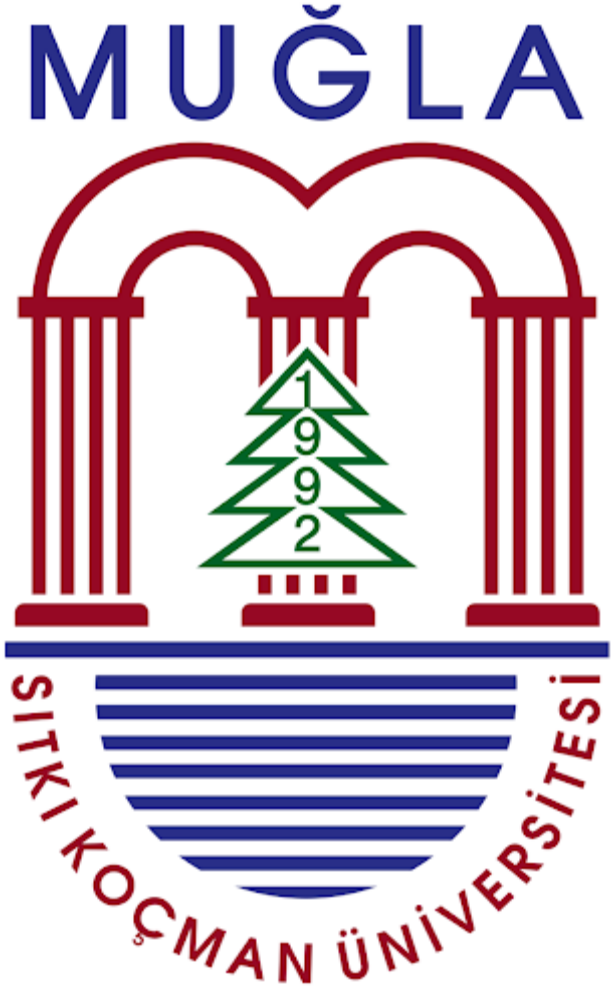
`



COMPUTER ENGINEERING DEPARTMENT

Web Data Mining

Final Homework: LinkedIn Scraper

Hasibullah Mahmood - 160709073

`

1. Introduction

In this Homework, We developed a scraper bot that establishes a session with LinkedIn Website. Then, it searches for a query on Google. After that, it fetches all LinkedIn profiles URLs from the Google page. Then, it scrapes each profile on LinkedIn and fetches his name, address, and contact information. Finally, the script saves the gathered data on the CSV file.

Used Python Packages:

- selenium
- selenium.webdriver
- parsel
- pandas
- time

Teams Tasks:

Ahmed H. Ibrahim	Hasibullah Mahmood
• Establishing autonomous Session with LinkedIn	• Querying Google and Scarping profile URLs from it.
• Scarping LinkedIn Profiles data.	• Manages data storage process and Saving Data to CSV
• + Creating Parameters file with many Variables.	• + Creating Person Class.

2. Names of the files with a short explanation

seleniumScraper.py – the main script that establishes a LinkedIn session, queries Google, scraps Profiles, saves data to CSV file.

Person.py – A class that is imported by seleniumScarper.py to create an object for each profile.

Parameters.py – A python file that has many variables that are imported by seleniumScraper.py

`

3. Methods (with their signature) in seleniumScraper.py:

a) `establishLinkedInSession(in_email,in_password)` **--- it takes LinkedIn email and password as arguments. Then, it automates the login process with LinkedIn.**

```
def establishLinkedInSession(in_email,in_password):

    #### start 1. Making a new login session with linkedin
    # Login to LinkedIn
    driver.get(parameters.linkedIn_login_page)

    # Find Email Field and send email info.
    email = driver.find_element_by_id('username')
    email.send_keys(in_email)

    time.sleep(3)

    # Find password Field and send password info.
    password = driver.find_element_by_id('password')
    password.send_keys(in_password)

    time.sleep(3)

    # Find Submit Button and Click it.
    logInBtn = driver.find_element_by_xpath('//*[@type="submit"]')
    logInBtn.click()

    time.sleep(5)
```

b) `queryGoogle(query,numOfPages)` **--- it takes the query string and numOfResults as arguments. numOfResults = number of profiles, you want to scrap. Then, it fetches all URLs from google to an array.**

```
def queryGoogle(query,numOfResults):
    driver.get(parameters.googleUrl)

    time.sleep(5)

    # Find Search input field and send search info to it.
    searchingField=driver.find_element_by_name('q')
    searchingField.send_keys(query)

    time.sleep(5)
```

`

```
    searchingField.send_keys(Keys.RETURN)
    time.sleep(5)

    driver.get(driver.current_url + numOfResults)

    # Get Profiles Urls from Google querying result
    for rapper in driver.find_elements_by_class_name('r'):
        for a in rapper.find_elements_by_xpath('.//a'):
            search_urls_holder.append(a.get_attribute('href'))

    print(search_urls_holder)
    time.sleep(2)
```

c) `getProfileInfo(profile_url)` **--- This function will be called each time to scrape each profile data. It will scrape the name, address, and Contact Information. Contact Information may include 'Twitter Url', 'Email Address', 'Blog Url', and many other links. It will create an object for this profile and saves profile data in it. Also, it appends the cleaned data to many arrays that will be group in a CSV data frame later.**

```
def getProfileInfo(profile_url):
    # Session with linkedin profile
    html_page=driver.get(profile_url)

    time.sleep(3)

    # Get page Source
    selector = Selector(text=driver.page_source)

    # Get person name
    name = selector.xpath(".//ul[contains(@class,'pv-top-card--
list')]/li[contains(@class,'t-24')]/text()").get()
    # Get person address
    address = selector.xpath(".//ul[contains(@class,'pv-top-card--
list')]/li[contains(@class,'t-16')]/text()").get()

    time.sleep(4)

    # Find contact info Button and Click it.
    contactInfoBtn = driver.find_element_by_xpath('//*[@data-control-
name="contact_see_more"]')
    contactInfoBtn.click()

    time.sleep(4)
```

`

```
    # Update the selector to fetch contact information
    selector = Selector(text=driver.page_source)

    contact_links = selector.css('.pv-contact-info__contact-
link *::attr(href)').getall()

    # Outputting
    p_name = name.strip()
    p_address = address.strip()

    p1 = Person(profile_url, p_name, p_address, contact_links)

    # Adding Person info to csv dataframe arrays
    profile_urls_holder.append(profile_url)
    names_holder.append(p1.name)
    addresses_holder.append(p1.address)
    contact_info_holder.append(p1.listToString())
```

d) `saveProfilesData(csvFile,urls,names,addresses,contact_infos):` **--- it takes csvFile path, URLs array, names array, addresses array, contact_infos array as arguments. Then, it will create a data frame that groups all of the arrays. Then, it will write the data frame to the CSV file.**

```
def saveProfilesData(csvFile,urls,names,addresses,contact_infos):
    # Writing dataframe that combines arrays to csv file
    # dictionary of lists
    dict = {'Url': urls, 'Name': names, 'Address': addresses, 'Contact Info':
contact_infos}

    df = pd.DataFrame(dict)

    # saving the dataframe
    df.to_csv(csvFile, index=False, encoding='utf-8-sig')
```

`

4. Person Class in Person.py:

```
class Person:
    def __init__(self, url, name, address, contactInfo):
        self.url = url
        self.name = name
        self.address = address
        self.contactInfo = contactInfo

    def listToString(self):
        # initialize an empty contact Info links String holder.
        links = ""

        # Adding Links to the String.
        for link in self.contactInfo:
            links = links + link + " | "

            # return string
        return links
```

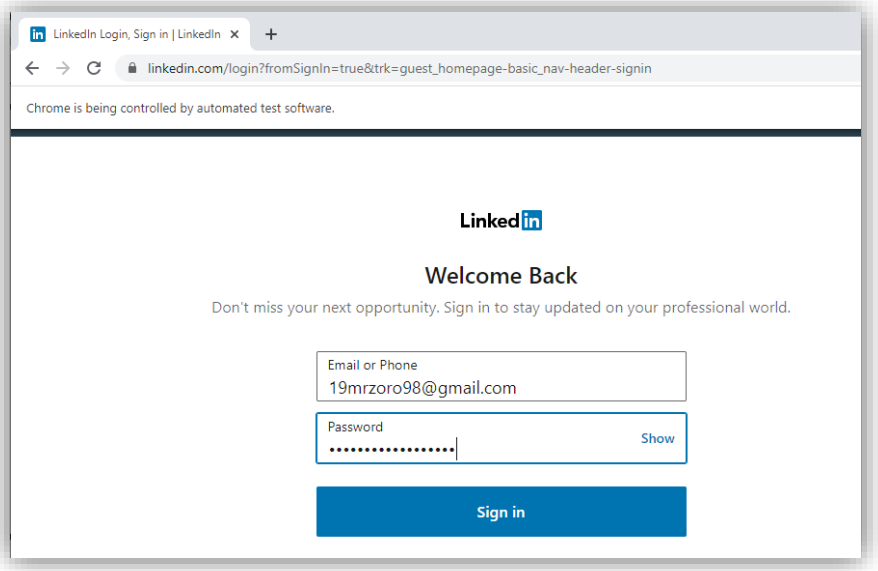
5. **Variables in parameters.py:**

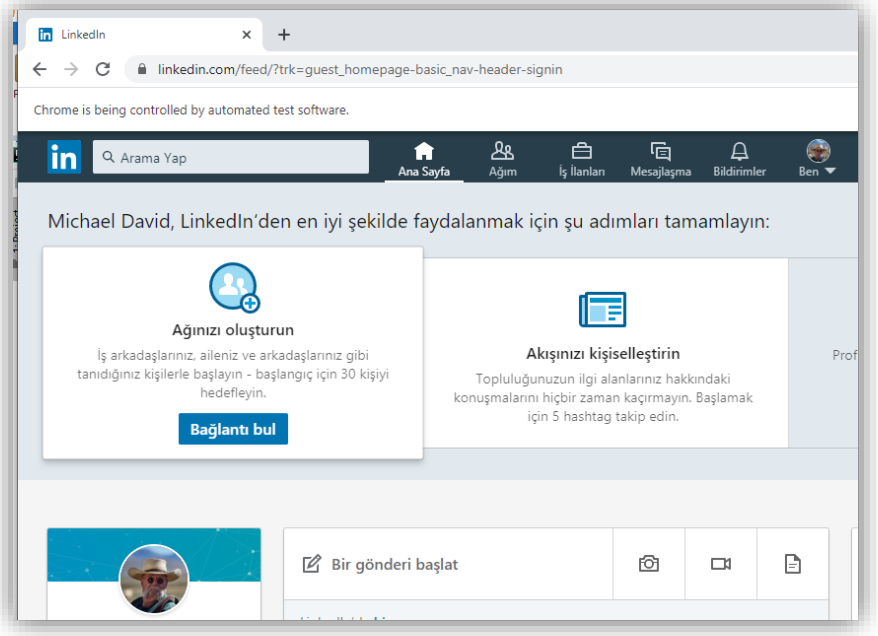
```
chromeWebDriverPath = 'D:/Programs/chromedriver'

linkedIn_login_page = 'https://www.linkedin.com/login?fromSignIn=true&trk=gue
st_homepage-basic_nav-header-signin'

googleUrl = 'https://www.google.com'

# login details
in_email = 'example@gmail.com'
in_password = 'linkedInPassword'

# Search details
google_search_query = 'site:linkedin.com/in/ AND "student" AND "Izmir"'

# CSV file name
csv_file_name='LinkedIn_Profiles_Info.csv'
```

`

6. sample output/screenshots:

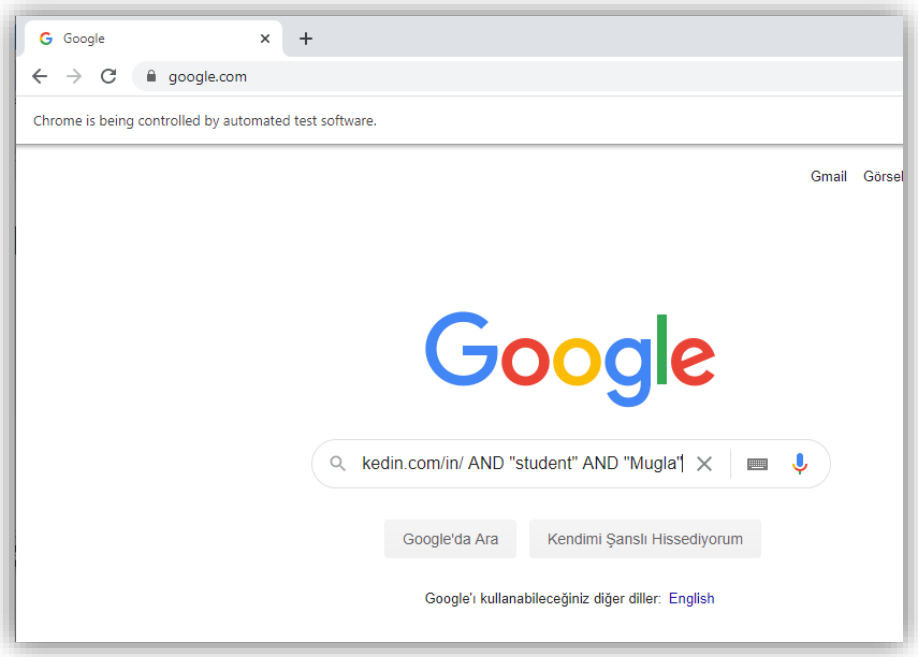
a) Automatic Login

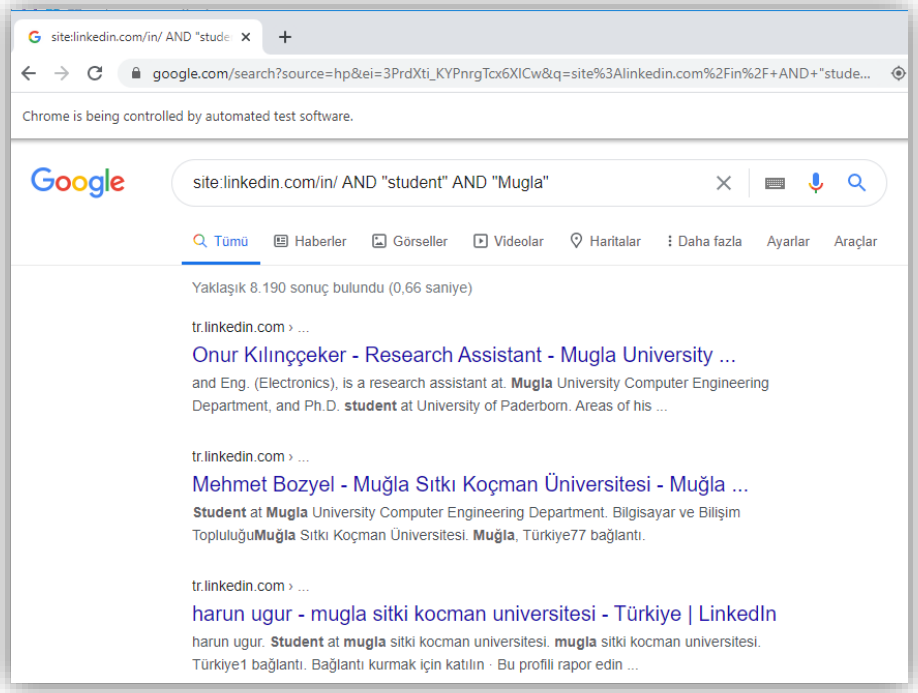




`

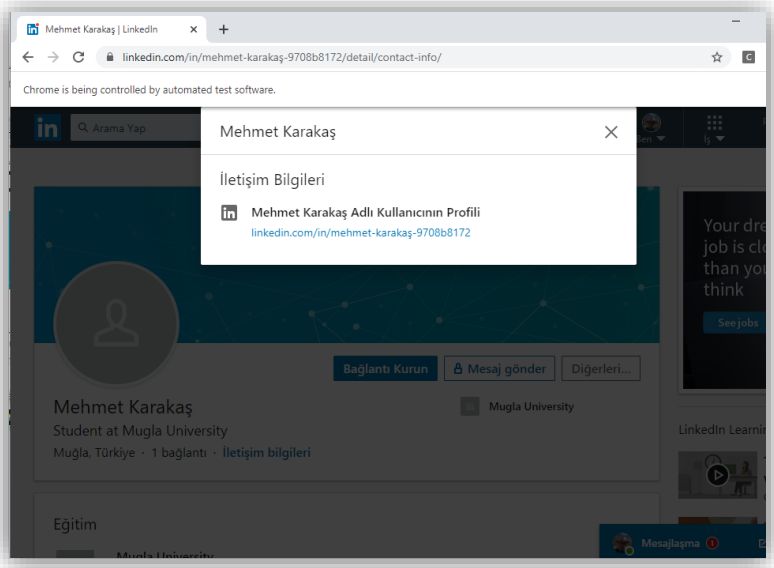
b) Automatic Querying Google

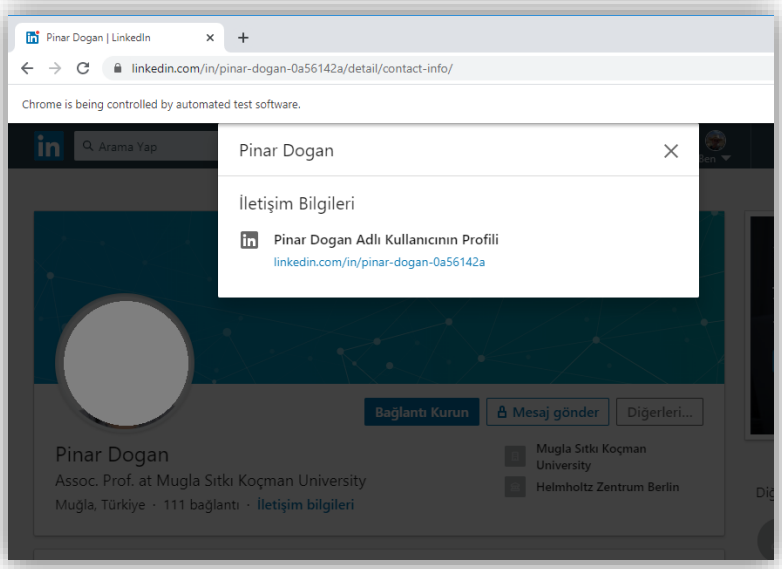




`

c) Scraping Data from gathered accounts URLs





d) Storing Gathered Data in CSV file

	Url	Name	Address	Contact Info
2	https://tr.linkedin.com/in/pinar-dogan-0a56142a	Pinar Dogan	Muğla, Türkiye	https://www.linkedin.com/in/pinar-dogan-0a56142a |
3	https://tr.linkedin.com/in/mehmet-karaka%C5%9F-9708b8172?trk=	Mehmet Karakaş	Muğla, Türkiye	https://www.linkedin.com/in/mehmet-karakaş-9708b8172 |
4	https://tr.linkedin.com/in/ayse-g%C3%BCnay-b3202317	AYSE GÜNAY	Muğla, Türkiye	https://www.linkedin.com/in/ayse-günay-b3202317 |
5	https://tr.linkedin.com/in/gizem-%C3%A7ak%C4%B1r-533a4534?tr	Gizem Çakır	Türkiye	https://www.linkedin.com/in/gizem-çakır-533a4534 |
6	https://www.linkedin.com/in/salih-kaplan-563460173	Salih Kaplan	Muğla, Türkiye	
7	https://tr.linkedin.com/in/esadiye-pekdemir-6084ab182	Esadiye Pekdemir	Muğla, Türkiye	https://www.linkedin.com/in/esadiye-pekdemir-6084ab182 |
8	https://tr.linkedin.com/in/vakkas-celik-a9869819	vakkas celik	Türkiye	https://www.linkedin.com/in/vakkas-celik-a9869819 |
9	https://tr.linkedin.com/in/faruk-d%C3%B6nmez-191991196	Faruk Dönmez	İzmir, Türkiye	https://www.linkedin.com/in/faruk-dönmez-191991196 |
10	https://tr.linkedin.com/in/zekisaglam	Zeki Sağlam	Muğla, Türkiye	https://www.linkedin.com/in/zekisaglam |