Web Scraping using Selenium - Website Login

Objectives:

- How to use Selenium to login a website

We will use following website to demonstrate

- https://newsapi.org/login (https://newsapi.org/login)

Install Python library `selenium` and `webdriver_manager` using `pip` .

In [1]:
```
!pip install selenium
!pip install webdriver_manager
```

```
Requirement already satisfied: selenium in c:\users\isszq\anaconda3\envs\py
dot36\lib\site-packages (3.141.0)
Requirement already satisfied: urllib3 in c:\users\isszq\anaconda3\envs\pyd
ot36\lib\site-packages (from selenium) (1.25.9)
Requirement already satisfied: webdriver_manager in c:\users\isszq\anaconda
3\envs\pydot36\lib\site-packages (3.2.1)
Requirement already satisfied: requests in c:\users\isszq\anaconda3\envs\py
dot36\lib\site-packages (from webdriver_manager) (2.24.0)
Requirement already satisfied: crayons in c:\users\isszq\anaconda3\envs\pyd
ot36\lib\site-packages (from webdriver_manager) (0.3.1)
Requirement already satisfied: configparser in c:\users\isszq\anaconda3\env
s\pydot36\lib\site-packages (from webdriver_manager) (5.0.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\isszq\anacond
a3\envs\pydot36\lib\site-packages (from requests->webdriver_manager) (2020.
6.20)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
c:\users\isszq\anaconda3\envs\pydot36\lib\site-packages (from requests->web
driver_manager) (1.25.9)
Requirement already satisfied: idna<3,>=2.5 in c:\users\isszq\anaconda3\env
s\pydot36\lib\site-packages (from requests->webdriver_manager) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in c:\users\isszq\anaconda
3\envs\pydot36\lib\site-packages (from requests->webdriver_manager) (3.0.4)
Requirement already satisfied: colorama in c:\users\isszq\anaconda3\envs\py
dot36\lib\site-packages (from crayons->webdriver_manager) (0.4.3)
```

Import libraries

In [2]:
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.wait import WebDriverWait

from webdriver_manager.chrome import ChromeDriverManager
```

1. Interaction with UI

Selenium not only help you to load web pages. It also able to interact with websites to simulate user actions, e.g. login to a website.

We will demonstrate how to login https://newsapi.org (https://newsapi.org) using Selenium.

Open Website

In [31]:
```
from selenium import webdriver
from webdriver_manager.chrome import ChromeDriverManager

browser = webdriver.Chrome(ChromeDriverManager().install())

url = 'https://newsapi.org/login'
browser.get(url)
```

```
[WDM] - Current google-chrome version is 83.0.4103
[WDM] - Get LATEST driver version for 83.0.4103
[WDM] - Driver [C:\Users\isszq\.wdm\drivers\chromedriver\win32\83.0.4103.39
\chromedriver.exe] found in cache
```

Identify Elements and Interact with Them

Find elements in the browser and create reference to them.

Inspect the text inputs for email and password. They are identified by `id`.

In [32]:
```
input_email = browser.find_element_by_id('Email')
input_password = browser.find_element_by_id('Password')
input_email
input_password
```

Out[32]:
```
<selenium.webdriver.remote.webelement.WebElement (session="033065d6b5cacc88
1599d8cec0dcb8df", element="a8b9bb7b-cf1e-473e-98ae-1f5dbef4dbe8")>
```

Inspect the Login button. It contains a span with text `Login`.

In [33]:
```
button_login = browser.find_element_by_xpath('//button/span[contains(tex
# button_login = browser.find_element_by_xpath('//button[@class="btn btn
button_login.text
```

Out[33]: 'Login'

Enter text into inputs using `send_keys()` function.

In [34]:
```
input_email.send_keys('mark.qj@gmail.com')
input_password.send_keys('abcd1234')
```

Simulate a click event on the button using `click()` function.

In [35]:
```
button_login.click()
```

Verification

How to verify whether login is successful?

- Identify an element on the web page after successful login.
- Check whether that element exists or not.

In [36]:
```
try:
    button_logout = browser.find_element_by_xpath('//button/span[contains
    print('Login successful')
except:
    print('Login failed')
```

```
Login successful
```

Clean Up

Always close the browser after the task is completed.

In [37]:
```
browser.close()
```