In [ ]:
```
import pdb

import pyautogui

# selenium
import selenium
import pandas as pd
from selenium import webdriver


# BeautifulSoup
from bs4 import BeautifulSoup
import requests

# add time
import time

#to send keys from keyboard
from selenium.webdriver.common.keys import Keys

from selenium.common.exceptions import NoSuchElementException
```

In [ ]:
```
def open_url(url):
    """function to open the entered url in browser"""
    global driver
    # first, connect to the webdriver
    driver=webdriver.Chrome(r'C:/chromedriver.exe')
    # maximize window
    driver.maximize_window()
    #enter the url
    driver.get(url)
    time.sleep(3)
    # handle 'Privacy Error' from Chrome
    try:
        driver.find_element_by_xpath("//div[@class='interstitial-wrapper']//following:

        time.sleep(2)

        driver.find_element_by_xpath("//*[@id='proceed-link']").click()

        time.sleep(2)

    except NoSuchElementException:
        pass
```

In [ ]:
```
product_name=input("Which amazon product do you have in mind ?:")

url='https://www.amazon.in'

open_url(url)

# give some time to load the webpage
time.sleep(2)

# get web element for amazon search bar
search_wbe=driver.find_element_by_xpath("//input[@class='nav-input nav-progressive-att
#enter the product into search bar
```

```
search_wbe.send_keys(product_name,Keys.ENTER)

# give some time to loda the search results
time.sleep(2)
```

In [ ]:

In [ ]:
```
def amzn_search(input_prompt):
    """this function will search a particular product category on amazon.in website"""

    open_url('https://www.amazon.in')

    # give some time to load the webpage
    time.sleep(3)

    # get web element for amazon search bar
    search_wbe=driver.find_element_by_xpath("//input[@class='nav-input nav-progressive
    #enter the product into search bar
    search_wbe.send_keys(str(input_prompt),Keys.ENTER)

    # give some time to loda the search results
    time.sleep(3)
```

In [ ]:
```
def fetch_product():
    # first get all product tiles from current webpage
    #get product tiles webelements
    global tiles_wbe
    tiles_wbe=[]
    tiles_wbe.clear()

    tiles_wbe=driver.find_elements_by_xpath("//div[contains(@data-asin,'B0') and @data

    # click on each tile one by one and get product info
    for i in range(0,len(tiles_wbe)):

        temp_wbe=tiles_wbe[i]
        #add time
        time.sleep(2)
        temp_wbe.click()
        try:
            # switch to new window
            driver.switch_to.window(driver.window_handles[1])
        except IndexError:
            temp_wbe=tiles_wbe[i]
            time.sleep(2)
            temp_wbe.click()
            driver.switch_to.window(driver.window_handles[1])
        # add timer
        time.sleep(3)

        # below part is to enter pincode
        #=========================================================
        # get "enter pincode" button webelement
        pincode_wbe=driver.find_element_by_xpath("//div[@id='contextualIngressPtLabel_

        if pincode_wbe.text=='Select delivery location':
            #click on 'enter pincode' button
            pincode_wbe.click()
            #add time to load pincode sub-window
```

```
            time.sleep(2)
            #get the webelement for pincode entry bar
            pin_bar_wbe=driver.find_element_by_xpath("//input[@class='GLUX_Full_Width
            #enter into pincode bar
            pin_bar_wbe.send_keys('110091')
            #get webelement for pincode 'Apply' button
            apply_btn=driver.find_element_by_xpath("//input[@aria-labelledby='GLUXZipU
            #click on pincode 'Apply' button
            apply_btn.click()
            #add time to load the webpage after pincode enter
            time.sleep(3)
          #=====================================================

        time.sleep(3)
        # calling the function to fetch data from current webpage
        fetch_data()

        # give some time to fetch data
        time.sleep(3)

        # close the particular product tab
        driver.close()

        # switch driver to main window with search results
        driver.switch_to.window(driver.window_handles[0])

        time.sleep(2)
```

In [ ]:

```
def fetch_data():
    """this function gets required data from the current product webpage"""

    # get brand name
    brand_wbe=driver.find_element_by_xpath("//span[@id='productTitle']")
    brands.append(brand_wbe.text.split(' ')[0])
    # get name of the product
    products.append(brand_wbe.text)
    # get the rating
    try:
        rating_wbe=driver.find_element_by_xpath("//span[@data-hook='rating-out-of-text
        ratings.append(rating_wbe.text)
    except NoSuchElementException:
        ratings.append('--')

    # get number of ratings
    try:
        number_ratings_wbe=driver.find_element_by_xpath("//a[@id='acrCustomerReviewLir
        number_ratings.append(number_ratings_wbe.text.split(" ")[0])
    except NoSuchElementException:
        number_ratings.append('--')

    # get price
    try:
        price_wbe=driver.find_element_by_xpath("//span[@id='priceblock_ourprice']")
        prices.append(price_wbe.text)
    except NoSuchElementException:
        try:
            price_wbe=driver.find_element_by_xpath("//span[@id='priceblock_dealprice']
            prices.append(price_wbe.text+"(Deal price)")
        except NoSuchElementException:
```

```
            try:
                price_wbe=driver.find_element_by_xpath("//span[@class='a-price-whole']
                prices.append(price_wbe.text+"(Prime member price)")
            except NoSuchElementException:
                prices.append('--')


    # get return/exchange data
    try:
        xchange_wbe=driver.find_element_by_xpath("//a[@class='a-size-small a-link-norm
        xchanges.append(xchange_wbe.text)
    except NoSuchElementException:
        xchanges.append('--')


    # get expected delivery webelement
    try:
        xp_del_wbe=driver.find_element_by_xpath("//div[@id='ddmDeliveryMessage']")
        if xp_del_wbe.text!='':
            xp_deliveries.append(xp_del_wbe.text)
        else:
            xp_deliveries.append("Currently not available")
    except NoSuchElementException:
        xp_deliveries.append('--')


    # get product availability
    try:
        prod_avail_wbe=driver.find_element_by_xpath("//div[@id='availability']")
        prod_avail.append(prod_avail_wbe.text)
    except NoSuchElementException:
        prod_avail.append('--')

    # get other details
    try:
        other_details_wbe=driver.find_element_by_xpath("//div[@id='featurebullets_feat
        other_details.append(other_details_wbe.text.replace("\n","."))
    except NoSuchElementException:
        other_details.append('--')

    # get product url
    prod_url.append(driver.current_url)
```

In [ ]:

```
brands=[]
brands.clear()

products=[]
products.clear()

ratings=[]
ratings.clear()

number_ratings=[]
number_ratings.clear()

prices=[]
prices.clear()

xchanges=[]
```

```
xchanges.clear()

xp_deliveries=[]
xp_deliveries.clear()

prod_avail=[]
prod_avail.clear()

other_details=[]
other_details.clear()

prod_url=[]
prod_url.clear()

page_urls=[]
page_urls.clear()
input_prompt=input("Which amazon product do you have in mind ?:")

amzn_search(input_prompt)

#add time
time.sleep(3)

# get web element for webpage number button
page_wbe=driver.find_elements_by_xpath("//ul[@class='a-pagination']//following::a")

# get urls of 1st 3 pages
for i in page_wbe:
    temp_url= i.get_attribute('href')
    page_urls.append(temp_url)
    if len(page_urls)==3:
        break

counter=1
# get product info by iterating over each page
for url in page_urls:
    if counter == 1: # first page check
        time.sleep(2)
        fetch_product() # get webelemets for product titles, then click on each title
        counter=2
    elif counter != 1: # first page scraping complete
        time.sleep(2)
        # open the next webpage in current tab
        driver.get(url)

        # give time to load
        time.sleep(3)

        # fetch data from current webpage
        fetch_product()

        # time
        time.sleep(3)
```

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

In [ ]:
```
amzn_3pg=pd.DataFrame({})
amzn_3pg['Brand']=brands
amzn_3pg['Product']=products
```

```
amzn_3pg['Rating']=ratings
amzn_3pg['Number of ratings']=number_ratings
amzn_3pg['Price']=prices
amzn_3pg['Replacement']=xchanges
amzn_3pg['Expected delivery']=xp_deliveries
amzn_3pg['Product availability']=prod_avail
amzn_3pg['Product description']=other_details
amzn_3pg['Product link']=prod_url
```

In [ ]:
```
pd.set_option('display.max_colwidth',None)
```

In [ ]:
```
pd.set_option('display.max_rows',None)
```

In [ ]:
```
amzn_3pg
```

In [ ]:
```
amzn_3pg.to_csv("guitars_from_Amazon_India.csv",float_format='%.2f',index=False)
```

In [ ]:
```

```

In [ ]:
```
open_url('https://www.google.co.in/imghp?hl=en&ogbl')

time.sleep(5)
# get web element for search button
search_wbe=driver.find_element_by_xpath("//div[@class='pR49Ae gsfi']//following::input
#enter text into search bar
search_wbe.send_keys('fruits',Keys.ENTER)
# give time to load the images page
time.sleep(5)
# get webelement for 1st image
first_image_wbe=driver.find_element_by_xpath("//div[@class='bRMDJf islir']//following:
# get web element for 'more results' buttom
more_btn=driver.find_element_by_xpath("//div[@class='qvfT1']")
# scroll to 'more results' button
for i in range(5):
    driver.execute_script("arguments[0].scrollIntoView(true);",more_btn)
    time.sleep(10)
# get to the top of page
driver.execute_script("arguments[0].scrollIntoView(true);",first_image_wbe)
# initiate list before using it
images=[]
images.clear()
# get webelemnts for images on the page
images = driver.find_elements_by_xpath("//div[@class='bRMDJf islir']//following::img")
# loop for 100 webelements
for i in range(100):
    images[i].screenshot('F:/fruit_pics/fruits_'+str(i)+'.png')
```

In [ ]:
```
open_url('https://www.google.co.in/imghp?hl=en&ogbl')

time.sleep(5)
# get web element for search button
search_wbe=driver.find_element_by_xpath("//div[@class='pR49Ae gsfi']//following::input
#enter text into search bar
search_wbe.send_keys('cars',Keys.ENTER)
# give time to load the images page
time.sleep(5)
# get webelement for 1st image
```

```
first_image_wbe=driver.find_element_by_xpath("//div[@class='bRMDJf islir']//following:
# get web element for 'more results' buttom
more_btn=driver.find_element_by_xpath("//div[@class='qvfT1']")
# scroll to 'more results' button
for i in range(5):
    driver.execute_script("arguments[0].scrollIntoView(true);",more_btn)
    time.sleep(10)
# get to the top of page
driver.execute_script("arguments[0].scrollIntoView(true);",first_image_wbe)
# initiate list before using it
images=[]
images.clear()
# get webelemnts for images on the page
images = driver.find_elements_by_xpath("//div[@class='bRMDJf islir']//following::img")
# loop for 100 webelements
for i in range(100):
    images[i].screenshot('F:/car_pics/car_'+str(i)+'.png')
```

In [ ]:
```
open_url('https://www.google.co.in/imghp?hl=en&ogbl')

time.sleep(5)
# get web element for search button
search_wbe=driver.find_element_by_xpath("//div[@class='pR49Ae gsfi']//following::input
#enter text into search bar
search_wbe.send_keys('Machine Learning',Keys.ENTER)
# give time to load the images page
time.sleep(5)
# get webelement for 1st image
first_image_wbe=driver.find_element_by_xpath("//div[@class='bRMDJf islir']//following:
# get web element for 'more results' buttom
more_btn=driver.find_element_by_xpath("//div[@class='qvfT1']")
# scroll to 'more results' button
for i in range(5):
    driver.execute_script("arguments[0].scrollIntoView(true);",more_btn)
    time.sleep(10)
# get to the top of page
driver.execute_script("arguments[0].scrollIntoView(true);",first_image_wbe)
# initiate list before using it
images=[]
images.clear()
# get webelemnts for images on the page
images = driver.find_elements_by_xpath("//div[@class='bRMDJf islir']//following::img")
# loop for 100 webelements
for i in range(100):
    images[i].screenshot('F:/ml_pics/ml_'+str(i)+'.png')
```

In [ ]:
```
open_url('https://www.google.co.in/imghp?hl=en&ogbl')

time.sleep(5)
# get web element for search button
search_wbe=driver.find_element_by_xpath("//div[@class='pR49Ae gsfi']//following::input
#enter text into search bar
search_wbe.send_keys('Guitar',Keys.ENTER)
# give time to load the images page
time.sleep(5)
# get webelement for 1st image
first_image_wbe=driver.find_element_by_xpath("//div[@class='bRMDJf islir']//following:
# get web element for 'more results' buttom
more_btn=driver.find_element_by_xpath("//div[@class='qvfT1']")
```

```
# scroll to 'more results' button
for i in range(5):
    driver.execute_script("arguments[0].scrollIntoView(true);",more_btn)
    time.sleep(10)
# get to the top of page
driver.execute_script("arguments[0].scrollIntoView(true);",first_image_wbe)
# initiate list before using it
images=[]
images.clear()
# get webelemnts for images on the page
images = driver.find_elements_by_xpath("//div[@class='bRMDJf islir']//following::img")
# loop for 100 webelements
for i in range(100):
    images[i].screenshot('F:/ml_pics/ml_'+str(i)+'.png')
```

In [ ]:
```
open_url('https://www.google.co.in/imghp?hl=en&ogbl')

time.sleep(5)
# get web element for search button
search_wbe=driver.find_element_by_xpath("//div[@class='pR49Ae gsfi']//following::input
#enter text into search bar
search_wbe.send_keys('Cakes',Keys.ENTER)
# give time to load the images page
time.sleep(5)
# get webelement for 1st image
first_image_wbe=driver.find_element_by_xpath("//div[@class='bRMDJf islir']//following:
# get web element for 'more results' buttom
more_btn=driver.find_element_by_xpath("//div[@class='qvfT1']")
# scroll to 'more results' button
for i in range(5):
    driver.execute_script("arguments[0].scrollIntoView(true);",more_btn)
    time.sleep(10)
# get to the top of page
driver.execute_script("arguments[0].scrollIntoView(true);",first_image_wbe)
# initiate list before using it
images=[]
images.clear()
# get webelemnts for images on the page
images = driver.find_elements_by_xpath("//div[@class='bRMDJf islir']//following::img")
# loop for 100 webelements
for i in range(100):
    images[i].screenshot('F:/ml_pics/ml_'+str(i)+'.png')
```

In [ ]:
```

```

In [ ]:
```
def fetch_data():

    # get product title webelement
    title_wbe=driver.find_element_by_xpath("//span[@class='B_NuCI']")
    # append brand name
    brands.append(title_wbe.text.split(" ")[0])
    # append product name
    products.append(title_wbe.text.split(',')[0].replace("(",""))

    # get price webelement
    price_wbe=driver.find_element_by_xpath("//div[@class='_30jeq3 _16Jk6d']")
    # append price
    prices.append(price_wbe.text)
```

```
    # get product url
    urls.append(driver.current_url)

    #get "specifications" text webelement which has all features in text
    features_wbe=driver.find_elements_by_xpath("//div[@class='_3k-BhJ']")
    # get text from all types of 'specifications' in one place
    for wbe in features_wbe:
        features.append(wbe.text.replace("\n","**").replace(",","^^"))

    # looping through all specification categories
    for j in range(0,len(features)):
        # check if specification category is 'General'
        if (features[j].split("**"))[0].lower()=='general':
            general=features[j].split("**")
            general_dict={}
            for i in general:
                if general.index(i)%2!=0:
                    general_dict[i.lower()]=general[general.index(i)+1]
            color.append(general_dict['color'])
        # check if specification category is 'Display Features'
        elif (features[j].split("**"))[0].lower()=='display features':
            display=features[j].split("**")
            display_dict={}
            for i in display:
                if display.index(i)%2!=0:
                    display_dict[i.lower()]=display[display.index(i)+1]
            display_size.append(str(display_dict['display size']))
            display_reso.append(str(display_dict['resolution']))
        # check if specification category is 'OS & Processor Features'
        elif (features[j].split("**"))[0].lower()=='os & processor features':
            processor=features[j].split("**")
            processor_dict={}
            for i in processor:
                if processor.index(i)%2!=0:
                    processor_dict[i.lower()]=processor[processor.index(i)+1]
            if 'processor core' in processor_dict.keys():
                os_processors.append(str(processor_dict['operating system'])+str(proce
            else:
                os_processors.append(str(processor_dict['operating system']))
        # check if specification category is 'Memory & Storage Features'
        elif (features[j].split("**"))[0].lower()=='memory & storage features':
            memory=features[j].split("**")
            memory_dict={}
            for i in memory:
                if memory.index(i)%2!=0:
                    memory_dict[i.lower()]=memory[memory.index(i)+1]
            RAM.append(str(memory_dict['ram']))
            storage_ROM.append(str(memory_dict['internal storage']))
        # check if specification category is 'Camera Features'
        elif (features[j].split("**"))[0].lower()=='camera features':
            cameras=features[j].split("**")
            camera_dict={}
            for i in cameras:
                if cameras.index(i)%2!=0:
                    camera_dict[i.lower()]=cameras[cameras.index(i)+1]
            primary_cameras.append(str(camera_dict['primary camera']))
            if 'secondary camera' in camera_dict.keys():
                secondary_cameras.append(str(camera_dict['secondary camera']))
            else:
                secondary_cameras.append(str("--"))
```

```
        # check if specification category is 'Battery & Power Features'
        elif (features[j].split("**"))[0].lower()=='battery & power features':
            batt=features[j].split("**")
            batt_dict={}
            for i in batt:
                if batt.index(i)%2!=0:
                    batt_dict[i.lower()]=batt[batt.index(i)+1]
            battery.append(str(batt_dict['battery capacity']))

    features.clear()
```

In [ ]:

```
open_url("https://www.flipkart.com")

# get web element for login email search bar
email_wbe=driver.find_element_by_xpath("//input[contains(@class,'_2IX_2-')]")#@class='
# enter the email id
email_wbe.send_keys("8208507760")

#get web element for password bar
pwd_wbe=driver.find_element_by_xpath("//input[@type='password']")#class='_2IX_2- _3mct
# enter the password
pwd_wbe.send_keys("Newpassword@72")

# get the webelement for login button
login_btn=driver.find_element_by_xpath("//button[@class='_2KpZ6l _2HKlqd _3AWRsL']")
#click on login button
login_btn.click()

#add timer sleep
time.sleep(3)

# get the web element for search bar
search_wbe=driver.find_element_by_xpath("//input[@name='q' and @placeholder='Search fo
# write into search bar and pressing ENTER
search_wbe.send_keys("Nokia 7.1 plus",Keys.ENTER)

#add timer sleep
time.sleep(3)


# get webelements for each result tile
result_wbe=driver.find_elements_by_xpath("//div[@class='_2kHMtA']//following::a")

urls=[]
urls.clear()
brands=[]
brands.clear()
products=[]
products.clear()
features=[]
features.clear()
prices=[]
prices.clear()
color=[]
color.clear()
display_size=[]
display_size.clear()
display_reso=[]
display_reso.clear()
```

```
os_processors=[]
os_processors.clear()
RAM=[]
RAM.clear()
storage_ROM=[]
storage_ROM.clear()
primary_cameras=[]
primary_cameras.clear()
secondary_cameras=[]
secondary_cameras.clear()
battery=[]
battery.clear()


for product_tile in result_wbe:
    #click on the product_tile one by one
    product_tile.click()

    #add time
    time.sleep(2)

    #check if new tab has opened
    if len(driver.window_handles)>1:
        #switch driver to newopened tab
        driver.switch_to.window(driver.window_handles[1])
    else:
        break

    #click on the 'Read More' button
    read_more_btn=driver.find_element_by_xpath("//button[@class='_2KpZ6l _1FH0tX']")
    read_more_btn.click()

    # add time
    time.sleep(2)

    fetch_data()

    # close the newly opened tab
    driver.close()

    # switch to main search page
    driver.switch_to.window(driver.window_handles[0])

    # add time
    time.sleep(2)
```

In [ ]:

```
temp_df=pd.DataFrame({})
temp_df['Brand']=brands
temp_df['Product']=products
temp_df['Price']=prices
temp_df['Color']=color
temp_df['Display size']=display_size
temp_df['Resolution']=display_reso
temp_df['Operating system & processors']=os_processors
temp_df['RAM']=RAM
temp_df['Storage']=storage_ROM
temp_df['Primary Camera']=primary_cameras
temp_df['Secondary camera']=secondary_cameras
temp_df['Battery']=battery
```

In [ ]:
```
temp_df
```

In [ ]:
```
temp_df.to_csv("Nokia 7.1 plus models.csv",float_format='%.2f',index=False)
```

In [ ]:
```
pd.read_csv('Nokia 7.1 plus models.csv')
```

In [ ]:
```

```

In [ ]:
```
prompt=input("Enter a location: ")

open_url('https://www.google.com/maps/')

# get search bar webelement
search_bar=driver.find_element_by_xpath('//*[@id="searchboxinput"]')
# send city name
search_bar.send_keys(prompt,Keys.ENTER)
time.sleep(10)
current_url=driver.current_url
longitude=current_url.split('@')[1].split(',')[0]
lattitude=current_url.split('@')[1].split(',')[1]
print(prompt," Longitude: ",longitude,'"N')
print(prompt," Lattitude: ",lattitude,'"E')
```

In [ ]:
```

```

In [ ]:
```
open_url('https://www.digit.in')

# click on 'Best gaming laptops' link
driver.find_element_by_xpath('/html/body/div[3]/div/div[2]/div[2]/div[4]/ul/li[9]/a').

time.sleep(3)

# get laptop names
names=[]
names.clear()

names_wbe=driver.find_elements_by_xpath("//h3")

for i in names_wbe:
    names.append(i.text)

time.sleep(2)

# get specifications box weblements
spec_wbe=driver.find_elements_by_xpath("//div[@class='Spcs-details']")

specs=[]
specs.clear()

for wbe in spec_wbe:
    specs.append(wbe.text)
```

In [ ]:
```
names
```

In [ ]:
```
specs
```

In [ ]:
```
specs[0]
```

In [ ]:
```
temp_specs=[]
for specs in specs:
    temp_specs.append(specs.split('\n'))
temp_specs[0]
```

In [ ]:
```
temp_specs
```

In [ ]:
```
temp_specs[0][3][0:3]
```

In [ ]:
```
os=[]
dis=[]
processor=[]
memory=[]
weight=[]
dimension=[]
graphic=[]
price=[]
```

In [ ]:
```
for j in temp_specs:
    if len(j)==8:
        for i in j:
            if i[0]=='O':
                os.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0:3]=='Dis':
                dis.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0:3]=='Pro':
                processor.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0]=='M':
                memory.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0]=='W':
                weight.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0:3]=='Dim':
                dimension.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0]=='G':
                graphic.append(i.split(':')[-1])
                time.sleep(2)
                price.append('--')
                continue
    elif len(j)==9:
        for i in j:
            if i[0]=='O':
                os.append(i.split(':')[-1])
                time.sleep(2)
```

```
                continue
            if i[0:3]=='Dis':
                dis.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0:3]=='Pro':
                processor.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0]=='M':
                memory.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0]=='W':
                weight.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0:3]=='Dim':
                dimension.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0]=='G':
                graphic.append(i.split(':')[-1])
                time.sleep(2)
                continue
            if i[0:3]=='Pri':
                price.append(i.split(':')[-1])
                time.sleep(2)
                continue
```

In [ ]:
```
df_digit=pd.DataFrame({})
df_digit['Model Name']=names
df_digit['Operating System']=os
df_digit['Display']=dis
df_digit['Processor']=processor
df_digit['Memory']=memory
df_digit['Weight']=weight
df_digit['Dimension']=dimension
df_digit['Graphics Processor']=graphic
df_digit['Price']=price
```

In [ ]:
```
df_digit
```

In [ ]:
```

```

In [ ]:
```
# connecting to the webdriver
driver=webdriver.Chrome(r"C:/Users/Lenovo/Downloads/chromedriver_win32 (1)/chromedrive
```

In [ ]:
```
# getting the specified url
url = "https://www.forbes.com/?sh=41bd46d2254c"
driver.get(url)
```

In [ ]:
```
#let's get option button from the page
opt_btn = driver.find_element_by_xpath("//div[@class='header__left']//button")
opt_btn.click()
time.sleep(3)

#select billionaires from options
```

```
blns = driver.find_element_by_xpath("/html/body/div[1]/header/nav/div[3]/ul/li[1]")
blns.click()
time.sleep(3)
#select world billionaire
bln_list = driver.find_element_by_xpath("/html/body/div[1]/header/nav/div[3]/ul/li[1]/
bln_list.click()
time.sleep(4)
```

In [ ]:

```
# scraping required data from the web page
# creating empty lists
Rank = []
Person_Name = []
Net_worth = []
Age = []
Citizenship = []
Source = []
Industry = []


while(True):

    # scraping the data of rank of the billionaires
    rank_tag = driver.find_elements_by_xpath("//div[@class='rank']")
    for rank in rank_tag:
        Rank.append(rank.text)
    time.sleep(1)



    # scraping the data  of names of the billionaires
    name_tag = driver.find_elements_by_xpath("//div[@class='personName']/div")
    for name in name_tag:
        Person_Name.append(name.text)
    time.sleep(1)


    # scraping the data of age of the billionaires
    age_tag = driver.find_elements_by_xpath("//div[@class='age']/div")
    for age in age_tag:
        Age.append(age.text)
    time.sleep(1)


    # scraping the data of citizenship of the billionaires
    cit_tag = driver.find_elements_by_xpath("//div[@class='countryOfCitizenship']")
    for cit in cit_tag:
        Citizenship.append(cit.text)
    time.sleep(1)


    # scraping the data of source of income of the billionaires
    sour_tag = driver.find_elements_by_xpath("//div[@class='source']")
    for sour in sour_tag:
        Source.append(sour.text)
    time.sleep(1)


    # scraping data of industry of the billionaires
    ind_tag = driver.find_elements_by_xpath("//div[@class='category']//div")
```

```
    for ind in ind_tag:
        Industry.append(ind.text)
    time.sleep(1)


    # scraping data of net_worth of billionaires
    net_tag = driver.find_elements_by_xpath("//div[@class='netWorth']/div")
    for net in net_tag:
        Net_worth.append(net.text)
    time.sleep(1)


    # clicking on next button
    try:
        next_button = driver.find_element_by_xpath("//button[@class='pagination-btn pa
        next_button.click()
    except:
        break
```

In [ ]:
```
print(len(Rank),
len(Person_Name),
len(Net_worth),
len(Age),
len(Citizenship),
len(Source),
len(Industry))
```

In [ ]:
```
# framing Data
Billionaires = pd.DataFrame({})
Billionaires['Rank'] = Rank
Billionaires['Name'] = Person_Name
Billionaires['Net Worth'] = Net_worth
Billionaires['Age'] = Age
Billionaires['Citizenship'] = Citizenship
Billionaires['Source'] = Source
Billionaires['Industry'] = Industry
Billionaires
```

In [ ]:
```
# saving dataset in csv
Billionaires.to_csv('Forbes_Billionaires.csv')
```

In [ ]:
```
driver.close()
```

In [ ]:

In [ ]:
```
# connecting to the webdriver
driver=webdriver.Chrome(r"C:/Users/Lenovo/Downloads/chromedriver_win32 (1)/chromedrive
```

In [ ]:
```
# opening the youtube.com
url = "https://www.youtube.com/"
driver.get(url)
time.sleep(2)
```

In [ ]:
```
# finding element for search bar
search_bar = driver.find_element_by_xpath("//div[@class='ytd-searchbox-spt']/input")
```

```
search_bar.send_keys("GOT")      # entering video name
time.sleep(2)
```

In [ ]:
```
#clicking on search button
search_btn = driver.find_element_by_id("search-icon-legacy")
search_btn.click()
time.sleep(2)
```

In [ ]:
```
# clicking on first video
video = driver.find_element_by_xpath("//yt-formatted-string[@class='style-scope ytd-vi
video.click()
```

In [ ]:
```
# 1000 times we scroll down by 10000 in order to generate more comments
for _ in range(1000):
    driver.execute_script("window.scrollBy(0,10000)")
```

In [ ]:
```
# creating empty lists
comments = []
comment_time = []
Time = []
Likes = []
No_of_Likes = []

# scrape comments
cm = driver.find_elements_by_id("content-text")
for i in cm:
    if i.text is None:
        comments.append("--")
    else:
        comments.append(i.text)
time.sleep(4)


# scrape time when comment was posted
tm = driver.find_elements_by_xpath("//a[contains(text(),'ago')]")
for i in tm:
    Time.append(i.text)

for i in range(0,len(Time),2):
    comment_time.append(Time[i])
time.sleep(4)


# scrape the comment likes
like = driver.find_elements_by_xpath("//span[@class='style-scope ytd-comment-action-bu
for i in like:
    Likes.append(i.text)

for i in range(1,len(Likes),2):
    No_of_Likes.append(Likes[i])
```

In [ ]:
```
print(len(comments),len(comment_time),len(No_of_Likes))
```

In [ ]:
```
# creating dataframe for scraped data

Youtube = pd.DataFrame({})
Youtube['Comment'] = comments[:500]
Youtube['Comment Time'] = comment_time[:500]
```

```
Youtube['Comment Upvotes'] = No_of_Likes[:500]
Youtube
```

In [ ]:
```
#saving the dataframe to csv
Youtube.to_csv("Youtube GOT Comments.csv")
```

In [ ]:
```
driver.close()
```

In [ ]:
```

```

In [ ]:
```
# connecting to the webdriver
driver=webdriver.Chrome(r"C:/Users/Lenovo/Downloads/chromedriver_win32 (1)/chromedrive
```

In [ ]:
```
# getting the web page of mentioned url
url = "https://www.hostelworld.com/"
driver.get(url)
time.sleep(3)
```

In [ ]:
```
# locating the location search bar
search_bar = driver.find_element_by_id("search-input-field")

# entering London in search bar
search_bar.send_keys("London")
```

In [ ]:
```
# select London
London = driver.find_element_by_xpath("//ul[@id='predicted-search-results']//li[2]")
#clicking on button
London.click()

# do click on Let's Go button
search_btn = driver.find_element_by_id('search-button')
search_btn.click()
```

In [ ]:
```
# creating empty list & find required data
hostel_name = []
distance = []
pvt_prices = []
dorms_price = []
rating = []
reviews = []
over_all = []
facilities = []
description = []
url = []
```

In [ ]:
```
# scraping the required informations
for i in driver.find_elements_by_xpath("//div[@class='pagination-item pagination-curre
    i.click()
    time.sleep(3)


    # scraping  hostel name
    try:
        name = driver.find_elements_by_xpath("//h2[@class='title title-6']")
        for i in name:
            hostel_name.append(i.text)
```

```
    except NoSuchElementException:
        hostel_name.append('-')


    # scraping distance from city centre
    try:
        dist = driver.find_elements_by_xpath("//div[@class='subtitle body-3']//a//spar
        for i in name:
            distance.append(i.text.replace('Hostel - ',''))
    except NoSuchElementException:
        distance.append('-')


    for i in driver.find_elements_by_xpath("//div[@class='prices-col']"):
    # scraping privates from price
        try:
            pvt_price = driver.find_element_by_xpath("//a[@class='prices']//div[1]//di
            pvt_prices.append(pvt_price.text)
        except NoSuchElementException:
            pvt_prices.append('-')


    for i in driver.find_elements_by_xpath("//div[@class='prices-col']"):
    # scraping dorms from price
        try:
            dorms = driver.find_element_by_xpath("//a[@class='prices']//div[2]/div")
            dorms_price.append(dorms.text)
        except NoSuchElementException:
            dorms_price.append('-')


    # scraping facilities
    try:
        fac1 = driver.find_elements_by_xpath("//div[@class='has-wifi']")
        fac2 = driver.find_elements_by_xpath("//div[@class='has-sanitation']")
        for i in fac1:
            for j in fac2:
                facilities.append(i.text +', '+ j.text)
    except NoSuchElementException:
        facilities.append('-')


    #fetching url of each hostel
    p_url = driver.find_elements_by_xpath("//div[@class='prices-col']//a[2]")
    for i in p_url:
        url.append(i.get_attribute("href"))

for i in url:
    driver.get(i)
    time.sleep(3)


    # scraping ratings
    try:
        rat = driver.find_element_by_xpath("//div[@class='score orange big' or @class=
        rating.append(rat.text)
    except NoSuchElementException:
        rating.append('-')
```

```
    # scraping total review
    try:
        rws = driver.find_element_by_xpath("//div[@class='reviews']")
        reviews.append(rws.text.replace('Total Reviews',''))
    except NoSuchElementException:
        reviews.append('-')


    # fetching over all review
    try:
        overall = driver.find_element_by_xpath("//div[@class='keyword']//span")
        over_all.append(overall.text)
    except NoSuchElementException:
        over_all.append('-')


    # fetching property description
    try:
        disc = driver.find_element_by_xpath("//div[@class='content']")
        description.append(disc.text)
    except NoSuchElementException:
        over_all.append('-')

    # do click on show more button for description
    try:
        driver.find_element_by_xpath("//a[@class='toggle-content']").click()
        time.sleep(4)
    except NoSuchElementException:
        pass
```

In [ ]:
```
print(len(hostel_name),
len(distance),
len(pvt_prices),
len(dorms_price),
len(rating),
len(reviews),
len(over_all),
len(facilities),
len(description),
len(url))
```

In [ ]:
```
# creating DataFrame
Hostel = pd.DataFrame({})
Hostel['Hostel Name'] = hostel_name
Hostel['Distance from City Centre'] = distance
Hostel['Ratings'] = rating
Hostel['Total Reviews'] = reviews
Hostel['Overall Reviews'] = over_all
Hostel['Privates from Price'] = pvt_prices
Hostel['Dorms from Price'] = dorms_price
Hostel['Facilities'] = facilities[:74]
Hostel['Description'] = description
Hostel
```

In [ ]:
```
# saving the dataset to csv
Hostel.to_csv("London_Hostels.csv")
```

In [ ]:
```
driver.close()
```