Three separate projects have been implemented in this submission. The main theme of all the projects was to use selenium and beautifulsoup libraries for python to automate website actions like clicks and data extraction.

The first project is a python script which uses selenium to automatically attend all online classes in CodeTantra.

1,CodeTantra Automatic Login

Since I had lost a lot of marks in the previous semester for failing to attend many of my online classes, I have decided to make a python script which runs in the background and automatically joins all online classes on codetantra using the selenium chrome webdriver.

The program uses the schedule library for python to run the script at particular times on any given day. The user can change the days and the times at which the script is run in order to use the script according to their online class timetable.

Implementation:

First, The user has to enter their codetantra username and password. The chrome webdriver is initialized and it opens the login page for codetantra. It finds the username and password fields by searching for the relevant HTML element by its id and it enters the user's login details and presses the submit button.

An implicit wait is used in order to make the driver wait for the new webpage to load. A try, except code block has been implemented in order to check whether the login was successful or not. The driver now opens the 'View Classes/Meetings' link by searching for the corresponding HTML element using its XPATH.

The codetantra timetable opens up and there are multiple links each corresponding to a different class on that day. A list of all the class links has been created by finding all the relevant HTML elements using their class names.

Using a for loop, each link is opened and the driver searches for the 'join' button using its XPATH. Using a try except block, if the meeting has not been started yet, the script handles the exception and moves on to the next link. Once the 'join' button has been found on a link, the driver clicks on the link and joins the meeting.

There is a further option to join audio using microphone or listen only, however this does not matter for attendance, hence the script stops here.

Using schedule library, the starting times for the classes for CSE semester 3 have been hardcoded. This can be changed to fit the user's timetable.

Since the user should not have to run the script again everyday, the script can be run in the background by using nohup

Since internet access is needed to connect to the meetings, ensure that the computer stays awake during the start times of the meetings. This can be done in the Power and Battery settings of the computer.

Installation And Usage:

Selenium chrome driver is needed for this script. It can be downloaded at https://chromedriver.chromium.org/downloads

Move the chromedriver executable to the project folder.

In order to run the program, use nohup python auto_login.py &

Since nohup is used to run the script, it runs in the background and inputs cant be taken. Hence the username and password have to be changed in the script directly before running it.

In order to install schedule library, visit https://pypi.org/project/schedule/
Read documentation here : https://schedule.readthedocs.io/en/stable/

Future Of The Project:

Will try to make it more generalized, such as providing input for username and password, and let the user input their timetable instead of hardcoding it.

Will also look into automatically participating in polls and sending chat messages like 'Good Morning' at the beginning of the session and 'Thank You' at the end of the session in order to increase participation index.

2.CodeTantra Grades Scraper

IMPLEMENTATION

This python script makes use of selenium to login to codetantra and search for exams conducted between a particular start and end date inputted by the user, and it collects details such as date, duration, Exam Name and it opens the result page for each exam and scrapes the Marks Obtained. It then uses the pandas module for python to convert the information obtained into an excel sheet.

The data can be further crunched by using graphs, charts and percentages to track progress.

The script takes username, password, start_date and end_date as inputs. It aims to search for all exams which took place between start_date and end_date and collect data.

The chrome selenium webdriver logs in to codetantra by finding the login HTML elements and using sendkeys() to automatically enter the user's login data into the text boxes and then it searches for the submit button by its HTML id and clicks it.

The user inputs the date in DD/MM/YYYY format, however the html elements in codetantra use only 1 digit for 0 – 9. For example, the user enters 09 as the date however the codetantra html element corresponds to 9. Hence, the date entered by the user is spliced and each field is converted to an integer and back to a string in order to match the codetantra format.

Apon opening https://iiitb.codetantra.com/secure/home/tests.jsp , there is a dynamic calendar in which the month and year can be selected using a drop down list and the date can be selected as a button.

Apon inspecting the page and finding the relevant HTML XPATH for each element, the search date and end date have been entered.

Since the Calendar is dynamic, sometimes the driver might access an element before it has been loaded, leading to an exception. In order to avoid this, implicit waits have been implemented after some clicks. Implicit wait is used to wait for the elements to load completely before being accessed by the driver.

Since CodeTantra's element class names change dynamically, depending on which month and date has been selected, there are a lot of try except blocks to handle the exceptions which arise like ElementnotFound or ElementNotInteractable.

String Manipulation has been used in order to dynamically find the required HTML tags based on the user's start and end date input.

Once the start and end date has been set, the search button is clicked.

The exams between the start and end date have been loaded and each exam link has a result link. The driver scrapes and stores in a list details like Exam duration, date, time e.t.c Using string splicing and manipulation, these details have stored in separate lists.

For each exam, the driver opens the result page and scrapes the Marks Obtained by the student and stores it in a List. Percentage of marks is also calculated for each exam.

Finally, pandas library is used to create a dataframe and store the data in an excel sheet.

INSTALLATION AND USAGE

Selenium, pandas, difflib and datetime modules have been used in this project

Pandas : https://pandas.pydata.org/docs/getting_started/install.html
Selenium : https://selenium-python.readthedocs.io/
Datetime : https://docs.python.org/3/library/datetime.html

Difflib: https://docs.python.org/3/library/difflib.html

Make sure the chromedriver executable is in the project folder

In order to run the script, type python grade_scraper.py

Future Of The Project

As of now only the marksobtained and percentage has been displayed on the excel sheet.

In the future, marks obtained in each section, number of correct answers, incorrect answers and skipped questions can also be implemented.

The data can be graphed and shown in the form of charts in order to help students realize their mistakes in a particular subject and find ways to improve.

Short term and long term progress graphs in a particular subject can also be implemented.

the driver can also scrape the marks obtained in various sections, like MCQ, descriptive, truth or false e.t.c so the student can improve on certain areas.

3.**Finding the research output of the faculty of a college**

When choosing a college to join, A student has to consider many factors like location, subject, placements, cost, internships, college life e.t.c However I feel that a very underrated factor to consider while joining a college is the faculty.

Especially for a research oriented student, it is important to look at the research interests of the professors at the college since you will be collaborating with them and working under them later on. The current college choosing websites mainly compare colleges based on cutoffs, placements, internships, cost e.t.c.

Hence in order to help students get a better idea of which college's faculty suits their interests more, and for them to compare the professors of various colleges, I have written two python scripts which make use of the BeautifulSoup, Selenium and requests libraries in order to scrape the names of professors from the faculty page of a college, search for their profiles in Google Scholar and obtain number of citations, h – index and i10 index of the professor.

If possible, I have also scraped the Research Interests of the professor and I have displayed the data in the form of an excel sheet using the pandas module for python.

IMPLEMENTATION

I have created two python scripts, one for IIIT Bangalore and one for IIT Bombay.

Since IIIT Bangalore has three faculty pages, I have iterated through them one by one and I have used requests and beautifulSoup libraries to pull information from the websites. Since each faculty member had the tag 'div' with the class = faculty-info, I used BeautifulSoup to search for all such elements with the same tag and class.

Then, the script iterates through the list of elements and extracts the name of each faculty member and their educational info and stores them in their respective lists.
Now that we have the name of the faculty member, we need to search for their profile on Google Scholar Website. Since there can be many people with the same name, using string addition, I have added 'iiitb' to the end of their name. This ensures that only the profiles from iiitb will show up in the search.

Using the requests library and the BeautifulSoup library, the contents of the google scholar search results have been parsed. If a professor does not have any search results show up, then the citations, h_hindex and i10 index values are all taken to be 0.

If a result is present, then the resultant webpage is again parsed by the BeatifulSoup library and the citation, h_index and i10 data has been scraped by finding the relevant HTML tag and class. This information is then stored in separate lists.

Finally, pandas is used to create a dataframe and store the data in an excel sheet

In the second script, the faculty page for iit Bombay https://www.cse.iitb.ac.in/people/faculty.php was accessed by a chrome webdriver using the selenium library. Similar to the first script, however this time using the XPATH of the HTML elements, the webdriver scraped the names of the faculty. It also scraped their research interests and google scholar links.

Similar to the first script, the names were appende with "iitb" and searched in Google Scholar and the data was scraped.

The pandas library is used to present the data in an excel sheet.

Installation and Usage

Requests, BeautifulSoup, selenium and pandas libraries have been used.

In order to run the script, type python iiitb.py
And for the second script, python iitb.py

The resultant files are iiitb.xlsx and iitb.xlsx

FUTURE OF THE PROJECT

I plan on adding multiple colleges and making a website which gives students the option to select a college and download the faculty details of that college as an excel file.

Since each college has a different way of storing data, a different script has to be designed for each different college.

I also plan on adding more visual representations of the data, such as bar graphs to compare the total number of citations between different professors or colleges.

I also plan on sorting the colleges based on a particular research interest.
For example, if the student's research interest is Artificial Intelligence, then the colleges can be sorted in the descending order of number of professors whose research interest includes Artificial Intelligence.