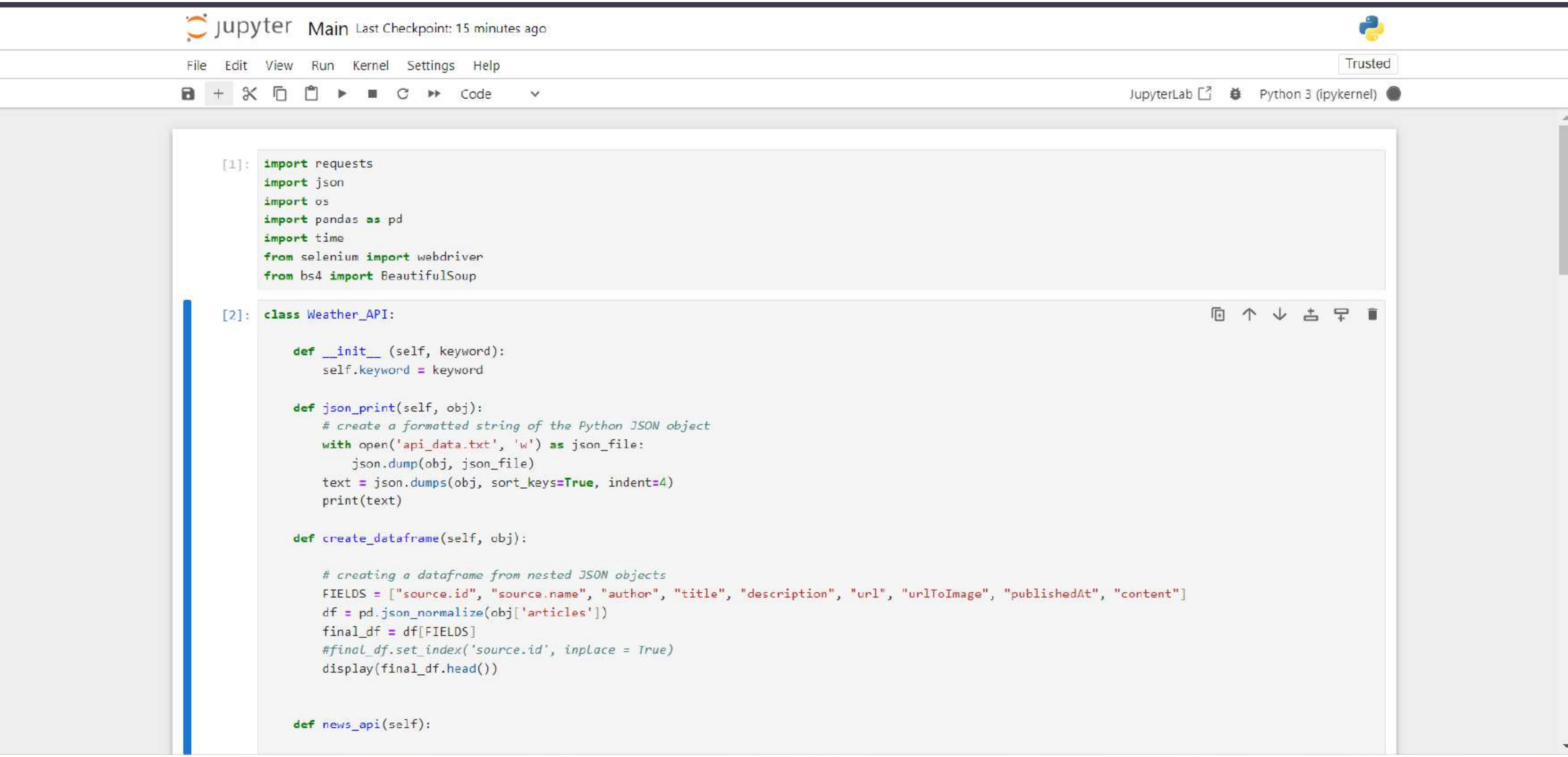
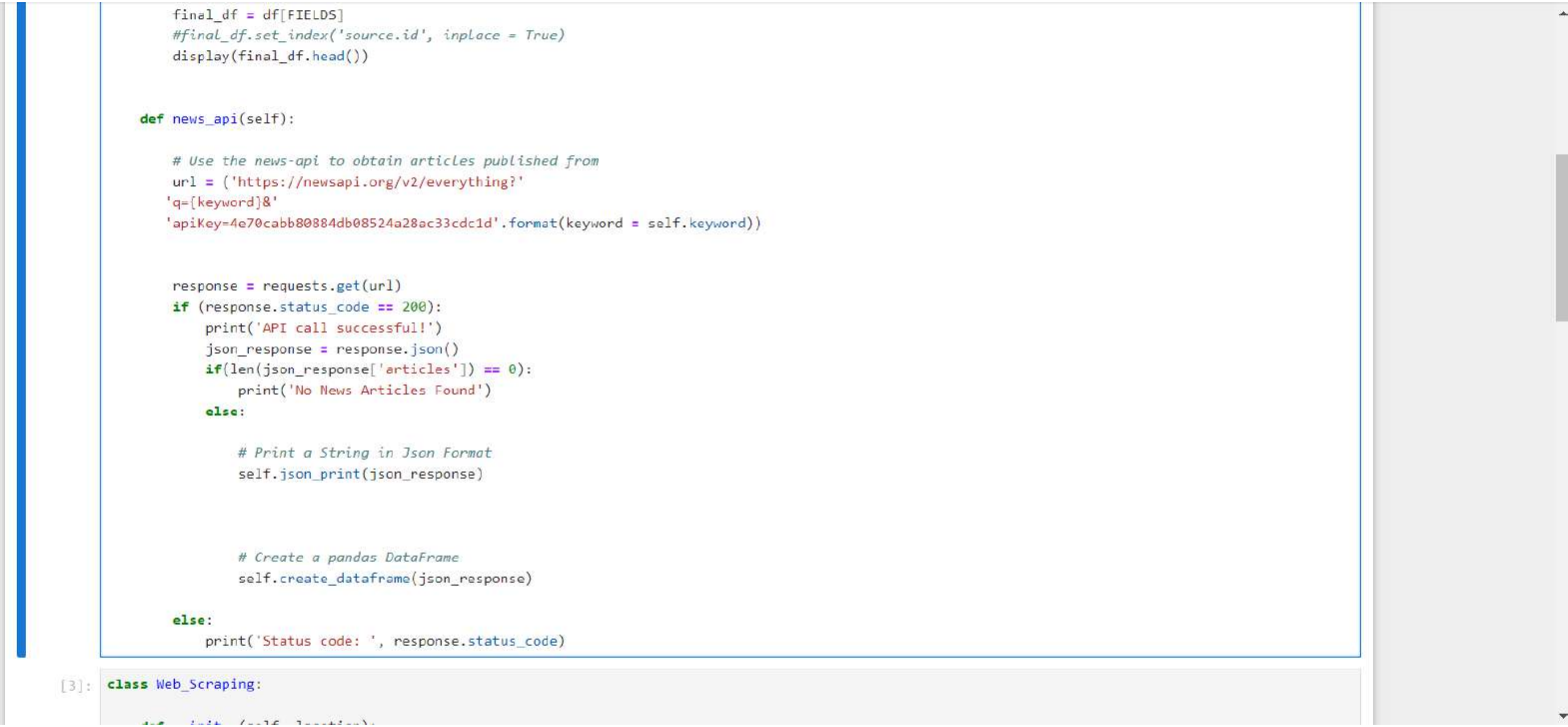


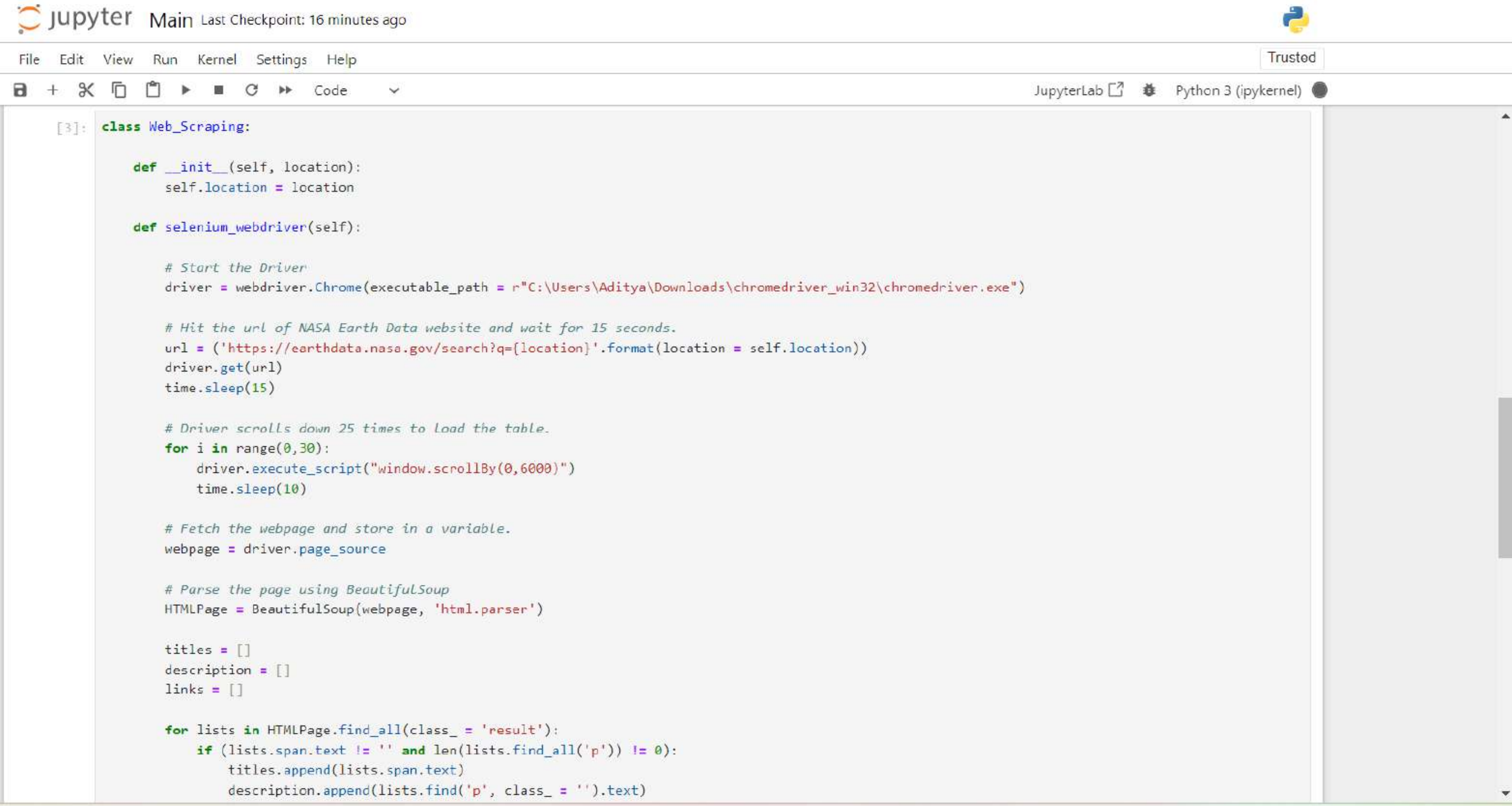
[1]:
```python
import requests
import json
import os
import pandas as pd
import time
from selenium import webdriver
from bs4 import BeautifulSoup
```

[2]:
```python
class Weather_API:

    def __init__ (self, keyword):
        self.keyword = keyword

    def json_print(self, obj):
        # create a formatted string of the Python JSON object
        with open('api_data.txt', 'w') as json_file:
            json.dump(obj, json_file)
        text = json.dumps(obj, sort_keys=True, indent=4)
        print(text)

    def create_dataframe(self, obj):

        # creating a dataframe from nested JSON objects
        FIELDS = ["source.id", "source.name", "author", "title", "description", "url", "urlToImage", "publishedAt", "content"]
        df = pd.json_normalize(obj['articles'])
        final_df = df[FIELDS]
        #final_df.set_index('source.id', inplace = True)
        display(final_df.head())


    def news_api(self):
```

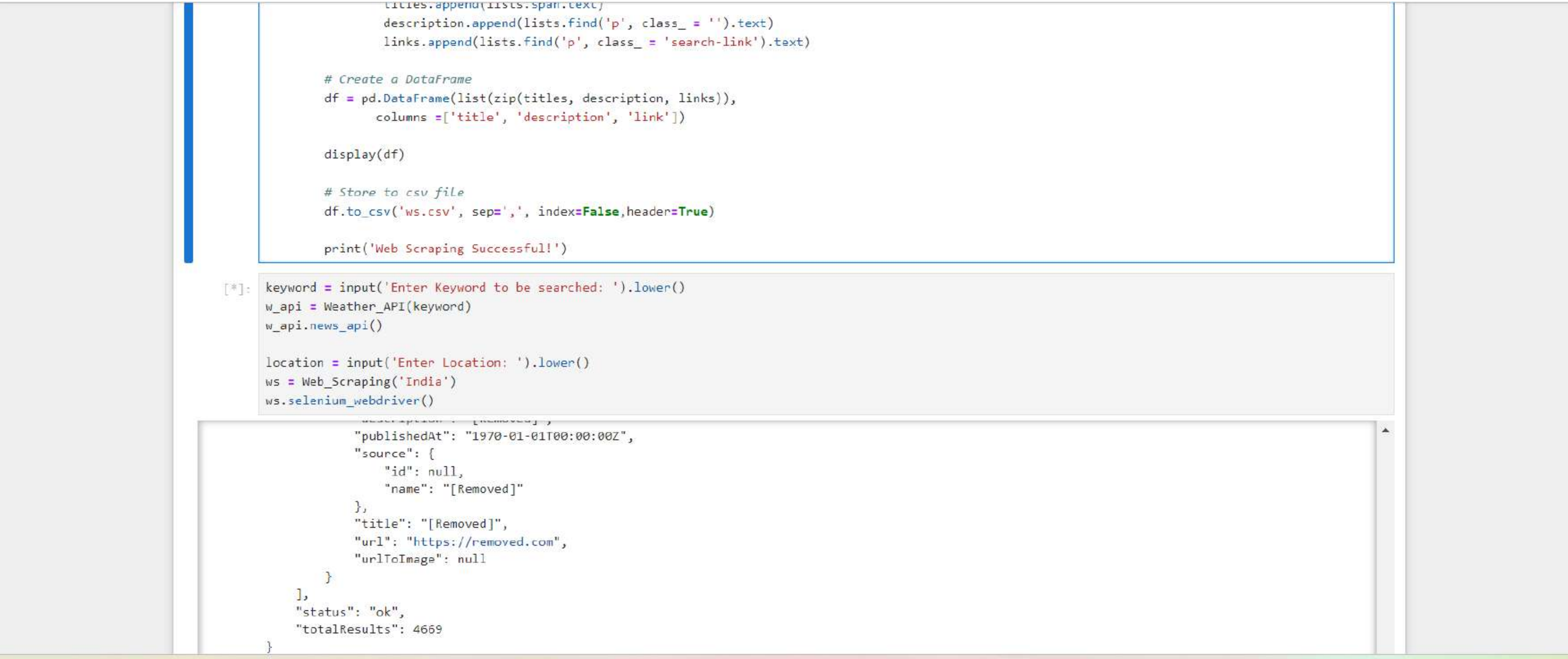


```
        final_df = df[FIELDS]
        #final_df.set_index('source.id', inplace = True)
        display(final_df.head())


    def news_api(self):

        # Use the news-api to obtain articles published from
        url = ('https://newsapi.org/v2/everything?'
       'q={keyword}&'
       'apiKey=4e70cabb80884db08524a28ac33cdc1d'.format(keyword = self.keyword))


        response = requests.get(url)
        if (response.status_code == 200):
            print('API call successful!')
            json_response = response.json()
            if(len(json_response['articles']) == 0):
                print('No News Articles Found')
            else:

                # Print a String in Json Format
                self.json_print(json_response)



                # Create a pandas DataFrame
                self.create_dataframe(json_response)

        else:
            print('Status code: ', response.status_code)
```

```
[3]: class Web_Scraping:

         def __init__(self, location):
```



```
class Web_Scraping:

    def __init__(self, location):
        self.location = location

    def selenium_webdriver(self):

        # Start the Driver
        driver = webdriver.Chrome(executable_path = r"C:\Users\Aditya\Downloads\chromedriver_win32\chromedriver.exe")

        # Hit the url of NASA Earth Data website and wait for 15 seconds.
        url = ('https://earthdata.nasa.gov/search?q={location}'.format(location = self.location))
        driver.get(url)
        time.sleep(15)

        # Driver scrolls down 25 times to load the table.
        for i in range(0,30):
            driver.execute_script("window.scrollBy(0,6000)")
            time.sleep(10)

        # Fetch the webpage and store in a variable.
        webpage = driver.page_source

        # Parse the page using BeautifulSoup
        HTMLPage = BeautifulSoup(webpage, 'html.parser')

        titles = []
        description = []
        links = []

        for lists in HTMLPage.find_all(class_ = 'result'):
            if (lists.span.text != '' and len(lists.find_all('p')) != 0):
                titles.append(lists.span.text)
                description.append(lists.find('p', class_ = '').text)
```



```
            titles.append(lists.span.text)
            description.append(lists.find('p', class_ = '').text)
            links.append(lists.find('p', class_ = 'search-link').text)

    # Create a DataFrame
    df = pd.DataFrame(list(zip(titles, description, links)),
           columns =['title', 'description', 'link'])

    display(df)

    # Store to csv file
    df.to_csv('ws.csv', sep=',', index=False,header=True)

    print('Web Scraping Successful!')
```

```
keyword = input('Enter Keyword to be searched: ').lower()
w_api = Weather_API(keyword)
w_api.news_api()

location = input('Enter Location: ').lower()
ws = Web_Scraping('India')
ws.selenium_webdriver()
```

```
                "description": "[Removed]",
                "publishedAt": "1970-01-01T00:00:00Z",
                "source": {
                    "id": null,
                    "name": "[Removed]"
                },
                "title": "[Removed]",
                "url": "https://removed.com",
                "urlToImage": null
            }
        ],
        "status": "ok",
        "totalResults": 4669
    }
```