In [3]:
```
!pip install selenium
```

```
Requirement already satisfied: selenium in c:\users\dravin mishra\anaconda3 2\lib\site-packages (4.17.2)
Requirement already satisfied: urllib3[socks]<3,>=1.26 in c:\users\dravin mishra\anaconda3 2\lib\site-package
s (from selenium) (1.26.16)
Requirement already satisfied: trio~=0.17 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from selen
ium) (0.24.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (f
rom selenium) (0.11.1)
Requirement already satisfied: certifi>=2021.10.8 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (fr
om selenium) (2023.7.22)
Requirement already satisfied: typing_extensions>=4.9.0 in c:\users\dravin mishra\anaconda3 2\lib\site-packag
es (from selenium) (4.9.0)
Requirement already satisfied: attrs>=20.1.0 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from tr
io~=0.17->selenium) (22.1.0)
Requirement already satisfied: sortedcontainers in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from
trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: idna in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from trio~=0.17-
>selenium) (3.4)
Requirement already satisfied: outcome in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from trio~=0.
17->selenium) (1.3.0.post0)
Requirement already satisfied: sniffio>=1.3.0 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from t
rio~=0.17->selenium) (1.3.0)
Requirement already satisfied: cffi>=1.14 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from trio~
=0.17->selenium) (1.15.1)
Requirement already satisfied: wsproto>=0.14 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from tr
io-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: PySocks!=1.5.7,<2.0,>=1.5.6 in c:\users\dravin mishra\anaconda3 2\lib\site-pac
kages (from urllib3[socks]<3,>=1.26->selenium) (1.7.1)
Requirement already satisfied: pycparser in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from cffi>=
1.14->trio~=0.17->selenium) (2.21)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\dravin mishra\anaconda3 2\lib\site-packages (from ws
proto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
```

In [125]:
```
import selenium
from selenium import webdriver
import time
import warnings
warnings.filterwarnings('ignore')
from selenium.webdriver.common.by import By
from bs4 import BeautifulSoup
import requests

import pandas as pd

#importing required exceptions
from selenium.common.exceptions import StaleElementReferenceException , NoSuchElementException

from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
```

In [ ]:
```
#Q1) Write a python program which searches all the product under a particular product from 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 [44]:
```
driver = webdriver.Chrome()
driver.get('https://www.amazon.in/')
time.sleep(5)
```

In [30]:
```

user_input = input("Enter product name ") #taking input from user
search_box = driver.find_element(By.XPATH , '/html/body/div[1]/header/div/div[1]/div[2]/div/form/div[2]/div[1]
search_box.send_keys(user_input)

search_button = driver.find_element(By.XPATH , '/html/body/div[1]/header/div/div[1]/div[2]/div/form/div[3]/div
search_button.click()
```

```
Enter product name guitar
```

In [ ]:
```
#Q2) In the above question, now scrape the following details of each product listed in first 3 pages of your s
# results and save it in a data frame and csv. In case if any product has less than 3 pages in search results
# 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 [31]:
```
start = 0
end = 3
```

In [ ]:
```

```

In [32]:
```
# from selenium.common.exceptions import TimeoutException, StaleElementReferenceException
# from selenium.webdriver.common.by import By
# from selenium.webdriver.support.ui import WebDriverWait
# from selenium.webdriver.support import expected_conditions as EC

product_url = []

for page in range(start, end):
    try:
        url = driver.find_elements(By.XPATH , '//a[@class="a-link-normal s-underline-text s-underline-link-tex
#         url = driver.find_elements(By.XPATH, '//div[@class="a-section a-spacing-small puis-padding-left-smal

        for i in url:
            try:
                product_url.append(i.get_attribute("href"))
            except:
                product_url.append("-")

        delay = 20  # in seconds
        wait = WebDriverWait(driver, delay)

        next_button_xpath = '//a[@class="s-pagination-item s-pagination-next s-pagination-button s-pagination-
        next_button = wait.until(EC.element_to_be_clickable((By.XPATH, '//a[@class="s-pagination-item s-pagina
        next_button.click()

    except TimeoutException:
        print("TimeoutException: Waiting for the next button took too long.")
        break  # Break out of the loop if TimeoutException occurs

    except StaleElementReferenceException:
        print("StaleElementReferenceException: Retrying to locate elements.")
        continue  # Retry the loop if StaleElementReferenceException occurs

print(product_url)
```

['https://www.amazon.in/sspa/click?ie=UTF8&spc=MTo5Mjk1NDY2NTgyODkwNTI6MTcwODU4NDQ5MTpzcF9hdGY6MjAwOTY2MDg3NTA0OTg6OjA6Og&url=%2FKadence-Frontier-Acoustic-Guitar-Strings%2Fdp%2FB01GDZ46AA%2Fref%3Dsr_1_1%3Fdib%3DeyJ2IjoiMSJ9.crxhhcuVe8Gag3-0mnCXJiOSSKJki5hOYLJU7nup4DW59QOn3hymzKiKSHkyawR6gJfZjQy82hz491UcZZnvLcKKhewsG9sZkT3mc4fnxq8CZBGEp-a2AhGKs6FYimMQsNgK6O0YoIcaJYc5qCErhvOuR3Kbr0uo29IKzo0lrVT9QqF0s0immzN1PfQxFE8kOQexRsAgco0RjEt2F8_fID5LvY80YtQWsrsr9-AzfOOIufFe_AQSdz_Qd7-dm8ZuX1xG_Glpn3pv5LLHcpsNs1YEd5Krhw7ESU2TdKPTybU.DOXoTri9Pfo8vqOK6fM8VFPPjn9UEFhMQz7qq0QPF1s%26dib_tag%3Dse%26keywords%3Dguitar%26qid%3D1708584491%26sr%3D8-1-spons%26sp_csd%3Dd2lkZ2V0TmFtZT1zcF9hdGY%26psc%3D1', 'https://www.amazon.in/sspa/click?ie=UTF8&spc=MTo5Mjk1NDY2NTgyODkwNTI6MTcwODU4NDQ5MTpzcF9hdGY6MzAwMDE1NTM3OTc5MjMyOjowOjo&url=%2FKadence-Frontier-Acoustic-Guitar-Strings%2Fdp%2FB078GTJP5Y%2Fref%3Dsr_1_2%3Fdib%3DeyJ2IjoiMSJ9.crxhhcuVe8Gag3-0mnCXJiOSSKJki5hOYLJU7nup4DW59QOn3hymzKiKSHkyawR6gJfZjQy82hz491UcZZnvLcKKhewsG9sZkT3mc4fnxq8CZBGEp-a2AhGKs6FYimMQsNgK6O0YoIcaJYc5qCErhvOuR3Kbr0uo29IKzo0lrVT9QqF0s0immzN1PfQxFE8kOQexRsAgco0RjEt2F8_fID5LvY80YtQWsrsr9-AzfOOIufFe_AQSdz_Qd7-dm8ZuX1xG_Glpn3pv5LLHcpsNs1YEd5Krhw7ESU2TdKPTybU.DOXoTri9Pfo8vqOK6fM8VFPPjn9UEFhMQz7qq0QPF1s%26dib_tag%3Dse%26keywords%3Dguitar%26qid%3D1708584491%26sr%3D8-2-spons%26sp_csd%3Dd2lkZ2V0TmFtZT1zcF9hdGY%26psc%3D1', 'https://www.amazon.in/sspa/click?ie=UTF8&spc=MTo5Mjk1NDY2NTgyODkwNTI6MTcwODU4NDQ5MTpzcF9hdGY6MjAwNTcxOTMwNTYyMDQ6OjA6Og&url=%2FKadence-Slowhand-Premium-Acoustic-Instrument%2Fdp%2FB077SZ667X%2Fref%3Dsr_1_3%3Fdib%3DeyJ2IjoiMSJ9.crxhhcuVe8Gag3-0mnCXJiOSSKJki5hOYLJU7nup4DW59QOn3hymzKiKSHkyawR6gJfZjQy82hz491UcZZnvLcKKhewsG9sZkT3mc4fnxq8CZBGEp-a2AhGKs6FYimMQsNgK6O0YoIcaJYc5qCErhvOuR3Kbr0uo29IKzo0lrVT9QqF0s0immzN1PfQxFE8kOQexRsAgco0RjEt2F8_fID5LvY80YtQWsrsr9-AzfOOIufFe_AQSdz_Qd7-dm8ZuX1xG_Glpn3pv5LLHcpsNs1YEd5Krhw7ESU2TdKPTybU.DOXoTri9Pfo8vqOK6fM8VFPPjn9UEFhMQz7qq0QPF1s%26dib_tag%3Dse%26keywords%3Dguitar%26qid%3D1708584491%26sr%3D8-3-spons%26sp_csd%3Dd2lkZ2

In [23]:
```
len(product_url)
```

Out[23]: 140

In [ ]:
```
brand_name = []
price = []
product_name = []
expected_delivery = []
availability = []
return_exchange = []
url_product = []
for url in product_url :
    driver.get(url)
    time.sleep(5)

    try :
        brand = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[4]/div[48]/div/table/t
        brand_name.append(brand.text)
    except NoSuchElementException :
        brand_name.append("-")

    try :
        prce = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[1]/div[3]/div/div[1]/di
        price.append(prce.text)
    except NoSuchElementException :
        price.append("-")

    try :
        pr_name = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[4]/div[1]/div/h1/spa
        product_name.append(pr_name.text)
    except NoSuchElementException :
        product_name.append("-")

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

    try :
        ava = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[1]/div[3]/div/div[1]/div
        availability.append(ava.text)
    except NoSuchElementException :
        availability.append("-")

#     try :
#         exchange = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[4]/div[24]/div[2]
```

```
#         return_exchange.append(exchange.text)
#     except NoSuchElementException :
#         return_exchange.append("-")

data = {
    'Brand Name' : brand_name ,
    'Price' : price ,
    'Name of the product' : product_name ,
    'Expected_Delivery' : expected_delivery ,
    'Availability' : availability ,
    'Return or Exchange' : return_exchange

}

df = pd.DataFrame(data)

# Save the DataFrame to a CSV file
df.to_csv('product_details.csv', index=False)
```

In [159]:
```
brand_name = []
price = []
product_name = []
expected_delivery = []
availability = []
return_exchange = []
url_product = []


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

    try :
        brand = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[4]/div[48]/div/table/t
        brand_name.append(brand.text)
    except NoSuchElementException :
        brand_name.append("-")

    try :
        prce = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[1]/div[3]/div/div[1]/di
        price.append(prce.text)
    except NoSuchElementException :
        price.append("-")

    try :
        pr_name = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[4]/div[1]/div/h1/spa
        product_name.append(pr_name.text)
    except NoSuchElementException :
        product_name.append("-")

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

    try :
        ava = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[1]/div[3]/div/div[1]/div
        availability.append(ava.text)
    except NoSuchElementException :
        availability.append("-")
```

```
    try :
        exchange = driver.find_element(By.XPATH , '/html/body/div[2]/div/div[5]/div[3]/div[4]/div[24]/div[2]/d
        return_exchange.append(exchange.text)
    except NoSuchElementException :
        return_exchange.append("-")

    except TimeoutException:
            print("TimeoutException: Waiting for the next button took too long.")
            break  # Break out of the loop if TimeoutException occurs

    except StaleElementReferenceException:
            print("StaleElementReferenceException: Retrying to locate elements.")
            continue  # Retry the loop if StaleElementReferenceException occurs

data = {
    'Brand Name' : brand_name ,
    'Price' : price ,
    'Name of the product' : product_name ,
    'Expected_Delivery' : expected_delivery ,
    'Availability' : availability ,
    'Return or Exchange' : return_exchange

}

df = pd.DataFrame(data)

# Save the DataFrame to a CSV file
df.to_csv('product_details.csv', index=False)
```

In [ ]:
```
'''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 [52]:
```
driver = webdriver.Chrome()
driver.get('https://www.google.co.in/webhp')
# driver.get('https://www.google.co.in/')
time.sleep(5)
```

In [53]:
```
user_input = input("enter")
search_box = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/form/div[1]/div[1]/div[1]/div/div[2]/tex
search_box.send_keys(user_input)
search_button = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/form/div[1]/div[1]/div[4]/center/inpu
search_button.click()
```

```
entercakes
```

In [54]:
```
#accessing the image tags

image_button = driver.find_element(By.XPATH , '/html/body/div[5]/div/div[4]/div/div/div[1]/div[1]/div/div[1]/d
image_button.click()
```

In [65]:
```
for _ in range(20) :
    driver.execute_script("window.scrollBy(0 , 500)")
images = driver.find_elements(By.XPATH , '//img[@class="rg_i Q4LuWd"]')
img_url = []
for image in images :
    source = image.get_attribute('src')
    if source is not None :
        if (source[0 : 4] == 'http') :
            img_url.append(source)
for i in range(len(img_url)) :
    if i > 10 :
        break
    print("Downloading {0} of {1} images".format(i , 10))
    response = requests.get(img_url[i])
    file = open(r"C:\Users\DRAVIN MISHRA\Pictures\Camera Roll" + str(i) + ".jpg" , "wb")
    file.write(response.content)
```

```
Downloading 0 of 10 images
Downloading 1 of 10 images
Downloading 2 of 10 images
Downloading 3 of 10 images
Downloading 4 of 10 images
Downloading 5 of 10 images
Downloading 6 of 10 images
Downloading 7 of 10 images
Downloading 8 of 10 images
Downloading 9 of 10 images
Downloading 10 of 10 images
```

In [ ]:

In [ ]:
```
# Q4)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 [87]:
```
driver = webdriver.Chrome()
driver.get('https://www.flipkart.com/')
time.sleep(5)
```

In [88]:
```
user_input = input("Enter your Product here : ")
search_box = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[1]/div/div/div/div/div[1]/div/div[1]/di
search_box.send_keys(user_input)
search_button = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[1]/div/div/div/div/div[1]/div/div[1]
search_button.click()
```

```
Enter your Product here : oneplus nord
```

In [89]:
```
product_url = []
url_of_product = driver.find_elements(By.XPATH , '//a[@class="_1fQZEK"]')
for url in url_of_product :
    product_url.append(url.get_attribute('href'))
len(product_url)
#print(product_url)
```

Out[89]:
```
24
```

In [90]:
```
Brand_Name = []
Product_Name = []
Colour = []
Ram = []
Rom = []
Primary_camera = []
Secondary_camera = []
Display_size = []
Battery_capacity = []
Price = []
Product_url = []
```

In [92]:
```
for url in product_url :
    driver.get(url)
    time.sleep(5)

    try :
        product = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[2]/div/div[1]
        Product_Name.append(product)
    except NoSuchElementException:
        Product_Name.append("-")

    try :
        colour = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]/div/
        Colour.append(colour)
    except NoSuchElementException :
        Colour.append("-")

    try :
        ram = driver.find_element(By.XPATH ,'/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]/div/div[
        Ram.append(ram)
    except NoSuchElementException :
        Ram.append("-")

    try :
        rom = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]/div/div
        Rom.append(rom)
    except NoSuchElementException :
        Rom.append("-")

    try :
        primary_cam = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]
        Primary_camera.append(primary_cam)
    except NoSuchElementException :
        Primary_camera.append("-")

    try :
        sec_cam = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]/div
        Secondary_camera.append(sec_cam)
    except NoSuchElementException :
        Secondary_camera.append("-")

    try :
        display = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]/div
        Display_size.append(display)
```

```
    except NoSuchElementException :
        Display_size.append("-")

    try :
        battery = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[8]/div[4]/div
        Battery_capacity.append(battery)
    except NoSuchElementException :
        Battery_capacity.append("-")

    try :
        price = driver.find_element(By.XPATH , '/html/body/div[1]/div/div[3]/div[1]/div[2]/div[2]/div/div[4]/d
        Price.append(price)
    except NoSuchElementException :
        Price.append("-")

#     try :
#         pr_url = driver.find_element(By.XPATH , '/html/body/div/div/div[3]/div[1]/div[2]/div[2]/div/div/div/
#         Product_url.append(pr_url.get_attribute("href"))
#     except NoSuchElementException :
#         Product_url.append("-")

    except TimeoutException:
            print("TimeoutException: Waiting for the next button took too long.")
            break  # Break out of the loop if TimeoutException occurs

    except StaleElementReferenceException:
            print("StaleElementReferenceException: Retrying to locate elements.")
            continue  # Retry the loop if StaleElementReferenceException occurs

#     product_details = {
#             "Brand Name": Brand_Name,
#             "Smartphone Name": Product_Name,
#             "Colour": Colour,
#             "RAM": Ram,
#             "Storage(ROM)": Rom,
#             "Primary Camera": Primary_camera,
#             "Secondary Camera": Secondary_camera,
#             "Display Size": Display_size,
#             "Battery Capacity": Battery_capacity,
#             "Price": Price,
#         }

# df = pd.DataFrame(product_details)
```

```
df = pd.DataFrame({
    "Smartphone Name": Product_Name,
    "Colour": Colour,
    "RAM": Ram,
    "Storage(ROM)": Rom,
    "Primary Camera": Primary_camera,
    "Secondary Camera": Secondary_camera,
    "Display Size": Display_size,
    "Battery Capacity": Battery_capacity,
    "Price": Price,
})
df.to_csv('smartphone_details.csv', index=False)
```

In [ ]:

In [ ]:
```
# Q5)Write a program to scrap geospatial coordinates (latitude, longitude) of a city searched on google maps.
```

In [93]:
```
driver = webdriver.Chrome()
driver.get('https://www.google.com/maps')
time.sleep(5)
```

In [94]:
```
user_input = input("Enter location : ")
search_box = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[8]/div[3]/div[1]/div[1]/div/div[2]/f
search_box.send_keys(user_input)
search_button = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[8]/div[3]/div[1]/div[1]/div/div[2
search_button.click()
```

```
Enter location : varanasi
```

In [101]:
```
import regex as re
```

In [109]:
```
try :
    url_string = driver.current_url
    print("URL EXTRACTED : " , url_string)
    #lat_lng = re.findall(r'@(.*)data' , url_string)
    lat_lng = url_string.split('@')[1].split(',')[0:2]
    latitude , longitude = map(float , lat_lng)
    print(f'Coordinats of {user_input} : Latitude {latitude} , Longitude {longitude}')
except Exception as e :
    print("Exception raised")
```

```
URL EXTRACTED :  https://www.google.com/maps/place/Varanasi,+Uttar+Pradesh/@25.3209818,82.8266361,11z/data=!3m1!4b1!4m6!3m5!1s0x398e2db76febcf4d:0x68131710853ff0b5!8m2!3d25.3176452!4d82.9739144!16zL20vMDFqOTIy?entry=ttu (https://www.google.com/maps/place/Varanasi,+Uttar+Pradesh/@25.3209818,82.8266361,11z/data=!3m1!4b1!4m6!3m5!1s0x398e2db76febcf4d:0x68131710853ff0b5!8m2!3d25.3176452!4d82.9739144!16zL20vMDFqOTIy?entry=ttu)
Coordinats of varanasi : Latitude 25.3209818 , Longitude 82.8266361
```

In [ ]:

In [ ]:
```
# Q6)Write a program to scrap all the available details of best gaming laptops from digit.in
```

In [117]:
```
driver = webdriver.Chrome()
driver.get('https://www.digit.in/top-products/best-gaming-laptops-40.html')
time.sleep(5)
```

In [120]:
```
product_name = []
product = driver.find_elements(By.XPATH , '//div[@class="rh_gr_top_middle mb10 colored_rate_bar"]/h3/a')
for name in product :
    product_name.append(name.text)
print(product_name)
```

```
['HP Omen 17-ck2008TX 13th Gen Core i7-13700HX', 'MSI GT77 Titan 12UHS-054IN 12th Gen Core i9-12900HX', 'Lenovo Legion 5i Pro 12th Gen Core i7-12700H (82RF00E1IN)', 'ASUS ROG Strix Scar 18 G834JZ-N5041WS 13th Gen Core i9-13980HX', 'Acer Predator Helios Neo 16 13th Gen Core i7-13700HX (PHN16-71)', 'ASUS ROG Zephyrus G14 Ryzen 9-6900HS (GA402RJZ-L4136WS)', 'MSI Cyborg 15 12th Gen Core i7-12650H (A12VF-205IN)']
```

In [122]:
```
product_url = []
product = driver.find_elements(By.XPATH , '//div[@class="rh_gr_top_middle mb10 colored_rate_bar"]/h3/a')
for name in product :
    product_url.append(name.get_attribute('href'))
print(product_url)
```

```
['https://www.digit.in/laptops/hp-omen-17-ck2008tx-13th-gen-core-i7-13700hx-price-346659.html', 'https://www.
digit.in/laptops/msi-gt77-titan-12uhs-054in-12th-gen-core-i9-12900hx-2022-price-330036.html', 'https://www.di
git.in/laptops/lenovo-legion-5i-pro-12th-gen-core-i7-12700h-82rf00e1in-price-346668.html', 'https://www.digi
t.in/laptops/asus-rog-strix-scar-18-price-344598.html', 'https://www.digit.in/laptops/acer-predator-helios-ne
o-16-13th-gen-core-i7-13700hx-phn16-71-price-346677.html', 'https://www.digit.in/laptops/asus-rog-zephyrus-g1
4-ryzen-9-6900hs-ga402rjz-l4136ws-price-346686.html', 'https://www.digit.in/laptops/msi-cyborg-15-12th-gen-co
re-i7-12650h-a12vf-205in-price-346695.html']
```

In [126]:
```
name_product = []
oprating_system = []
processor = []
storage = []
display = []
availability = []
price = []
```

In [130]:
```
for url in product_url :
    driver.get(url)
    time.sleep(5)

    try :
        name = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[1]/div/div/div/div/div[2]/div[1]/d
        name_product.append(name.text)
    except NoSuchElementException :
        name_product.append("-")

    try :
        os = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[1]/div/div/div/div/div[2]/div[1]/div
        oprating_system.append(os.text)
    except NoSuchElementException :
        oprating_system.append("-")

    try :
        prcsr = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[1]/div/div/div/div/div[2]/div[1]/
        processor.append(prcsr.text)
    except NoSuchElementException :
        processor.append("-")

    try :
        store = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[1]/div/div/div/div/div[2]/div[1]/
        storage.append(store.text)
    except NoSuchElementException :
        storage.append("-")

    try :
        display_size = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[1]/div/div/div/div/div[2]/
        display.append(display_size.text)
    except NoSuchElementException :
        display.append("-")

    try :
        available = driver.find_element(By.XPATH , '/html/body/div[1]/div[3]/div[1]/div/div/div/div/div[2]/div
        availability.append(available.text)
    except NoSuchElementException :
        availability.append("-")

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

```
    except NoSuchElementException :
        price.append("-")

    except TimeoutException:
            print("TimeoutException: Waiting for the next button took too long.")
            break  # Break out of the loop if TimeoutException occurs

    except StaleElementReferenceException:
            print("StaleElementReferenceException: Retrying to locate elements.")
            continue  # Retry the loop if StaleElementReferenceException occurs

product_details = {
    'Name of the product' : name_product ,
    'Operating System' : oprating_system ,
    'Processor' : processor ,
    'Storage' : storage ,
    'Display Size' : display ,
    'Availability' : availability ,
    'Price' : price
}

df = pd.DataFrame(product_details)
df
```

In [ ]:

In [ ]:
```
'''Q7)Write a python program to scrape the details for all billionaires from www.forbes.com. Details to be scr
“Rank”, “Name”, “Net worth”, “Age”, “Citizenship”, “Source”, “Industry”.'''
```

In [132]:
```
driver = webdriver.Chrome()
driver.get('https://www.forbes.com/billionaires/')
time.sleep(5)
```

In [134]:
```
Rank = []
rank = driver.find_elements(By.XPATH , '//div[@class="Table_rank___YBhk Table_dataCell__2QCve"]')
for i in rank :
    x = i.text
    Rank.append(x)
len(Rank)
```

Out[134]: 200

In [137]:
```
Rank
```

Out[137]:
```
['1',
 '2',
 '3',
 '4',
 '5',
 '6',
 '7',
 '8',
 '9',
 '10',
 '11',
 '12',
 '13',
 '14',
 '15',
 '16',
 '17',
 '17',
 '19',
```

In [138]:
```
Name = []
name = driver.find_elements(By.XPATH , '//div[@class="TableRow_rowContainer__IC1Tv"]/div/div[1]/div[2]')
for NAME in name :
    Name.append(NAME.text)
len(Name)
```

Out[138]: 200

In [139]:
```
print(Name)
```

```
['Bernard Arnault & family', 'Elon Musk', 'Jeff Bezos', 'Larry Ellison', 'Warren Buffett', 'Bill Gates', 'Mic
hael Bloomberg', 'Carlos Slim Helu & family', 'Mukesh Ambani', 'Steve Ballmer', 'Francoise Bettencourt Meyers
& family', 'Larry Page', 'Amancio Ortega', 'Sergey Brin', 'Zhong Shanshan', 'Mark Zuckerberg', 'Charles Koch
& family', 'Julia Koch & family', 'Jim Walton', 'Rob Walton & family', 'Alice Walton', 'David Thomson & famil
y', 'Michael Dell', 'Gautam Adani', 'Phil Knight & family', 'Zhang Yiming', 'Dieter Schwarz', 'François Pinau
lt & family', 'Klaus-Michael Kuehne', 'Giovanni Ferrero', 'Jacqueline Mars', 'John Mars', 'Li Ka-shing', 'Ma
Huateng', 'Miriam Adelson & family', 'Ken Griffin', 'Mark Mateschitz', 'Robin Zeng', 'Tadashi Yanai & famil
y', 'Len Blavatnik', 'Alain Wertheimer', 'Gerard Wertheimer', 'Gianluigi Aponte', 'Rafaela Aponte-Diamant',
'Colin Huang', 'Reinhold Wuerth & family', 'Lee Shau Kee', 'Jeff Yass', 'Jim Simons', 'Stephen Schwarzman',
'Susanne Klatten', 'Gina Rinehart', 'William Ding', 'Germán Larrea Mota Velasco & family', 'Shiv Nadar', 'Low
Tuck Kwong', 'Thomas Peterffy', 'Andrey Melnichenko & family', 'Stefan Quandt', 'MacKenzie Scott', 'R. Budi H
artono', 'Vladimir Potanin', 'Jack Ma', 'He Xiangjian & family', 'Iris Fontbona & family', 'Michael Hartono',
'James Ratcliffe', 'Cyrus Poonawalla', 'Masayoshi Son', 'Vladimir Lisin', 'Emmanuel Besnier', 'Abigail Johnso
n', 'Leonid Mikhelson & family', 'Lukas Walton', 'Wang Wei', 'Jensen Huang', 'Leonard Lauder', 'Takemitsu Tak
izaki', 'Alexey Mordashov & family', 'Vagit Alekperov', 'Thomas Frist, Jr. & family', 'Andrew Forrest & famil
y', 'Ray Dalio', 'Eric Li', 'Wang Wenyin', 'Eyal Ofer', 'Qin Yinglin', 'Wang Chuanfu', 'Harold Hamm & famil
y', 'David Tepper', 'Gennady Timchenko', 'Daniel Gilbert', 'Lakshmi Mittal', 'Steve Cohen', 'Carl Icahn', 'Sa
vitri Jindal & family', 'Donald Bren', 'John Menard, Jr.', 'Rupert Murdoch & family', 'Vicky Safra & family',
'Theo Albrecht, Jr. & family', 'Renata Kellnerova & family', 'Li Xiting', 'Stefan Persson', 'Eric Schmidt',
'Michael Platt', 'Pang Kang', 'Karl Albrecht Jr. & family', 'Beate Heister', 'Jorge Paulo Lemann & family',
'Peter Woo', 'Dilip Shanghvi', 'Robert Pera', 'Radhakishan Damani', 'Huang Shilin', 'Dhanin Chearavanont', 'D
avid Green & family', 'Charoen Sirivadhanabhakdi', 'Charlene de Carvalho-Heineken & family', 'Xu Hang', 'Wei
Jianjun & family', 'Alisher Usmanov', 'Goh Cheng Liang', 'Kumar Birla', 'Aliko Dangote', 'Kwong Siu-hing', 'I
dan Ofer', 'Chen Bang', 'Lui Che Woo', 'John Fredriksen', 'Diane Hendricks', 'Jan Koum', 'Jerry Jones', 'Geor
ge Kaiser', 'Joseph Lau', 'Lu Xiangyang', 'Harry Triguboff', 'Uday Kotak', 'Stanley Kroenke', 'Mikhail Fridma
n', 'Sarath Ratanavadi', 'Dang Yanbao', 'Jiang Rensheng & family', 'Shahid Khan', 'Laurene Powell Jobs & fami
ly', 'Robert Kuok', 'Stephen Ross', 'Pavel Durov', 'Andreas Struengmann & family', 'Thomas Struengmann & fami
ly', 'Liu Hanyuan', 'Michael Rubin', 'Israel Englander', 'Viatcheslav Kantor', 'Anthony Pratt', 'Mikhail Prok
horov', 'Giorgio Armani', 'Johann Rupert & family', 'Gong Hongjia & family', 'Zhang Zhidong', 'Philip Anschut
z', 'Judy Love & family', 'Ricardo Salinas Pliego & family', 'Donald Newhouse', 'Robert Kraft', 'Marcel Herrm
ann Telles & family', 'Suleiman Kerimov & family', 'Sky Xu', 'Changpeng Zhao', 'Andrew Beal', 'Mike Cannon-Br
ookes', 'Carl Cook', 'David Duffield', 'Jeffery Hildebrand', 'Viktor Rashnikov', 'Eduardo Saverin', 'Georg Sc
haeffler', 'Christy Walton', 'Scott Farquhar', 'Quek Leng Chan', 'Wu Yajun', 'Autry Stephens', 'Liu Yongxin
g', 'Vinod Adani', 'Nicolas Puech', 'Jacques Saadé, Jr.', 'Rodolphe Saadé', 'Tanya Saadé Zeenny', 'Melker Sch
orling & family', 'Andrei Guryev & family', 'Michael Kim', 'Lei Jun', 'Friedhelm Loh', 'Sun Piaoyang', 'Rick
Cohen & family', 'Jin Baofang', 'Luo Liguo & family', 'Marijke Mars', 'Pamela Mars', 'Valerie Mars']
```

In [185]:
```
Net_Worth = []
net_worth = driver.find_elements(By.XPATH , '//div[@class="TableRow_rowContainer__IC1Tv"]/div/div[1]/div[3]/di
for worth in net_worth :
    x = worth.text
    Net_Worth.append(x)
len(Net_Worth)

# Net_Worth = []
# net_worth_elements = driver.find_elements(By.XPATH, '//div[@class="TableRow_rowContainer__IC1Tv"]/div/div/di
# for worth_element in net_worth_elements:
#     Net_Worth.append(worth_element.text)
#     #Net_Worth.append(worth_element)

# print(Net_Worth)
```

Out[185]: 200

In [ ]:
```

```

In [176]:
```
# print(Net_Worth)
```

In [178]:
```
Age = []
age = driver.find_elements(By.XPATH , '//div[@class="TableRow_rowContainer__IC1Tv"]/div/div[1]/div[4]/div')
for i in age :
    x = i.text
    Age.append(x)
len(Age)
```

Out[178]: 200

In [182]:
```
Source = []
source = driver.find_elements(By.XPATH , '//div[@class="TableRow_rowContainer__IC1Tv"]/div/div[1]/div[7]/div')
for i in source :
    x = i.text
    Source.append(x)
len(Source)
```

Out[182]: 200

In [189]:
```python
print(Source)
```

```
['Fashion & Retail', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '']
```

In [177]:
```python
Country = []
country = driver.find_elements(By.XPATH , '//div[@class="TableRow_rowContainer__IC1Tv"]/div/div[1]/div[6]/div'
for state in country :
    x = state.text
    Country.append(x)
len(Country)
```

Out[177]:
```
200
```

In [190]:
```python
Country
```

Out[190]:
```
['LVMH',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 '',
 ''
```

In [179]:
```
import pandas as pd
```

In [186]:
```
list_of_billionaries = {
    'RANK' : Rank ,
    'NAME' : Name ,
    'AGE' : Age ,
    'NET WORTH' : Net_Worth ,
    'SOURCE' : Source ,
    'COUNTRY' : Country
}

df = pd.DataFrame(list_of_billionaries)
df
```

Out[186]:

	RANK	NAME	AGE	NET WORTH	SOURCE	COUNTRY
0	1	Bernard Arnault & family	74	$211 B	Fashion & Retail	LVMH
1	2	Elon Musk	51	$180 B		
2	3	Jeff Bezos	59	$114 B		
3	4	Larry Ellison	78	$107 B		
4	5	Warren Buffett	92	$106 B		
...	...	...	...	...	...	...
195	195	Jin Baofang	70	$9.6 B		
196	195	Luo Liguo & family	67	$9.6 B		
197	195	Marijke Mars	58	$9.6 B		
198	195	Pamela Mars	62	$9.6 B		
199	195	Valerie Mars	64	$9.6 B		

200 rows × 6 columns

In [ ]:
```
'''Q8) Write a program to extract at least 500 Comments, Comment upvote and time when comment was posted
from any YouTube Video. '''
```

In [193]:
```
driver = webdriver.Chrome()
driver.get('https://www.youtube.com/watch?v=S980-z1qx3g')
```

In [*]:
```
for _ in range(1000) :
    driver.execute_script("window.scrollBy(0 , 300)")
    time.sleep(5)
# Extract comments, upvotes, and time
comments = []
upvotes = []
timestamps = []

comment_elements = driver.find_elements(By.XPATH, '//div[@class="style-scope ytd-expander"]/yt-formatted-strin
upvote_elements = driver.find_elements(By.XPATH, '//span[@class="style-scope ytd-comment-engagement-bar"]')
timestamp_elements = driver.find_elements(By.XPATH, '//span[@class="style-scope ytd-comment-view-model"]/a')

num_comments = min(len(comment_elements), 500)

for i in range(num_comments):
    comments.append(comment_elements[i].text)
    upvotes.append(upvote_elements[i].text)
    timestamps.append(timestamp_elements[i].get_attribute('title'))

driver.quit()

for i in range(num_comments):
    print(f"Comment: {comments[i]}")
    print(f"Upvotes: {upvotes[i]}")
    print(f"Timestamp: {timestamps[i]}")
    print("------")
```

In [ ]:

In [ ]:

In [ ]:
```
'''Q9) Write a python program to scrape a data for all available Hostels from 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 [*]:
```
driver = webdriver.Chrome()
driver.get('https://www.hostelworld.com/')
time.sleep(5)
```

In [ ]:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

# Set up the WebDriver
driver = webdriver.Chrome(executable_path='path/to/chromedriver')  # Update with your chromedriver path
driver.get("https://www.hostelworld.com/")
time.sleep(3)  # Let the page load

# Search for hostels in London
search_box = driver.find_element(By.ID, "search-input-field")
search_box.send_keys("London")
search_box.send_keys(Keys.RETURN)
time.sleep(3)  # Wait for the results to load

# Click on the "View More" button to load all results
view_more_button = driver.find_element(By.XPATH, "//button[contains(text(), 'View More')]")
ActionChains(driver).move_to_element(view_more_button).click().perform()
time.sleep(5)  # Wait for the additional results to load

# Get information for each hostel
hostels = driver.find_elements(By.CSS_SELECTOR, ".fabr-card")
for hostel in hostels:
    name = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__title").text
    distance = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__distance").text
    ratings = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__rating").text
    total_reviews = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__count").text
    overall_reviews = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__summary").text
    privates_price = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__price.privates-from-price").text
    dorms_price = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__price.dorms-from-price").text
    facilities = ", ".join([facility.text for facility in hostel.find_elements(By.CSS_SELECTOR, ".fabr-card__f
    description = hostel.find_element(By.CSS_SELECTOR, ".fabr-card__description").text

    # Print or store the information as needed
    print(f"Name: {name}")
    print(f"Distance from City Centre: {distance}")
    print(f"Ratings: {ratings}")
    print(f"Total Reviews: {total_reviews}")
    print(f"Overall Reviews: {overall_reviews}")
```

```
    print(f"Privates from Price: {privates_price}")
    print(f"Dorms from Price: {dorms_price}")
    print(f"Facilities: {facilities}")
    print(f"Description: {description}")
    print("\n" + "-" * 50 + "\n")

# Close the browser
driver.quit()
```