Web Scraping Assignment 3

In [1]:

```
# importing libraries
import selenium
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.common.exceptions import NoSuchElementException, StaleElementReferenceException
from selenium.webdriver.support import expected_conditions as EC

import pandas as pd
import time
import re
import warnings
warnings.filterwarnings("ignore")
```

Q1.

Write a python program which searches all the product under a particular product from www.amazon.in (http://www.amazon.in). The product to be searched will be taken as input from user. For e.g. If user input is 'guitar'. Then search for guitars.

In [12]:

```
#loading url
driver=webdriver.Chrome(r"chromedriver.exe")
driver.get("https://www.amazon.in/")
WebDriverWait(driver,10).until(EC.presence_of_element_located((By.XPATH,'//li[@class="nav_last "]')))

#search required item through user input
search_box=driver.find_element(By.XPATH,'//input[@id="twotabsearchtextbox"]')
search_box.send_keys(input("Enter searching items : "))

#click on search button
search_button=driver.find_element(By.XPATH,'//div[@class="nav-search-submit nav-sprite"]')
search_button.click()
```

```
Enter searching items : laptop
```

2. In the above question, now scrape the following details of each product listed in first 3 pages of your search results and save it in a data frame and csv. In case if any product has less than 3 pages in search results then scrape all the products available under that product name. Details to be scraped are: "Brand Name", "Name of the Product", "Price", "Return/Exchange", "Expected Delivery", "Availability" and "Product URL". In case, if any of the details are missing for any of the product then replace it by "-".

In [13]:

```
#creating empty list

product_url=[]

#initialization
start=0
end=3
for i in range(start,end):
    #findng urls
    urls=driver.find_elements(By.XPATH,'//a[@class="a-link-normal s-underline-text s-underline-link-t
    time.sleep(3)
    try:
        for d in urls:
            product_url.append(d.get_attribute('href'))
    except NoSuchElementException:
        product_url.append("-")

    # next button
    next_button=driver.find_element(By.XPATH,'//a[@class="s-pagination-item s-pagination-next s-pagi
    next_button.click()
    time.sleep(10)

```

In [14]:

```
product_url
```

Out[14]:

```
['https://www.amazon.in/sspa/click?ie=UTF8&spc=MTo3NjI5MjcyMzAxODIyMjI1OjE2ODcyNDc3Mzk6c
3BfYXRmOjIwMTA4MDE1ODEzMDk4OjowOjo&url=%2FLenovo-IdeaPad-35-56cm-Warranty-82H701DNIN%2Fd
p%2FB0B4JN3WYP%2Fref%3Dsr_1_1_sspa%3Fkeywords%3Dlaptop%26qid%3D1687247739%26sr%3D8-1-spo
ns%26sp_csd%3Dd2lkZ2V0TmFtZT1zcF9hdGY%26psc%3D1',
 'https://www.amazon.in/sspa/click?ie=UTF8&spc=MTo3NjI5MjcyMzAxODIyMjI1OjE2ODcyNDc3Mzk6c
3BfYXRmOjIwMTEwMjQ4MjcwNjk4OjowOjo&url=%2FLenovo-IdeaPad-Warranty-Platinum-81X800LGIN%2F
dp%2FB0B2RBP83P%2Fref%3Dsr_1_2_sspa%3Fkeywords%3Dlaptop%26qid%3D1687247739%26sr%3D8-2-sp
ons%26sp_csd%3Dd2lkZ2V0TmFtZT1zcF9hdGY%26psc%3D1',
 'https://www.amazon.in/HP-Celeron-Graphics-Speakers-15s-fq3071TU/dp/B0C3CLH1PV/ref=sr_1
_3?keywords=laptop&qid=1687247739&sr=8-3',
 'https://www.amazon.in/Acer-Extensa-Processor-Graphics-EX215-33/dp/B0BXL9KVCL/ref=sr_1_
4?keywords=laptop&qid=1687247739&sr=8-4',
 'https://www.amazon.in/Dell-Inspiron-Windows-i3-1115G4-39-62Cms/dp/B0B468SB8G/ref=sr_1_
5?keywords=laptop&qid=1687247739&sr=8-5',
 'https://www.amazon.in/HP-Micro-Edge-Anti-Glare-15s-fq5111TU/dp/B0B6F5V23N/ref=sr_1_6?k
eywords=laptop&qid=1687247739&sr=8-6',
 'https://www.amazon.in/Dell-Vostro-i5-1135G7-35-54Cms-1-48Kgs/dp/B0C15CDKTN/ref=sr_1_7?
keywords=laptop&qid=1687247739&sr=8-7',
```

In [15]:

```
#creating empty list
product_name=[]
brand_name=[]
price=[]
return_exchange=[]
expected_delivery=[]
availability=[]

for url in product_url:
    driver.get(url)
    WebDriverWait(driver,20).until(EC.presence_of_element_located((By.XPATH,'//li[@class="nav_last "]

    # product name
    try:
        product_title=driver.find_element(By.XPATH,'//span[@id="productTitle"]')
        product_name.append(product_title.text)
    except NoSuchElementException:
        product_name.append("-")
    time.sleep(1)

    #brand name
    try:
        brand=driver.find_element(By.XPATH,'//td[@class="a-span9"]')
        brand_name.append(brand.text)
    except NoSuchElementException:
        brand_name.append("-")
    time.sleep(1)

    # Price
    try:
        prc=driver.find_element(By.XPATH,'//span[@class="a-price aok-align-center reinventPricePrice
        price.append(prc.text)
    except NoSuchElementException:
        price.append("-")
    time.sleep(1)

    # Return Exchange
    try:
        rtrn=driver.find_elements(By.XPATH,'//div[@class="a-section a-spacing-none icon-content"]')[1
        return_exchange.append(rtrn.text)
    except NoSuchElementException:
        return_exchange.append("-")
    time.sleep(1)

    # Expected Delivery
    try:
        exptd_delv=driver.find_element(By.XPATH,'/html/body/div[2]/div[2]/div[5]/div[3]/div[1]/div[3]
        expected_delivery.append(exptd_delv.text)
    except NoSuchElementException:
        expected_delivery.append("-")
    time.sleep(1)

    # availability
    try:
        avail=driver.find_element(By.XPATH,'//div[@id="availability"]')
        availability.append(avail.text)
    except NoSuchElementException:
        availability.append("-")
    time.sleep(1)

print("product name = ",len(product_name))
print("Brand name = ",len(brand_name))
print("price = ",len(price))
print("return_exchange = ",len(return_exchange))
print("expected_delivery = ",len(expected_delivery))
print("availability = ",len(availability))
```

```
print("product url = ",len(product_url))
```

```
product name =  66
Brar
price =  66
return_exchange =  66
expected_delivery =  66
availability =  66
product url =  66
```

In [16]:

```
#making dataframe
df=pd.DataFrame({"Product Name": product_name,
                 "Brand Name" : brand_name,
                 "Price" : price,
                 "Return / Exchange" : return_exchange,
                 "Expected Delivery" : expected_delivery,
                 "Availability" : availability,
                 "Product URL" : product_url})
df
```

Out[16]:

	Product Name	Brand Name	Price	Return / Exchange	Expected Delivery	Availability	Product URL
0	Lenovo IdeaPad Slim 3 Intel Core i3 11th Gen 1...	Lenovo	₹36,990	7 days Replacement	Thursday, 22 June	In stock	https://www.amazon.in/sspa/click?ie=UTF8&spc=M...
1	Lenovo IdeaPad 3 11th Gen Intel Core i3 15.6" ...	Lenovo	₹37,300	7 days Replacement	-		https://www.amazon.in/sspa/click?ie=UTF8&spc=M...
2	HP Laptop 15s, Intel Celeron N4500, 15.6 inch(...	HP	₹29,990	Pay on Delivery	Thursday, 22 June	In stock	https://www.amazon.in/HP-Celeron-Graphics-Spea...
3	Acer Extensa 15 Laptop Intel Core i3 N305 8 co...	Acer	₹31,990	7 days Replacement	Thursday, 22 June	In stock	https://www.amazon.in/Acer-Extensa-Processor-G...
4	Dell Inspiron 3511 Laptop, Intel i3-1115G4, 8G...	Dell	₹38,490	7 days Replacement	Thursday, 22 June	In stock	https://www.amazon.in/Dell-Inspiron-Windows-i3...
...	...	...	...	...	...	...	...
61	ASUS Vivobook 15 (2023), Intel Core i3-1315U 1...	ASUS	₹45,990	7 days Replacement	Thursday, 22 June	In stock	https://www.amazon.in/ASUS-Vivobook-i3-1315U-F...
62	Acer Aspire 3 Thin and Light Laptop Intel Core...	Acer	₹48,990	7 days Replacement	Thursday, 22 June	In stock	https://www.amazon.in/Acer-Generation-Windows-...
63	Lenovo IdeaPad Slim 5 Intel Core i5 11th Gen 1...	Lenovo	₹60,990	7 days Replacement	Thursday, 22 June	In stock	https://www.amazon.in/Lenovo-IdeaPad-39-62cm-G...
64	Apple 2023 MacBook Pro Laptop M2 Pro chip with...	Apple	₹1,99,900	7 days Replacement	-	Available to ship in 1-2 days	https://www.amazon.in/sspa/click?ie=UTF8&spc=M...
65	Apple 2023 MacBook Pro Laptop M2 Pro chip with...	Apple	₹1,99,900	7 days Replacement	-	Available to ship in 1-2 days	https://www.amazon.in/sspa/click?ie=UTF8&spc=M...

66 rows × 7 columns

In [17]:

```
driver.quit()
```

Q3.Write a python program to access the search bar and search button on images.google.com and scrape 10 images each for keywords ‘fruits’, ‘cars’ and ‘Machine Learning’, ‘Guitar’, ‘Cakes’.

In [18]:

```
#Loading url
driver=webdriver.Chrome(r'chromedriver.exe')
url="https://images.google.com/"
driver.get(url)
```

In [19]:

```
# hare mannually we need to press "No thanks button" .
#I have tried several times to click on that button click button. every times it raise an error
try:
    no_thanks_button=driver.find_element(By.XPATH,'//div[@class="QlyBfb"]/button')
    no_thanks_button.click()
except NoSuchElementException as e:
    print("Exception raised")
```

```
Exception raised
```

In [20]:

```
#fruit list to be searched
search_list=["fruits",'cars','Machine Learning','guiter','Cakes']

#empty lists
fruit=[]
cars=[]
machine_learning=[]
guiter=[]
cakes=[]

#nested empty list
final_list=[fruit,cars,machine_learning,guiter,cakes]


x=0    #initialisation


for i in search_list:
    driver.get(url)
    time.sleep(5)

    #search box
    search_text=driver.find_element(By.XPATH,'//textarea[@class="gLFyf"]')
    search_text.send_keys(i)

    #click on search button
    search_button=driver.find_element(By.XPATH,'//div[@class="zgAlFc"]')
    search_button.click()
    time.sleep(5)

    #finding for elements
    items=driver.find_elements(By.XPATH,'//img[@class="rg_i Q4LuWd"]')[0:10]

    #exception Handeling
    try:
        for a in items:
            final_list[x].append(a.get_attribute("src"))
    except NoSuchElementException as e:
        print(e)

    #increment of x
    x=x+1
```

In [21]:

```
df=pd.DataFrame({"Fruit":final_list[0],
                 "Cars":final_list[1],
                 "Machine Learning":final_list[2],
                 "Guiter":final_list[3],
                 "Cakes":final_list[4]})
df
```

Out[21]:

	Fruit	Cars	
0	...	...	data
1	...	...	data
2	...	...	data
3	...	...	data
4	...	...	data
5	...	...	data
6	...	...	data
7	...	...	data
8	...	...	data:ima
9	...	...	data

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

In [22]:

```
#Loading the flipkart page
url="https://www.flipkart.com/"
driver.get(url)
time.sleep(5)

#cross_button
cross_button=driver.find_element(By.XPATH,'/html/body/div[2]/div/div/button')
cross_button.click()
```

In [23]:

```
# search box
search_box=driver.find_element(By.XPATH,'//input[@class="_3704LK"]')
search_box.send_keys("smart mobiles")

#Search button
search_button=driver.find_element(By.XPATH,'//button[@class="L0Z3Pu"]')
search_button.click()
```

In [24]:

```
product_url=[]
urls=driver.find_elements(By.XPATH,'//a[@class="_1fQZEK"]')
for i in urls:
    product_url.append(i.get_attribute("href"))
len(product_url)
```

Out[24]:

24

In [25]:

```
#creating empty list
Brand_Name=[]
Smartphone_name=[]
Colour=[]
RAM=[]
ROM=[]
Primary_Camera=[]
Secondary_Camera=[]
Display_Size=[]
Battery_Capacity=[]
Price=[]


for url in product_url:
    # getting url one by one
    driver.get(url)
    time.sleep(8)


    # click on read more option
    read_more=driver.find_element(By.XPATH,'//button[@class="_2KpZ6l _1FH0tX"]')
    read_more.click()

    #brand name
    try:
        brand=driver.find_element(By.XPATH,'//span[@class="B_NuCI"]')
        Brand_Name.append(brand.text.split(" ")[0])

    except NoSuchElementException as e:
        Brand_Name.append("-")


    # SmartPhone Name
    try:
        model_name=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]/d
        Smartphone_name.append(model_name.text)

    except NoSuchElementException as e:
        Smartphone_name.append("-")

    # Color
    try:
        color=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]/div[5]
        Colour.append(color.text)

    except NoSuchElementException as e:
        Colour.append("-")

    # ram
    try:
        ram=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]/div[5]/d
        RAM.append(ram.text)

    except NoSuchElementException as e:
        RAM.append("-")

    # ROM
    try:
        rom=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]/div[5]/d
        ROM.append(rom.text)

    except NoSuchElementException as e:
        ROM.append("-")

    # Primary Camera
    try:
        primary_cam=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]/
        Primary_Camera.append(primary_cam.text)
```

```
    except NoSuchElementException as e:
        Primary_Camera.append("-")

    # Secondary_Camera
    try:
        secondary_cam=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9
        Secondary_Camera.append(secondary_cam.text)

    except NoSuchElementException as e:
        Secondary_Camera.append("-")

    #display Size
    try:
        display_size=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]
        Display_Size.append(display_size.text)

    except NoSuchElementException as e:
        Display_Size.append("-")

    # Battery Capacity
    try:
        battery=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[9]/div[
        Battery_Capacity.append(battery.text)

    except NoSuchElementException as e:
        Battery_Capacity.append("-")

    # Price
    try:
        price=driver.find_element(By.XPATH,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[2]/div/di
        Price.append(price.text)

    except NoSuchElementException as e:
        Price.append("-")


print("brand name = ", len(Brand_Name))
print("Smartphone name = ",len(Smartphone_name))
print("Colour = ",len(Colour))
print("RAM = ",len(RAM))
print("ROM = ",len(ROM))
print("Primary Camera = ",len(Primary_Camera))
print("Secondary Camera = ",len(Secondary_Camera))
print('Display Size = ',len(Display_Size))
print("Battery capacity = ",len(Battery_Capacity))
print("price = ",len(Price))
```

```
brand name =  24
Smartphone name =  24
Colour =  24
RAM =  24
ROM =  24
Primary Camera =  24
Secondary Camera =  24
Display Size =  24
Battery capacity =  24
price =  24
```

In [26]:

```
#DataFrame Making
mobile_df=pd.DataFrame({"Brand Name":Brand_Name,
                        "Smartphone_name":Smartphone_name,
                        "Colour":Colour,
                        "RAM":RAM,
                        "ROM" : ROM,
                        "Primary_Camera" : Primary_Camera ,
                        "Secondary_Camera" : Secondary_Camera ,
                        "Display_Size" : Display_Size ,
                        "Battery_Capacity" : Battery_Capacity ,
                        "Price" :Price ,
                        "Product URL":product_url
                       })


mobile_df
```

Out[26]:

	Brand Name	Smartphone_name	Colour	RAM	ROM	Primary_Camera	Secondary_Camera	Display_Size	Bat
0	Infinix	Smart 7 HD	Green Apple	2 GB	64 GB	8MP + AI Lens	5MP Front Camera: Camera Features:AI Portrait,...	16.76 cm (6.6 inch)	
1	Infinix	SMART 7	Night Black	4 GB	64 GB	13MP + AI Lens	5MP Front Camera: Camera Features: AI Portrait...	16.76 cm (6.6 inch)	
2	Infinix	SMART 7	Azure Blue	4 GB	64 GB	13MP + AI Lens	5MP Front Camera: Camera Features: AI Portrait...	16.76 cm (6.6 inch)	
3	SAMSUNG	Galaxy F13	Waterfall Blue	4 GB	128 GB	50MP + 5MP + 2MP	8MP Front Camera	16.76 cm (6.6 inch)	
4	Infinix	HOT 30i	Diamond White	8 GB	128 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
5	Infinix	HOT 30i	Mirror Black	8 GB	128 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
6	Infinix	HOT 30i	Glacier Blue	8 GB	128 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
7	Infinix	HOT 30i	Marigold	8 GB	128 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
8	Infinix	HOT 30i	Diamond White	4 GB	64 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
9	Infinix	HOT 30i	Glacier Blue	4 GB	64 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
10	Infinix	HOT 30i	Mirror Black	4 GB	64 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
11	Infinix	SMART 7	Emerald Green	4 GB	64 GB	13MP + AI Lens	5MP Front Camera: Camera Features: AI Portrait...	16.76 cm (6.6 inch)	
12	Infinix	HOT 30i	Marigold	4 GB	64 GB	50MP + AI Lens	5MP AI Camera: Camera Features: Portrait, Face...	16.76 cm (6.6 inch)	
13	SAMSUNG	Galaxy F04	Opal Green	4 GB	64 GB	13MP + 2MP	5MP Front Camera	16.51 cm (6.5 inch)	
14	POCO	C51	Royal Blue	4 GB	64 GB	8MP Dual Rear Camera	5MP Front Camera	16.56 cm (6.52 inch)	
15	MOTOROLA	G32	Mineral Gray	8 GB	128 GB	50MP + 8MP + 2MP	16MP Camera Setup: (f/2.2 Aperture, 1.0um Pixe...	16.64 cm (6.55 inch)	
16	POCO	C51	Power Black	4 GB	64 GB	8MP Dual Rear Camera	5MP Front Camera	16.56 cm (6.52 inch)	
17	POCO	C50	Country Green	2 GB	32 GB	8MP Dual Camera	5 MP Camera Setup	16.56 cm (6.52 inch)	
18	POCO	C55	Cool Blue	4 GB	64 GB	50MP Dual Rear Camera	5MP Front Camera: Camera Features: Portrait Mo...	17.04 cm (6.71 inch)	
19	REDMI	-	-	-	-	-	-	-	
20	REDMI	A1+	Light Green	3 GB	32 GB	8MP Rear Camera	5MP Camera Setup	16.56 cm (6.52 inch)	
21	REDMI	A1+	Light Blue	3 GB	32 GB	8MP Rear Camera	5MP Camera Setup	16.56 cm (6.52 inch)	

	Brand Name	Smartphone_name	Colour	RAM	ROM	Primary_Camera	Secondary_Camera	Display_Size	Bat
22	SAMSUNG	Galaxy F13	Nightsky Green	4 GB	64 GB	50MP + 5MP + 2MP	8MP Front Camera	16.76 cm (6.6 inch)	
23	POCO	C55	Power Black	4 GB	64 GB	50MP Dual Rear Camera	5MP Front Camera: Camera Features: Portrait Mo...	17.04 cm (6.71 inch)	

In [27]:

```
mobile_df.to_csv("smart_mobile_flipkart.csv",sep=';', index=False,header=True)
```

In [28]:

```
driver.quit()
```

Q5.

Write a program to scrap geospatial coordinates (latitude, longitude) of a city searched on google maps.

In [29]:

```
#Loadong the chrome driver
driver=webdriver.Chrome(r"chromedriver.exe")

#Loading the url page
url="https://maps.google.com/"
driver.get(url)
```

In [30]:

```
# city name
city= driver.find_element(By.XPATH,'/html/body/div[3]/div[9]/div[3]/div[1]/div[1]/div/div[2]/form/inp
city.send_keys(input("Enter a city name: "))

# Search button
search_button= driver.find_element(By.XPATH,'/html/body/div[3]/div[9]/div[3]/div[1]/div[1]/div/div[2]
search_button.click()
time.sleep(2)
```

```
Enter a city name: kolkata
```

In [32]:

```
page_url=driver.current_url
lat_lng=re.findall(r"@(.*)data", page_url)
lat_lng
```

Out[32]:

```
['22.5353434,88.1825384,11z/']
```

In [33]:

```
latitude=lat_lng[0].split(",")[0]
longitude=lat_lng[0].split(",")[1]
print("latitude of searched place",latitude)
print("longitude of searched place",longitude)
```

```
latitude of searched place 22.5353434
longitude of searched place 88.1825384
```

In [34]:

```
driver.quit()
```

Q6.

Write a program to scrap all the available details of best gaming laptops from digit.in.

In [35]:

```
#Loading the url
driver=webdriver.Chrome(r"chromedriver.exe")
url='https://www.digit.in/'
driver.get(url)
time.sleep(5)

#click on laptop menu
laptop=driver.find_element(By.XPATH,'/html/body/div[2]/div/ul/li[2]/span')
laptop.click()
time.sleep(1)

#click on best
best=driver.find_element(By.XPATH,'/html/body/div[2]/div/ul/li[2]/div[2]/div/div[1]/span[4]')
best.click()
time.sleep(2)

#click on best laptop
best_laptop=driver.find_element(By.XPATH,'/html/body/div[2]/div/ul/li[2]/div[2]/div/div[5]/div/div[1]
best_laptop.click()
time.sleep(5)

#creating empty list
products=[]
processor=[]
display=[]
os=[]
memory=[]
graphics_processor=[]
body=[]
price=[]


#to get product names
try:
    product_name=driver.find_elements(By.XPATH,'//span[@class="datahreflink"]/h3')
    for product in product_name:
        products.append(product.text)
except NoSuchElementException:
    products.append("-")

#to get specification details
try:
    specification_tables=driver.find_elements(By.CLASS_NAME,"Spcs-details")
    for tables in specification_tables:
        processor.append(tables.text.split("\n")[1].split(":")[1])
        display.append(tables.text.split("\n")[2].split(":")[1])
        os.append(tables.text.split("\n")[3].split(":")[1])
        memory.append(tables.text.split("\n")[4].split(":")[1])
        graphics_processor.append(tables.text.split("\n")[5].split(":")[1])
        body.append(tables.text.split("\n")[6].split(":")[1])
        price.append(tables.text.split("\n")[7].split(":")[1])
except NoSuchElementException:
    processor.append("-")
    display.append("-")
    os.append("-")
    memory.append("-")
    graphics_processor.append("-")
    body.append("-")
    price.append("-")

#creating dataframe
df_Best_Gaming_Laptop=pd.DataFrame({"Product Name":products,
                                    "Processor":processor,
                                    "Display":display,
                                    "OS":os,
                                    "Memory":memory,
                                    "Graphics Processor":graphics_processor,
                                    "Body":body,
```

```
                                "Price":price})
df_Best_Gaming_Laptop
```

Out[35]:

	Product Name	Processor	Display	OS	Memory	Graphics Processor	Body	Price
0	Acer Aspire 7 A715	AMD Ryzen 5-5500U processor with 2.1 GHz cloc...	15.6″ (1920 x 1080) screen	Windows 11 Home	16 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce GTX 1650 Graphics card	255 x 363 x 23 mm dimension & 2.15 kg weight	₹ 49,990
1	MSI GF63 Thin	11th Gen Intel Core i5-11400H processor with ...	15.6″ (1280 x 720) screen, 144 Hz refresh rate	Windows 10 Home	8 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce GTX1650 Graphics card	359 x 254 x 22 mm dimension & 1.86 kg weight	₹ 52,990
2	Acer Nitro 5 AN515-57	11th Gen Intel Core i5-11400H processor with ...	15.6″ (1920 x 1080) screen, 144 Hz refresh rate	Windows 11	16 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce GTX 1650 Graphics card	255 x 363 x 24 mm dimension & 2.2 kg weight	₹ 64,350
3	Lenovo Legion 5	4th Gen AMD Ryzen 5-(4600H) 6 core processor ...	15.6″ (1920 x 1080) screen, 120 Hz refresh rate	Windows 10 Home	8 GB DDR4GB RAM & 1 TB HDD with 256 GB SSD	NVIDIA GeForce GTX 1650 Graphics card	36.3 x 26 x 2.4 mm dimension & 2.3 kg weight	₹ 74,990
4	Lenovo Ideapad Gaming 3	11th Gen Intel Core i5-11300H 4 core processo...	15.6″ (1920 x 1080) screen, 120 Hz refresh rate	Windows 11 Home	8 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce GTX 1650 Graphics card	360 x 252 x 24 mm dimension & 2.25 kg weight	₹ 51,990
5	HP Pavilion 15 (With Updated Specs)	AMD Ryzen 7-5800H 8 core processor with 4.4 G...	15.6″ (1920 x 1080) screen	Windows 11 Home	16 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce RTX 3050 Graphics card	360 x 257 x 24 mm dimension & 1.98 kg weight	₹ 84,990
6	HP Victus 16-D0354TX	11th Gen Intel Core i5-11400H 6 core processo...	16.1″ (1920 x 1080) screen, 144 Hz refresh rate	Windows 11 Home	8 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce RTX 3050 Graphics card	370 x 260 x 23.5 mm dimension & 2.46 kg weight	₹ 66,990
7	Dell G15 511 Gaming Laptop	11th Gen Intel Core i5-11260H 6 core processo...	15.6″ (1920 x 1080) screen, 120 Hz refresh rate	Windows 11 Home	8 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce RTX 3050 Graphics card	272 x 357 x 27 mm dimension & 2.65 kg weight	₹ 67,690
8	Acer Aspire 5 A515-57G	12th Gen Intel Core i5-1240P processor with 3...	15.6″ (1920 x 1080) screen	Windows 11 Home	8 GB DDR4GB RAM & 512 GB SSD	4 GB DDR6 NVIDIA GeForce RTX 2050 Graphics card	238 x 363 x 18 mm dimension & 1.8 kg weight	₹ 59,990

In [36]:

```
driver.quit()
```

Q7.

Write a python program to scrape the details for all billionaires from www.forbes.com (http://www.forbes.com). Details to be scrapped: “Rank”, “Name”, “Net worth”, “Age”, “Citizenship”, “Source”, “Industry”.

In [38]:

```
driver=webdriver.Chrome(r"chromedriver.exe")
driver.get("https://www.forbes.com/")
time.sleep(5)

#click on menu button
menu_button=driver.find_element(By.XPATH,'//div[@aria-label="Open Navigation Menu"]')
menu_button.click()
time.sleep(1)

#Click on right arrow
arrow=driver.find_element(By.XPATH,'//div[@role="button"]/span')
arrow.click()
time.sleep(1)

#click on world billionaris
billo=driver.find_element(By.XPATH,'//li[@class="TjJgrPSg _2bNo56RE secondary"]/a')
billo.click()
WebDriverWait(driver,20).until(EC.presence_of_element_located((By.XPATH,'//li[@class="_2eUBXdbq"]')))
```

Out[38]:

```
<selenium.webdriver.remote.webelement.WebElement (session="e835de2cc758a3ff1128b7de0e2fc02
d", element="B3EEFD15E683EDDA7BAC5F24A6859FE9_element_104")>
```

In [39]:

```
#creating empty list
sl_no=[]
name=[]
networth=[]
age=[]
country=[]
source=[]



# inserting 1st row
try:
    row_1=driver.find_element(By.XPATH,'//div[@class="TableRow_row__L-0Km TableRow_activeRow__g4oSF"]
    sl_no.append(row_1.text.split("\n")[0])
    name.append(row_1.text.split("\n")[1])
    networth.append(row_1.text.split("\n")[2])
    age.append(row_1.text.split("\n")[3])
    country.append(row_1.text.split("\n")[4])
    source.append(row_1.text.split("\n")[5])

except NoSuchExceptionError:
    sl_no.append("-")
    name.append("-")
    networth.append("-")
    age.append("-")
    country.append('-')
    source.append("-")



# inserting rest rows
entire_rows=driver.find_elements(By.XPATH,'//div[@class="TableRow_row__L-0Km"]')
try:
    for row in entire_rows:
        sl_no.append(row.text.split("\n")[0])
        name.append(row.text.split("\n")[1])
        networth.append(row.text.split("\n")[2])
        age.append(row.text.split("\n")[3])
        country.append(row.text.split("\n")[4])
        source.append(row.text.split("\n")[5])

except NoSuchExceptionError:
    sl_no.append("-")
    name.append("-")
    networth.append("-")
    age.append("-")
    country.append('-')
    source.append("-")




#counting length
print("sl no    : ",len(sl_no))
print("Name     : ",len(name))
print("Networth : ",len(networth))
print("Age      : ",len(age))
print("country  : ",len(country))
print("Source   : ",len(source))

```

```
sl no    :  200
Name     :  200
Networth :  200
Age      :  200
country  :  200
Source   :  200
```

In [40]:

```
#making DataFrame
billionaires=pd.DataFrame({"SL No": sl_no,
                           "NAME" : name,
                           "NETWORTH" : networth,
                           "AGE" : age,
                           "COUNTRY" : country,
                           "SOURCE" : source})
billionaires
```

Out[40]:

	SL No	NAME	NETWORTH	AGE	COUNTRY	SOURCE
0	1	Bernard Arnault & family	$211 B	74	France	LVMH
1	2	Elon Musk	$180 B	51	United States	Tesla, SpaceX
2	3	Jeff Bezos	$114 B	59	United States	Amazon
3	4	Larry Ellison	$107 B	78	United States	Oracle
4	5	Warren Buffett	$106 B	92	United States	Berkshire Hathaway
...	...	...	...	...	...	...
195	195	Jin Baofang	$9.6 B	70	China	Solar panels
196	195	Luo Liguo & family	$9.6 B	67	China	Chemicals
197	195	Marijke Mars	$9.6 B	58	United States	Candy, pet food
198	195	Pamela Mars	$9.6 B	62	United States	Candy, pet food
199	195	Valerie Mars	$9.6 B	64	United States	Candy, pet food

200 rows × 6 columns

In [41]:

```
driver.quit()
```

Q8.

Write a program to extract at least 500 Comments, Comment upvote and time when comment was posted from any YouTube Video.

In [42]:

```
driver=webdriver.Chrome(r"chromedriver.exe")
url='https://www.youtube.com/'
driver.get(url)
driver.maximize_window()

# serching
key='Imagine Dragons - Believer (Lyrics)'
search=driver.find_element(By.XPATH,'/html/body/ytd-app/div[1]/div/ytd-masthead/div[4]/div[2]/ytd-sea
search.send_keys(key)

#tao on search button
button=driver.find_element(By.XPATH,'//button[@id="search-icon-legacy"]')
button.click()
time.sleep(3)
#click on video
video=driver.find_element(By.XPATH,'//div[@class="style-scope ytd-video-renderer"]')
video.click()

#explicitly wait for skip ad button and click on it.

try:
    WebDriverWait(driver,30).until(EC.element_to_be_clickable((By.XPATH,'//button[@class="ytp-ad-skip
    skip_ad=driver.find_element(By.XPATH,'//button[@class="ytp-ad-skip-button ytp-button"]')
    skip_ad.click()
except:
    print("No need to click 'skip ad' button")

# click on "no thanks"
try:
    WebDriverWait(driver,10).until(EC.element_to_be_clickable((By.XPATH,'/html/body/ytd-app/ytd-popup
    no_thanks=driver.find_element(By.XPATH,'/html/body/ytd-app/ytd-popup-container/tp-yt-paper-dialog
    no_thanks.click()
except:
    print("no need to click 'no thanks' button")
#enjoy some time with music
time.sleep(180)



#click on pause button
pause=driver.find_element(By.XPATH,'//button[@class="ytp-play-button ytp-button"]')
pause.click()
time.sleep(10)

#scrolling the page
for _ in range(102):
    driver.execute_script("window.scrollBy(0,600)")
    time.sleep(3)

#creating empty list
comment=[]
upvote=[]
post_time=[]

#comments
try:
    comm=driver.find_elements(By.ID,"content-text")
    for c in comm:
        comment.append(c.text)
except:
    comment.append("-")

#upvote
try:
    vote=driver.find_elements(By.ID,"vote-count-middle")
    for v in vote:
        upvote.append(v.text)
except:
```

```
    upvote.append("-")
try:
    timings=driver.find_elements(By.XPATH,'//yt-formatted-string[@class="published-time-text style-sc
    for t in timings:
        post_time.append(t.text)
except:
    post_time.append("-")

print(len(comment))
print(len(upvote))
print(len(post_time))

#making datadrame
comment_df=pd.DataFrame({"COMMENT" : comment[0:500],
                         "COMMENT UPVOTE" : upvote[0:500], "TIME when comment was posted" : post_time
comment_df
```

```
520
520
520
```

Out[42]:

	COMMENT	COMMENT UPVOTE	TIME when comment was posted
0	Hey guys, we have finally got our new instagra...	40K	3 years ago (edited)
1	5 years passed but this song is still a Master...	1K	1 month ago
2	1 hour past and this song is still a masterpiece	44	5 days ago
3	5 years passed but still masterpiece	438	2 months ago
4	These guys are so underrated. Their lyrics are...	78	2 weeks ago
...	...	...	...
495	The song Is the best	2	1 month ago
496	It is now 2023in June and I still love this song		6 days ago
497	The energy level of this song is superb\nWho l...	3	9 days ago
498	MASTER PIECE	1	3 weeks ago
499	It's crazy how many people legends comback to ...	301	1 year ago

500 rows × 3 columns

In [43]:

```
1 comment_df.sample(20)
```

Out[43]:

	COMMENT	COMMENT UPVOTE	TIME when comment was posted
329	It's just great		1 month ago
356	This is the best song I have ever heard.	1	2 months ago
363	I listen this song every day		1 month ago
232	Outstanding	5	5 days ago
175	5 years but still a masterpiece		7 days ago
246	In 2023 this song is stilla master piece	1	2 weeks ago
223	This song motiavates us to be fearless and do ...	490	1 year ago
248	The best song, even my school's in love with it.	1	1 month ago
119	Música maravilhosa	132	1 year ago
306	I love this song soooooo!! much	1	4 weeks ago
360	I love this song too much		3 weeks ago
393	I like it	3	3 weeks ago
338	loved it when i am sad i play it.	1	2 months ago
237	Awesome Best Song Ever!	1	1 month ago
479	I love your videos so much keepmaking them		1 month ago
317	250 years passed,still this song is more than ...	7	1 month ago
493	Srsly tho such a good song i could listen 2 it...	107	2 years ago
284	I heard this song 100 times amazing lyrics top...	1	2 months ago
313	I love this song I can't stop listening to it.	55	2 years ago
135	Never gonna give you up\nNever gonna let you d...	5	1 month ago

In [44]:

```
1 driver.quit()
```

Q9.

Write a python program to scrape a data for all available Hostels from https://www.hostelworld.com/ (https://www.hostelworld.com/) in "London" location. You have to scrape hostel name, distance from city centre, ratings, total reviews, overall reviews, privates from price, dorms from price, facilities and property description.

In [47]:

```
#getting driver and loading url
url='https://www.hostelworld.com/'
driver=webdriver.Chrome(r"chromedriver.exe")
driver.maximize_window()
driver.get(url)
time.sleep(5)
#searching element:
search_clk=driver.find_element(By.XPATH,'//input[@id="location-text-input-field"]')
search_clk.click()
time.sleep(3)
search=driver.find_element(By.CLASS_NAME,"search-input")
search.send_keys("London")
WebDriverWait(driver,10).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[3]/div/div/div[2]
london=driver.find_element(By.XPATH,'/html/body/div[3]/div/div/div[2]/div[2]/div/div/div[4]/div/div[2
london.click()
#Lets go button click
lets_go=driver.find_element(By.XPATH,'//div[@class="search-button"]/button')
lets_go.click()
```

In [48]:

```
hostel_name=[]
distance=[]
ratings=[]
total_reviews=[]
overall_reviews=[]
privates_from_price=[]
dorms_from_price=[]


#for hotel name
try:
    hostels=driver.find_elements(By.XPATH,'//h2[@class="title title-6"]/a')
    for h in hostels:
        hostel_name.append(h.text)
except:
    hostel_name.append("-")

# distance
try:
    dists=driver.find_elements(By.XPATH,'//div[@class="subtitle body-3"]/a/span/span')
    for d in dists:
        distance.append(d.text)
except:
    distance.append('-')

# for ratings
try:
    rtngs=driver.find_elements(By.XPATH,'//div[@class="rating rating-summary-container big"]/div')[:
    for r in rtngs:
        ratings.append(r.text)
except:
    ratings.append('-')

#Total review
try:
    rvws=driver.find_elements(By.XPATH,'//div[@class="rating rating-summary-container big"]/div')[1:
    for r in rvws:
        total_reviews.append(r.text.split('\n')[1].split(" ")[0])
        overall_reviews.append(r.text.split('\n')[0])
except:
    total_reviews.append("-")
    overall_reviews.append("-")

#for private from price
try:
    pvt_prc=driver.find_elements(By.XPATH,'//div[@class="price-col"]')[::2]
    for r in pvt_prc:
        a=r.text
        if a.startswith("Privates"):
            b=a.split("\n")[1]
            if b.isspace():
                privates_from_price.append(b.split(" ")[1])
            else:
                privates_from_price.append(b)
        elif a.startswith("No"):
            privates_from_price.append(a)
except:
    privates_from_price.append("-")
#for Dorms price
try:
    drm_prc=driver.find_elements(By.XPATH,'//div[@class="price-col"]')[1::2]


    for r in drm_prc:
        a=r.text
        if a.startswith("Dorms"):
            b=a.split("\n")[1]
            if b.isspace():
```

```
                dorms_from_price.append(b.split(" ")[1])
            else:
                dorms_from_price.append(b)
        elif a.startswith("No"):
            dorms_from_price.append(a)
except:
    dorms_from_price.append("-")



print(len(hostel_name),len(distance),len(ratings),len(total_reviews), len(overall_reviews), len(priva
```

```
30 30 30 30 30 30 30
```

In [49]:

```
facility=[]
fci=driver.find_elements(By.XPATH,'//div[@class="facilities-label facilities"]')
for i in fci:
    facility.append(i.text)
len(facility)
```

Out[49]:

```
30
```

In [50]:

```
desp=[]
description=driver.find_elements(By.XPATH,'//div[@class="rating-factors prop-card-tablet rating-facto
for i in description:
    desp.append(i.text)
len(desp)
```

Out[50]:

```
30
```

In [51]:

```
#making DataFrame
hostel_df=pd.DataFrame({"hostel_name" : hostel_name,
                        "distance" :distance,
                        "ratings" : ratings,
                        "total_reviews " : total_reviews ,
                        "overall_reviews" :overall_reviews,
                        "privates_from_price" : privates_from_price,
                        "dorms_from_price" : dorms_from_price,
                        "facilities": facility,
                        "Description" : desp})
hostel_df
```

Out[51]:

	hostel_name	distance	ratings	total_reviews	overall_reviews	privates_from_price	dorms_from_price	
0	Wombat's City Hostel London	- 3.6km from city centre	9.0	14906	Superb	Rs24242.05 Rs19394	Rs7245.5 Rs5796	Wi
1	Onefam Notting Hill by Hostel One	- 5.5km from city centre	9.6	2060	Superb	No Privates Available	Rs7785	
2	St Christopher's Village	- 1.8km from city centre	7.9	12155	Very Good	No Privates Available	Rs3988.26 Rs3589	Wi
3	Urbany Hostel London	- 5.4km from city centre	9.4	781	Superb	Rs20242	Rs7174	Wi
4	NX London Hostel	- 6.1km from city centre	8.3	1886	Fabulous	No Privates Available	Rs4708.51 Rs3802	Free
5	Generator London	- 3km from city centre	7.4	7504	Very Good	Rs11869	Rs4933	
6	Safestay London Elephant & Castle	- 1.7km from city centre	7.5	4959	Very Good	No Privates Available	Rs2962	Wi
7	Phoenix Hostel	- 4.2km from city centre	7.5	4129	Very Good	No Privates Available	Rs4401.02 Rs4181	Wi
8	Safestay London Kensington Holland Park	- 5.8km from city centre	7.1	1529	Very Good	No Privates Available	Rs2790	
9	London Backpackers	- 11.9km from city centre	7.8	4470	Very Good	No Privates Available	Rs2330	Wi
10	No.8 Seven Sisters	- 9km from city centre	5.7	4011	Rating	No Privates Available	Rs1569	Wi
11	Queen Elizabeth Chelsea	- 5.7km from city centre	7.4	3481	Very Good	No Privates Available	Rs2833	
12	St Christopher's Inn - London Bridge	- 1.8km from city centre	8.0	3506	Fabulous	No Privates Available	Rs3949 Rs3554	Wi
13	St Christopher's Hammersmith	- 7.5km from city centre	7.7	4244	Very Good	No Privates Available	Rs2648.09 Rs2383	Wi

	hostel_name	distance	ratings	total_reviews	overall_reviews	privates_from_price	dorms_from_price	
14	International Students House	- 3.3km from city centre	9.3	1036	Superb	Rs13468	No Dorms Available	Wi
15	St Christopher's Camden	- 4.3km from city centre	7.0	4035	Very Good	No Privates Available	Rs3763.21 Rs3387	Wi
16	St Christopher's Greenwich	- 7.6km from city centre	7.5	3350	Very Good	No Privates Available	Rs2667.52 Rs2401	Wi
17	Bell House Hostel	- 4.2km from city centre	7.8	61	Very Good	No Privates Available	Rs2739	
18	Saint James Backpackers	- 5.5km from city centre	7.5	1884	Very Good	Rs12239	Rs5842	Free Breakfa Cc
19	PubLove @ The Steam Engine, Waterloo	- 0.5km from city centre	8.0	394	Fabulous	No Privates Available	Rs7871	
20	St Christopher's Shepherds Bush	- 7km from city centre	7.5	772	Very Good	No Privates Available	Rs2822.99 Rs2541	Wi
21	Book a Bed Hostels	- 6.9km from city centre	7.4	1255	Very Good	Rs6646.44 Rs5982	Rs2831.16 Rs2548	
22	Selina Camden	- 5.5km from city centre	8.9	50	Fabulous	Rs29678	Rs12048	
23	PubLove @ The Crown, Battersea	- 4.7km from city centre	7.5	301	Very Good	No Privates Available	Rs4338	
24	Kensal Green Backpackers	- 8.2km from city centre	5.0	3710	Rating	Rs9620	Rs2624	Follov
25	Hootananny Hostel	- 5km from city centre	7.6	1511	Very Good	No Privates Available	Rs3190	Free
26	Hostelle London	- 5.1km from city centre	8.7	11	Fabulous	No Privates Available	Rs7346.07 Rs6244	Wi
27	PubLove @ The White Ferry, Victoria	- 2.4km from city centre	7.3	338	Very Good	No Privates Available	Rs5947	
28	Elmwood Hotel	- 3.2km from city centre	7.7	128	Very Good	Rs22738	No Dorms Available	
29	London Waterloo Hostel	- 0.7km from city centre	5.5	2534	Rating	Rs13577	Rs3331	Wi

In [52]:
```
driver.quit()
```

In [ ]:

```
1
```