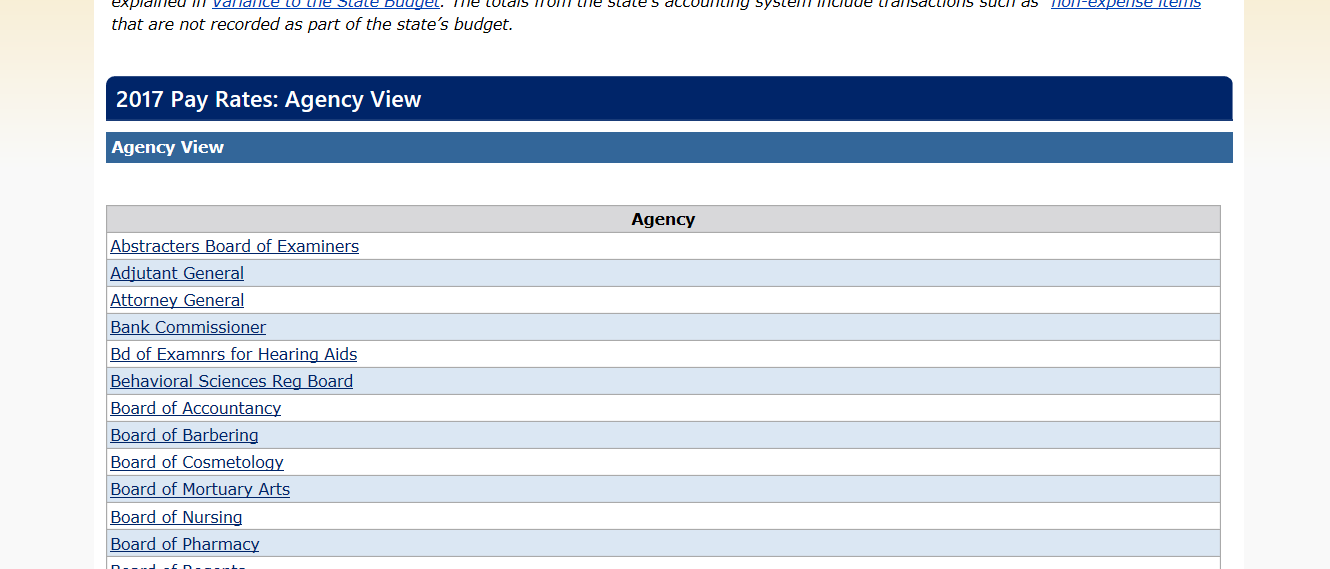
**Web Scrapper Example for Web Pages Using JavaScript Links:**

In the previous uploads we have talked about Web Scraping with Beautiful Soup, Crawling to Web pages etc. But there are some Websites which want us to click on some buttons or submit a form before loading some content. These web pages use JavaScript to serve the content. In case of these websites we cannot directly extract the data using Beautiful Soup. Here Selenium comes into picture. Selenium is a python web scraping library which is useful for extract data from the webpages which uses JavaScript.

Let’s take an example of a website like this –

<http://kanview.ks.gov/PayRates/PayRates_Agency.aspx>



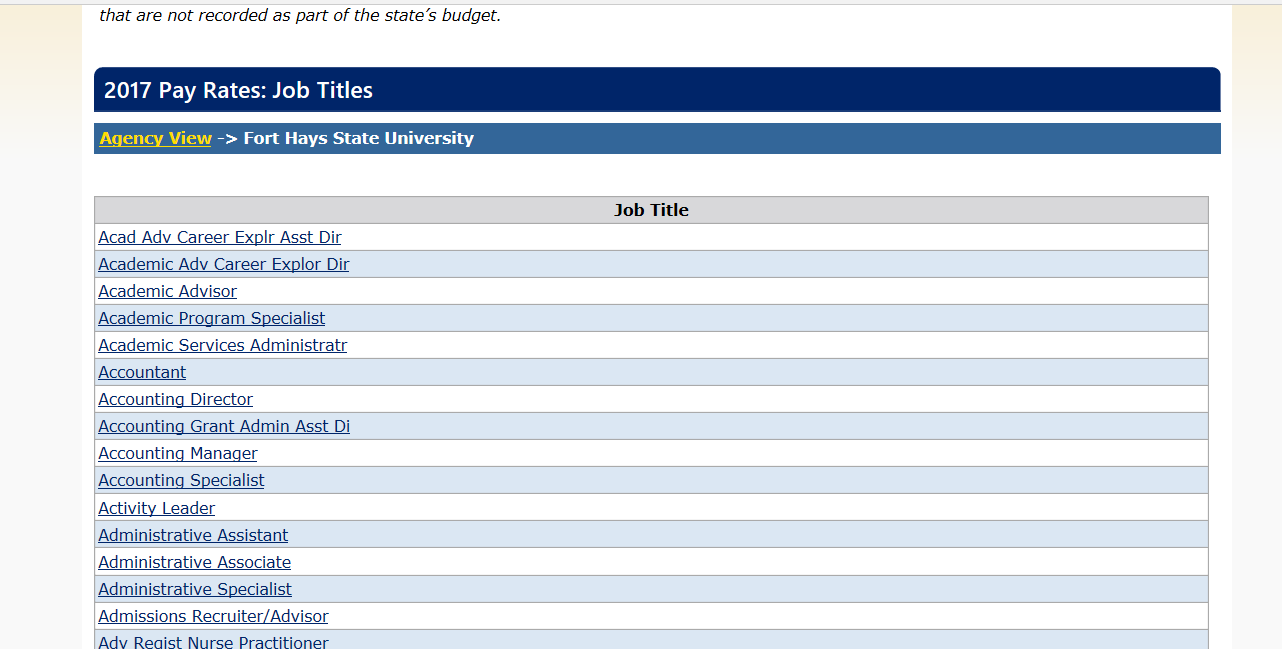


This website gives information about salaries for different job roles for different Govt. agencies.

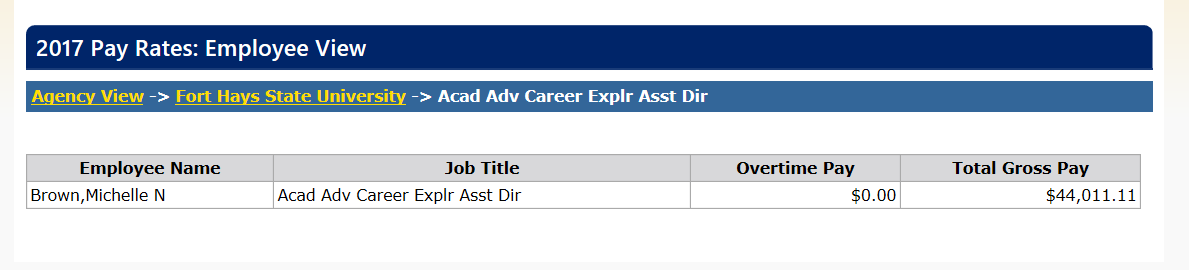
We want to know **the salaries for all the job titles of Fort Hays State University (FHSU) Agency**.

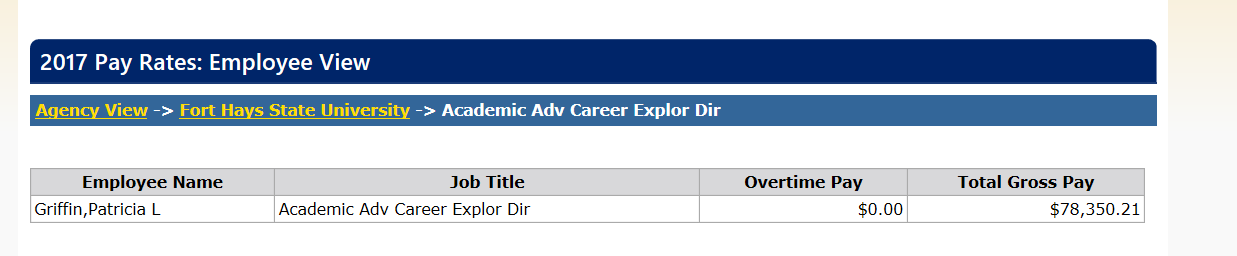


If we click on the FHSU link we can see the job titles –



If we click on any Job Title it shows the total gross pay for that job title –





We want a web scraper which will click on each job title for FHSU and will extract all the information available there.

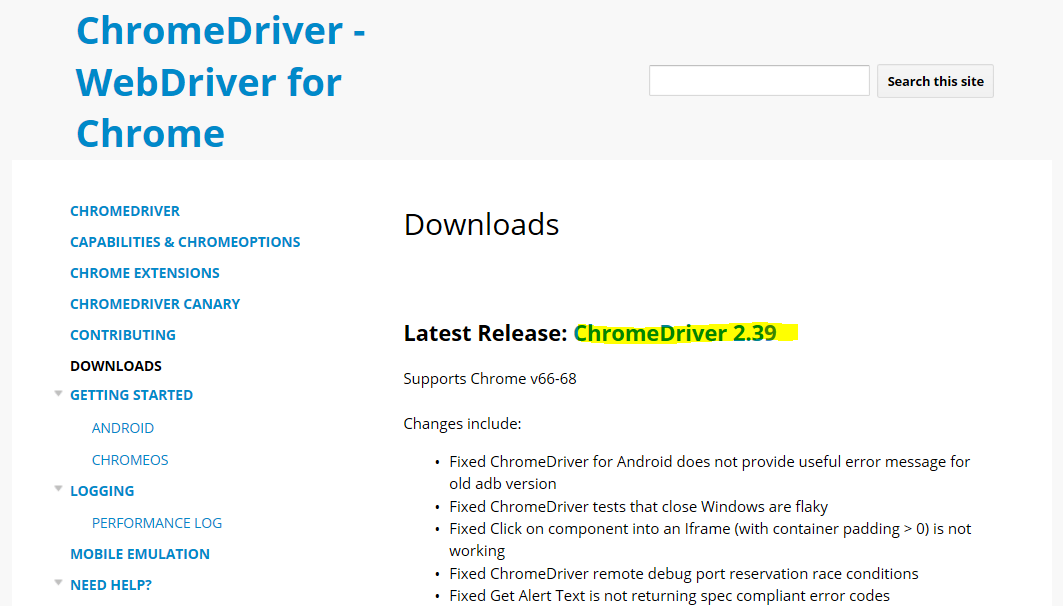
Beautiful Soup is helpful when we want web scraping by traversing the DOM (Document Object Model). But here this website uses JavaScript links. Here beautiful soup will not work without some extra addition. Here the role of Selenium comes. Using Selenium we can automate Web Browser interaction from Python.

Selenium will start a new Browser session. For selenium to work it needs to access the **browser’s driver**. The drivers for the standard browsers like Chrome, Firefox, Edge, Safari are available online. One must download the appropriate browser driver and place it in the same directory before running this script.

This is the link to download the web driver for google chrome browser-

<https://sites.google.com/a/chromium.org/chromedriver/downloads>

Click on the latest release and download and install it.



Later you need to mention the path of the web driver in the code.

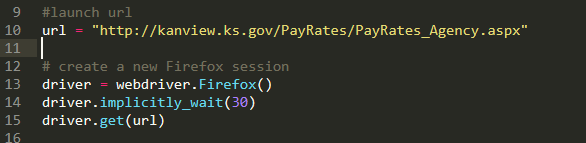
First we need to import these libraries for Selenium –

**from selenium import webdriver**

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

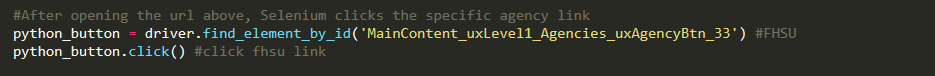
Please check this URL for the complete code.

Here I am explaining some parts of the code –

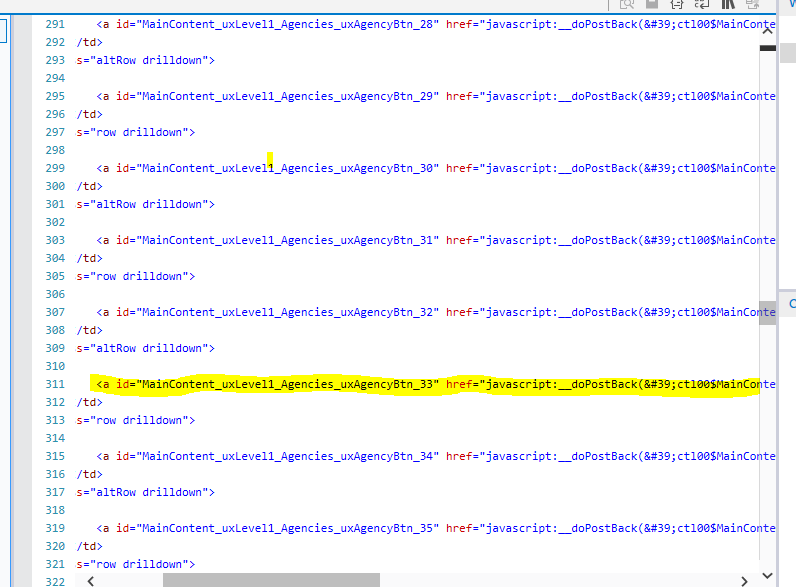


We are using a new Firefox session using code and opening this URL.

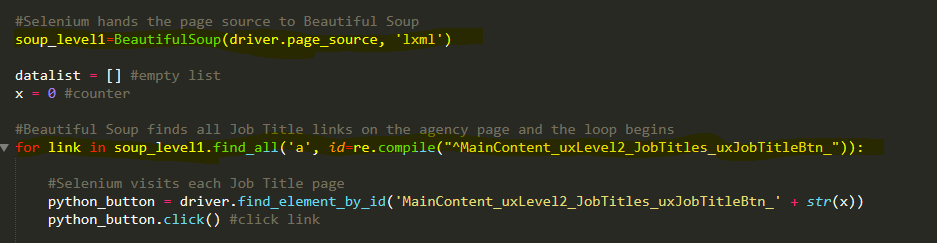
Selenium needs to find the FHSU link and click on it –



If we check the source code of this web page this is the id of FHSU link-

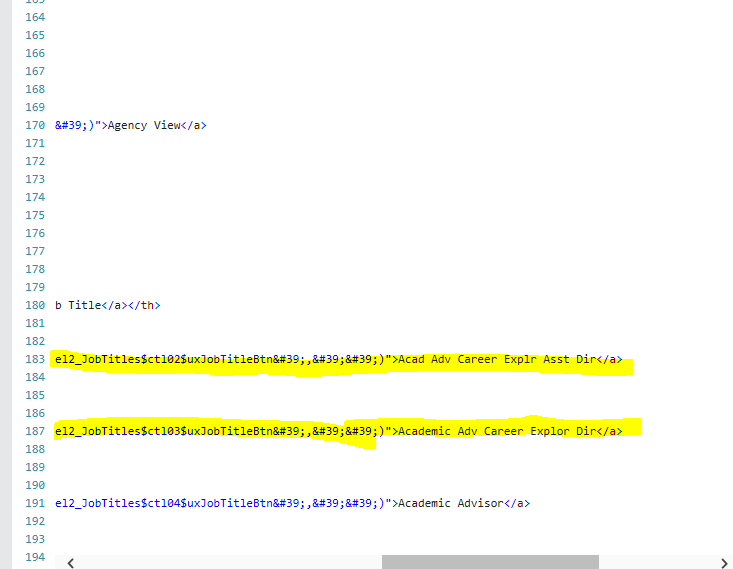




Selenium then hands the page source to beautiful soup. 

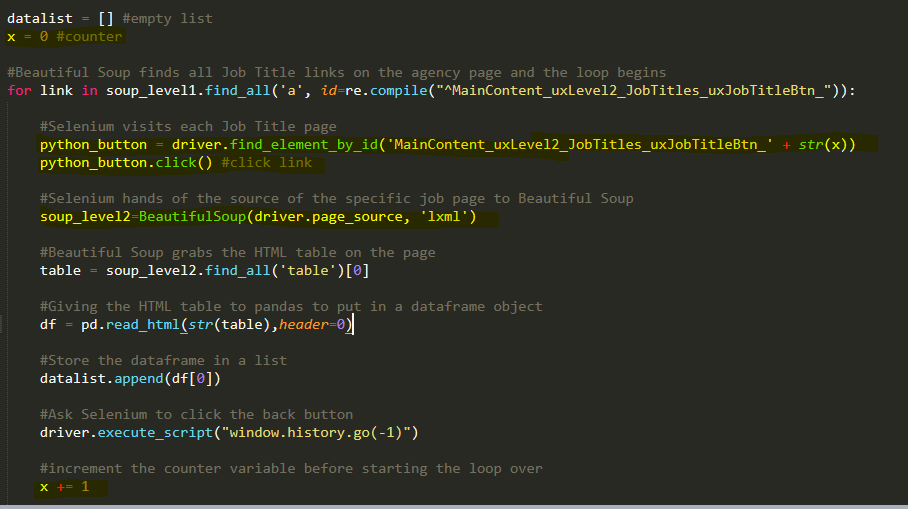
If we click on the FHSU link and then view the page source then we will find out how the id looks for each job title –



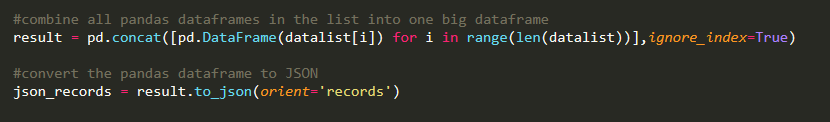


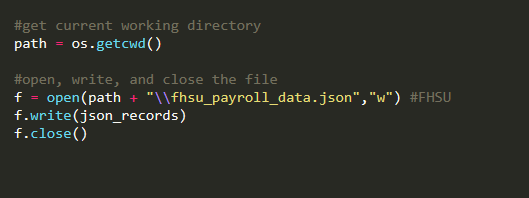
We can find that each job title id starts with **MainContent\_uxLevel2\_JobTitles\_uxJobTitleBtn\_** and then a number which increases from 0 onwards.

So we take a counter as **x=0,** create a **for loop** and increase the value of the counter after each iteration of the loop. In each iteration selenium hands the page source to Beautiful Soup. Beautiful Soups grabs the html table on the page and stores it in a Pandas dataframe.

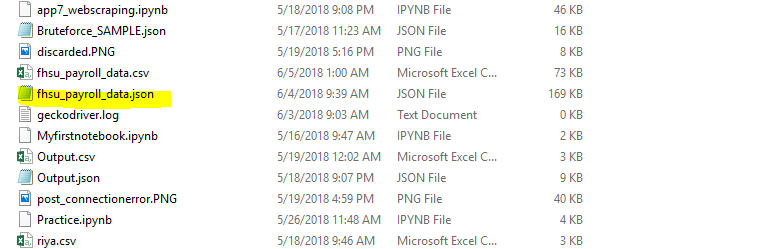


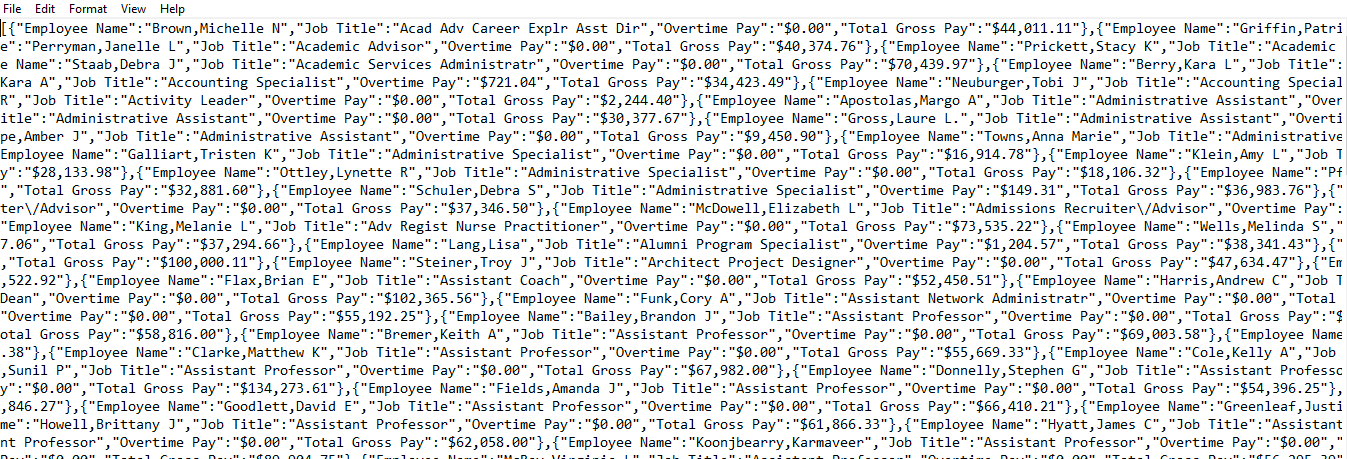
After selenium has clicked on each job title and extracted the data the loop is completed. After that each Pandas dataframe created in every loop are combines into a bigger data frame and then it is converted into json.





And this is the json file created.





We know that we can post dynamic data to DNIF if it is in json format. This is the code for the same.

Please refer this code for the same-

<https://github.com/dnif/DigiVigi/blob/master/Process_2/PS_SourceToDnif.py>

