CSE 322 : Software Engineering

LAB 02

1) Open **PyCharm Community Edition.**
2) Install **Selenium Package** for python under PyCharm**.**

Selenium is a package which enables us to perform **Testing / Web Scraping** for finding out the bugs or other information from an Online System that is developed and waiting for implemented in real world market.

After you have installed **Selenium** on your system, we can start our journey with testing a developed online system. Today we will start from the very basics and will cover the following points one by one. The lab will progress with two steps:

- Demonstration from the teacher on today's discussion topics.
- Submission of an assignment on those topics from students individually.

Our today's lab will cover different topics for initial starting. Let's focus on the following demonstrations and topics first:

1) Installing **Driver** for web browser access.
2) Setting up a Project in PyCharm IDE.
3) Running First Initial Test on **Chrome/Firefox/IE.**

- **Installing Driver for Browser:**

Follow the following steps to download the WebDriver for your specific browser.

1) Open the browser and open this link https://www.selenium.dev/downloads/
2) You will find a section called **Platform Supported by Selenium.**
3) Under that section you will see download option for different browsers, select the option for your browser and click the link for downloading the driver from there**.** Go to the documentation of your used browser and download the latest stable release of the webdriver for your specific browser.
4) There you will find different WEBDRIVER download links for different browsers.
5) I use Firefox, and so I downloaded the **GeckoDriver** for that browser as you will find there.
   You can download for whichever browser you use.

- **Setting Up a Project in PyCharm:**

Follow the following steps to open a new project in PyCharm.

1) Open PyCharm, click on **File** from the tabs at the top.
2) Open a New project and give it a proper name and set it up.
3) From the left hand side, select the newly created project.
4) Click on **File** tab again and open **Settings.**
5) A new window will pop-up, and you can see there the packages that is installed for this project in PyCharm.
6) If you don't find **Selenium** there, click on the **'+'** icon at the right hand side and find **Selenium** in the new window.
7) Select **Selenium,** and install the package and you will see that the package will be installed for the project selected.

- **Run First Test using the WebDriver in PyCharm:**

1) Open a new **.py** file under the newly created project.
2) The first step is to use the **webdriver** that we downloaded for accessing a website automatically. See the code below-

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

driver = webdriver.Firefox(executable_path='C:/Sel_Firefox/geckodriver.exe')
```

Ok, so here we are exporting the in-built asset of **webdriver** from selenium package. Then we need to initialize the object webdriver for our web browser. In the **2nd** line you can see that we have specified a path where the **Geckodriver** for my automatic browser access is stored since the webdriver needs to understand which software it is going to use.

The path specifying codeline might be a bit different for different browsers. The above one is for **Firefox** so I wrote **webdriver.Firefox()**. For chrome it will be **webdriver.Chrome().** As a parameter of that function we need to pass the **executable_path** where the driver you have stored.

3) Next step is to open up the browser automatically and access a website. To do that we need to do the following:

```
driver.get('https://www.google.com')
```

So this line of code should open up your browser and access the specified website mentioned inside. After we have completed our work here we now need to access some attribute of the website which we can test.

4) The very first and primary element of any website page is it's **Title.** So after we get access of the website first, we now want to access the title of that page. So in order to do that, let's write the following command-

```
    def page_title() :
        #return the title of the webpage
        return driver.title;
```

Here, we define a function which will return the Title of our current webpage that the **Webdriver** is in. Now after defining the function we simply just need to close the browser and check if it is retrieving the title or not.

5) To close and retrieve the title of the page in our PyCharm project, we need to do the following:

```
    driver.quit();
```

After closing the browser, we just need to call our function to see if the intended function could retrieve the title of the page or not. We just simply write,

```
test = page_title()   #will run the function and return the desired value

print(test)   #this will print out the retrieved title of the page
```

CONGRATULATIONS! Your journey with webscraping and testing with Python language has just started successfully.

P.T.O.

The next section is trying out some basic commands which my **webdriver** can perform. I am listing out the commands below, and your task is to use these and perform some basic functions with you Webdriver. The functions are name;y-

- **driver.title :** retrieves the title of the current page
- **driver.current_url :** retrieves the URL of the current page
- **driver.page_source :** retrieves the HTML code source of the whole page

Okay let's move on to some new works that we can proceed on from here. Below we discuss some basic **WebDriver commands.** Suppose we are in a website through our webdriver and we want to click on some **Button** there and move on to the redirected page.

- **Clicking a Button in a page automatically:**

  Follow the class demonstration first, and then take the code help from here. So we can automate the clicking process by different ways. We here discuss two of them-

  1) **By Element text**
  2) **By Element XPATH**

For both of the process, we need inspect the element and copy it's text or XPATH and either of the following.

```
def click_button() :
    #clicking a button by two ways
    driver.find_element_by_link_text('Mobile Dev').click()
    driver.find_element_by_xpath('YOUR XPATH HERE').click()
```

- **Navigating through website pages:**

  Suppose you have launched a page through the webdriver. Then after sometimes you have launched another one webpage from there. Now, you want to switch back to the previous webpage. So rather again opening up writing a URL, you can navigate to the pages in your history. And again if you want to move back to the latest page opened, you can navigate forward by your webdriver. There are simply two commands-

  driver.back() #moves to the previous page
  driver.forward() #moves to the forward page

Let’s observe the demonstration first and then try it out.