ARBISOFT ASSIGNMENT

Scrapping Universities Rankings Using Python

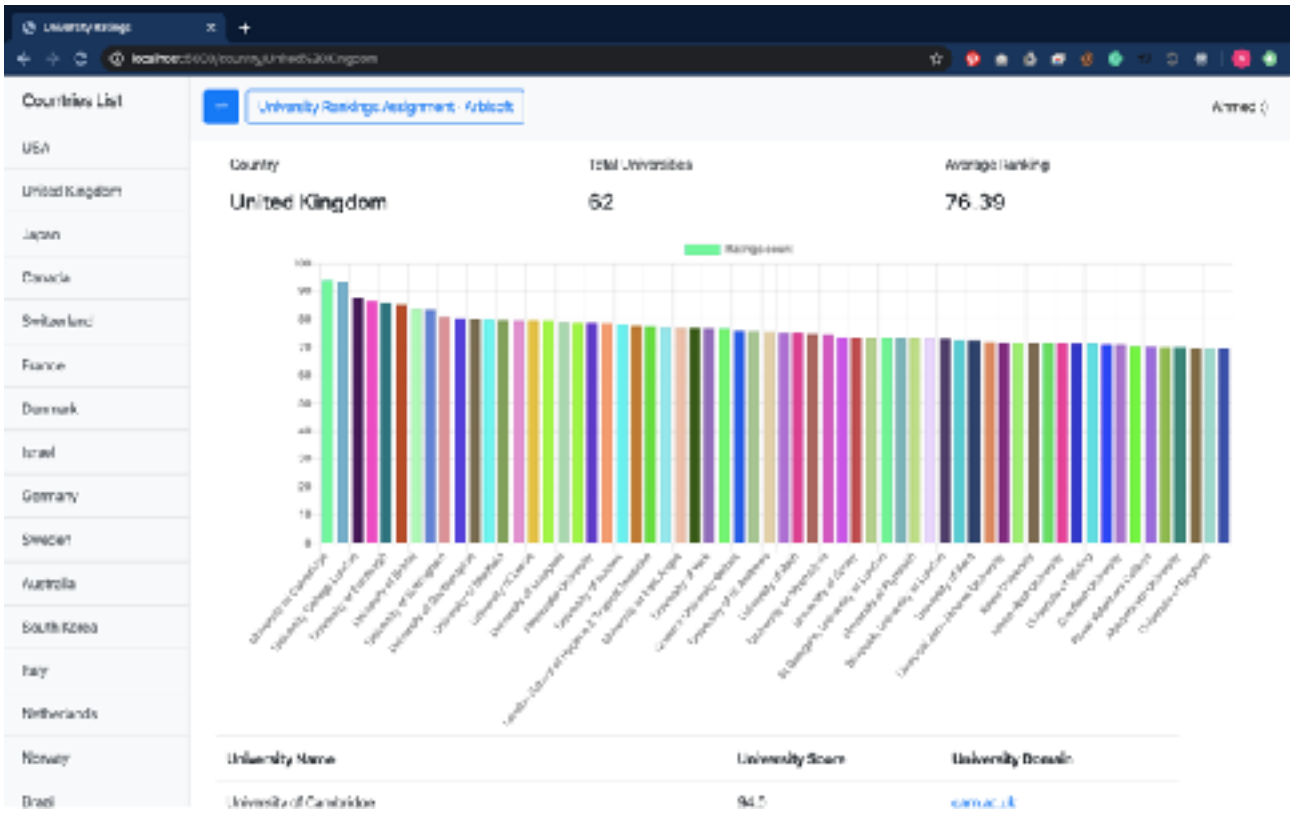
Output Screenshots:

CMD/TERMINAL:

```
/Users/ahmed/Pycharm/arbisoft/modules/bin/python /Users/ahmed/Pycharm/arbisoft/run.py
['USA' 'United Kingdom' 'Japan' 'Canada' 'Switzerland' 'France' 'Denmark'
 'Israel' 'Germany' 'Sweden' 'Australia' 'South Korea' 'Italy'
 'Netherlands' 'Norway' 'Brazil' 'Spain' 'China' 'Finland' 'Singapore'
 'Belgium' 'Taiwan' 'Russia' 'Hong Kong' 'Austria' 'Czech Republic'
 'Portugal' 'South Africa' 'Ireland' 'New Zealand' 'Mexico' 'Poland'
 'Argentina' 'Greece' 'Saudi Arabia' 'Chile' 'Serbia' 'Slovenia' 'Iran'
 'Hungary' 'India' 'Malaysia' 'Egypt' 'Croatia' 'Thailand' 'Iceland'
 'Estonia' 'Turkey' 'Slovak Republic' 'Uruguay' 'Lithuania' 'Colombia'
 'Uganda' 'Lebanon' 'Pakistan' 'Romania' 'Bulgaria' 'Cyprus' 'Tunisia'
 'Nigeria' 'Macau']
Here is the list of countries we have rankings for.

 Type the country name (without quotes):
Enter country name : Pakistan
Country:  Pakistan
Total Universities:  3
Average Ranking:  70.39999999999999
                          Institution Name  Overall Score          Domain
                     Quaid-i-Azam University           71.1      qau.edu.pk
 COMSATS Institute of Information Technology           70.3  comsats.edu.pk
                       Aga Khan University           69.8         aku.edu
```

WEB SERVER:



Technology stack:

This project is made in python using some basic libraries. Let's discuss which libraries are used any why.

1. **urllib;** This library is used to fetch the HTML pages from the website. The reason to use this library is its speed and flexibility. The HTML is easy to parse and manipulate with this library. The question arises why I have not used Selinium for this task. There are two reasons. First, it can not be deployed on the web server. Second, its huge; designed for browser automation. Which is not the case here. So a simple library which can get data from internet is enough for our use.
2. **CSV instead of JSON;** I dropped the idea of json because when we are using raw data and we want to manipulate it, json is not the best choice. We can use json where we have to transmit the data for example, to an API, but working with data is far better and easy with the use of CSV. I used pandas which provides great methods to play with data using dataframes.
3. **Flask Server;** For the graphical representation of this data, I had the option to use Jupiter Notebooks but that is not efficient when we have to generate a lot of graphics. Firing up a flask server with the data files in the background is very light-weight approach and HTML gives us the flexibility to generate any kind of graphs.

Hours it took to complete the task.

I worked in shifts and collectively it took around 24-28 hours to complete this task.

Improvements?

There is always room for improvements. If I had more time, I would schedule background tasks so the data could be updated without having to manually scrape it from the website.
Another thing which could be done is to convert the HTML into react app which would look more appealing.