ ]:
```
6. Write a program to scrap all the available details of best gaming laptops from digi
```

In [ ]:
```
from selenium import webdriver
import time
```

In [ ]:
```
# Set up the WebDriver
driver = webdriver.Chrome('path_to_chromedriver')
```

In [ ]:
```
# Open the website
driver.get('https://www.digit.in/')
```

In [ ]:
```
# Search for gaming laptops
search_bar = driver.find_element_by_id('searchDiv')
search_bar.send_keys('gaming laptops')
search_bar.submit()
```

In [ ]:
```
# Wait for the search results to load
time.sleep(2)
```

In [ ]:
```
# Scrape the details
laptop_elements = driver.find_elements_by_class_name('searchPage')
laptop_details = []
```

In [ ]:
```
for laptop in laptop_elements:
  name = laptop.find_element_by_class_name('searchProductTitle').text
  price = laptop.find_element_by_class_name('searchPrice').text
  specifications = laptop.find_element_by_class_name('searchSpec').text

  laptop_details.append({
  'Name': name,
  'Price': price,
  'Specifications': specifications
  })
```

In [ ]:
```
# Print the scraped details
for laptop in laptop_details:
  print(laptop)
```

In [ ]:
```
# Close the WebDriver
driver.quit()
```

In [ ]:
```
7. Write a python program to scrape the details for all billionaires from www.forbes.c
“Rank”, “Name”, “Net worth”, “Age”, “Citizenship”, “Source”, “Industry”.
```

In [ ]:
```
import requests
from bs4 import BeautifulSoup
```

In [ ]:
```
# Send a GET request to the Forbes website
url = "https://www.forbes.com/billionaires/"
response = requests.get(url)
```

In [ ]:
```
# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.content, "html.parser")
```

In [ ]:
```
# Find the table containing the billionaire details
table = soup.find("table", class_="table")
```

In [ ]:
```
# Find all the rows in the table
rows = table.find_all("tr")
```

In [ ]:
```
# Iterate over each row and extract the required details
for row in rows:
  # Find all the columns in the row
  columns = row.find_all("td")
```

In [ ]:
```
# Extract the required details from the columns
  rank = columns[0].text.strip()
  name = columns[1].text.strip()
  net_worth = columns[2].text.strip()
  age = columns[3].text.strip()
  citizenship = columns[4].text.strip()
  source = columns[5].text.strip()
  industry = columns[6].text.strip()
```

In [ ]:
```
# Print the extracted details
  print("Rank:", rank)
  print("Name:", name)
  print("Net Worth:", net_worth)
  print("Age:", age)
  print("Citizenship:", citizenship)
  print("Source:", source)
  print("Industry:", industry)
  print()
```

In [ ]:
```
8. Write a program to extract at least 500 Comments, Comment upvote and time when comm
from any YouTube Video.
```

In [ ]:
```
# Set up the WebDriver
driver = webdriver.Chrome('path_to_chromedriver')  # Replace with the path to your Web
```

In [ ]:
```
# Open the YouTube video
video_url = 'https://www.youtube.com/watch?v=your_video_id'  # Replace with the URL of
driver.get(video_url)
```

In [ ]:
```
# Scroll to load comments
scroll_pause_time = 2  # Adjust the pause time as needed
scrolls = 10  # Adjust the number of scrolls as needed
```

In [ ]:
```
for _ in range(scrolls):
  driver.execute_script("window.scrollTo(0, document.documentElement.scrollHeight);")
  time.sleep(scroll_pause_time)
```

In [ ]:
```
# Extract comments, upvotes, and time
comments = driver.find_elements_by_xpath('//yt-formatted-string[@id="content-text"]')
upvotes = driver.find_elements_by_xpath('//span[@id="vote-count-middle"]')
times = driver.find_elements_by_xpath('//a[@class="yt-simple-endpoint style-scope yt-f
```

In [ ]:
```
# Store the extracted data
extracted_data = []
```

```
for comment, upvote, time in zip(comments, upvotes, times):
  extracted_data.append({
  'comment': comment.text,
  'upvote': upvote.text,
  'time': time.text
  })
```

In [ ]:
```
# Close the WebDriver
driver.quit()
```

In [ ]:
```
# Print the extracted data
for data in extracted_data:
  print(data)
```

In [ ]:
```
9. Write a python program to scrape a data for all available Hostels from https://www.
“London” location. You have to scrape hostel name, distance from city centre, ratings,
reviews, privates from price, dorms from price, facilities and property description.
```

In [5]:
```
import requests
from bs4 import BeautifulSoup
```

In [ ]:
```
# Send a GET request to the website
url = "https://www.hostelworld.com/hostels/London"
response = requests.get(url)
```

In [ ]:
```
# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.content, "html.parser")
```

In [ ]:
```
# Find all the hostel containers
hostels = soup.find_all("div", class_="fabresult")
```

In [ ]:
```
# Iterate over each hostel container and extract the required information
for hostel in hostels:
  # Extract hostel name
  name = hostel.find("h2", class_="fabresult-title").text.strip()
```

In [ ]:
```
# Extract distance from city centre
  distance = hostel.find("span", class_="distance").text.strip()
```

In [ ]:
```
# Extract ratings
  ratings = hostel.find("div", class_="rating").text.strip()
```

In [ ]:
```
# Extract total reviews
  total_reviews = hostel.find("div", class_="reviews").text.strip()
```

In [ ]:
```
# Extract overall reviews
  overall_reviews = hostel.find("div", class_="overall").text.strip()
```

In [ ]:
```
# Extract privates from price
  privates_price = hostel.find("div", class_="price-col").find("div", class_="price").
```

In [ ]:
```
# Extract dorms from price
  dorms_price = hostel.find("div", class_="price-col").find("div", class_="price").fir
```

In [ ]:
```
# Extract facilities
facilities = hostel.find("div", class_="facilities").text.strip()
```

In [ ]:
```
# Extract property description
description = hostel.find("div", class_="description").text.strip()
```

In [ ]:
```
# Print the extracted information
print("Hostel Name:", name)
print("Distance from City Centre:", distance)
print("Ratings:", ratings)
print("Total Reviews:", total_reviews)
print("Overall Reviews:", overall_reviews)
print("Privates from Price:", privates_price)
print("Dorms from Price:", dorms_price)
print("Facilities:", facilities)
print("Property Description:", description)
print()
```