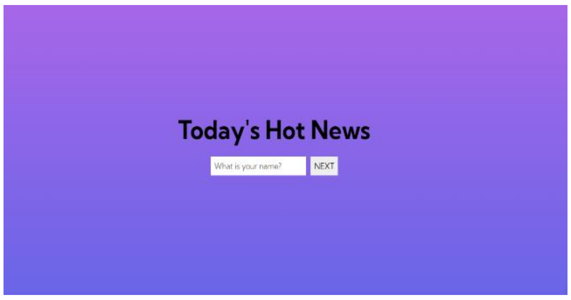
Hun Min Park

TODAY's HOT NEWS

FINAL PROJECT FROM QMNB COURSE

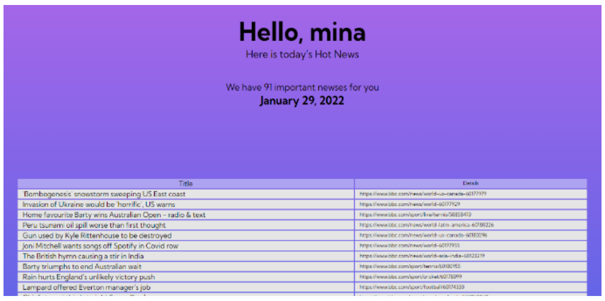
Result Preview


Today's Hot News

What is your name?

NEXT

Main Website Page


Hello, mina

Here is today's Hot News

We have 91 important newses for you

January 29, 2022

Title	Details
'Bombogenesis' snowstorm sweeping US East coast	https://www.bbc.com/news/world-us-canada-60177979
Invasion of Ukraine would be 'horrific', US warns	https://www.bbc.com/news/world-60177929
Home favourite Barty wins Australian Open – radio & text	https://www.bbc.com/sport/live/tennis/58858410
Peru tsunami oil spill worse than first thought	https://www.bbc.com/news/world-latin-america-60180226
Gun used by Kyle Rittenhouse to be destroyed	https://www.bbc.com/news/world-us-canada-60180296
Joni Mitchell wants songs off Spotify in Covid row	https://www.bbc.com/news/world-60177933
The British hymn causing a stir in India	https://www.bbc.com/news/world-asia-india-60123319
Barty triumphs to end Australian wait	https://www.bbc.com/sport/tennis/60180955
Rain hurts England's unlikely victory push	https://www.bbc.com/sport/cricket/60178099
Lampard offered Everton manager's job	https://www.bbc.com/sport/football/60174330

News Report Website Page

Purpose

Web scraping is one of must-know knowledge as a data analysis to collect the data efficiently in these days, because it helps access good data sources and save valuable time before going to data analysis phase. From this significant importance of accessing proper data sources, my project brings the today's headlines from two different news websites (BBC and CNN). It provides the web digital news service such as google news. This project is, especially, highly relevant to my interest for my future career, which is related to web development, so that I decided to go more further to present this project on the website.

Tasks

1. Scrap the main headlines from each websites
2. Save the headlines into CSV file
3. Present the headlines on the website for users

Process

1. Scrap the main headlines from each websites

- **Selenium Webdriver, BeautifulSoup**

  Instead of Python Basic Library *urllib*, I used more powerful library called *BeautifulSoup*, which we handled also in lecture. However, only with *BeautifulSoup*, I couldn't get some elements, what I want to scrap, which were used by Java Script. So, I used *selenium* with *webdriver* for keeping stronger and more stable.

  Only disadvantage of this selenium webdriver was that it takes long time to load pages. Since the website was opened by selenium for collecting the data, it was also quite burdensome to wait until the end of whole process.

Here was my idea, that users don't see the whole process from opening each website browsers by selenium to closing. The option has to be also added to not open the browser every time, when the *selenium* works with *webdriver* as following:

```
    op = webdriver.ChromeOptions()
    op.add_argument('headless')
    driver = webdriver.Chrome(ChromeDriverManager().install(), options=op)
```

- **Analysis of HTML Structures**
  Each website has different HTML Structures, so that I had to spend a lot of times at first to analyzes about the structure and how I can get most efficiently to get the results without duplication or with using a lot of for loops. And I had a difficulty to use the function such as *find_all*, because sometimes I couldn't use again *find_all* after using it in same flat. I found out later that *find_all* can be used again in for loop under itself.

  - **CNN Website**

```
        articles = page_soup.find_all("div", {"class":"cd__content"})
    for article in articles:
        for a in article.find_all('a', href=True):
        ...
        headline = article.find("span", {"class":"cd__headline-text"})
        headline_text = headline.get_text()
        headlines.append(headline_text)
```

  - **BBC Website**

```
      s1 = page_soup.find_all("a", {"class":"media__link"})
        for s in s1:
            href = s.attrs['href']
            links.append(URL_new+href)
            ...
            #strip for getting rid of unnecessary spaces
            headline_text = s.get_text().strip()
            headlines.append(headline_text)
```

- **Make Lists and Dictionaries**
  I learned from this part that the *for* loops can makes everything way easier and also difficult your tasks depends on where you put elements (inside or outside of for loops). Because, this was the most time-consuming part for me due to small mistake, that I put the empty lists [] for the data sets <u>not outside</u> of for loop, but <u>inside</u> of for loops.
  After solving this problem, I put each list from URL and news headlines into dictionary by sorting keys 'headline' and 'url' as following:

```
    links_with_text = []
    headlines = []
    ...
    for article in articles:
    ...
    dict_cnn = [{'headline' : headlines, 'url' : links_with_text} for headlines,
    links_with_text in zip(headlines,links_with_text)]
```

There was a problem for scraping the URL from each headline, because they have different patterns. Some of them starts only short version of URL without main website address such as https://www.cnn.com/ and some of them has this main address included. It was necessary to make the standard URL address for all with if-statement.

2. Save the headlines into CSV file

- **Saving a list of the dictionaries into CSV files**
  Keys and Values are saved in different columns. According to Python 3.X, It was necessary to use the option 'w', newline= ' ' for eliminating the blanks after each row.

```
keys = dict_cnn[0].keys()

a_file = open('output_cnn.csv', 'w', newline='')
dict_writer = csv.DictWriter(a_file, keys)
dict_writer.writeheader()
dict_writer.writerows(dict_cnn)
a_file.close()
```

- **Merge two different CSV files into one single CSV file**
  - **Problem: Duplication of URL**

For this, I needed to study about URL Structure and components, because it caused unexpected duplication of domain address. Because they had different URL structure as following:
/sport/cricket/60178099   or   https://www.bbc.com/sport/cricket/60178099

Before)

```
URL_new = "https://www.bbc.com"
for s in s1:
    href = s.attrs['href']
    links.append(URL_new+href)
```

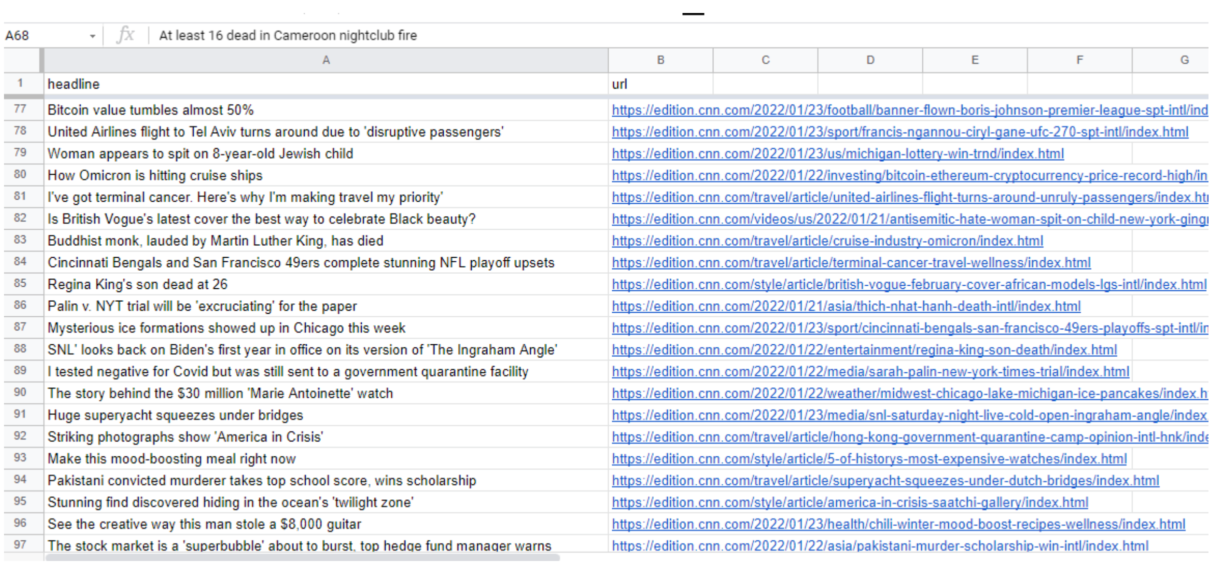
```
Result>>    https://www.bbc.comhttps://www.bbc.com/sport/cricket/60178099
```

To fix this, I tried every different method such as if-statement and try and so on. But they didn't work for my code.

I used *urlparse* API to let analyze each component of URLs and replaced the empty components into proper value, which makes complete URL address. After this processing, the result has to changed again from string to URLs as following:

After)

```
href = a.attrs['href']
href_n = urlparse(href)
href_new = href_n._replace(scheme='https', netloc='edition.cnn.com')
url = (href_new.geturl())
```


A68 | *fx* | At least 16 dead in Cameroon nightclub fire

	A	B	C	D	E	F	G
1	headline	url					
77	Bitcoin value tumbles almost 50%	https://edition.cnn.com/2022/01/23/football/banner-flown-boris-johnson-premier-league-spt-intl/ind					
78	United Airlines flight to Tel Aviv turns around due to 'disruptive passengers'	https://edition.cnn.com/2022/01/23/sport/francis-ngannou-ciryl-gane-ufc-270-spt-intl/index.html					
79	Woman appears to spit on 8-year-old Jewish child	https://edition.cnn.com/2022/01/23/us/michigan-lottery-win-trnd/index.html					
80	How Omicron is hitting cruise ships	https://edition.cnn.com/2022/01/22/investing/bitcoin-ethereum-cryptocurrency-price-record-high/in					
81	I've got terminal cancer. Here's why I'm making travel my priority'	https://edition.cnn.com/travel/article/united-airlines-flight-turns-around-unruly-passengers/index.ht					
82	Is British Vogue's latest cover the best way to celebrate Black beauty?	https://edition.cnn.com/videos/us/2022/01/21/antisemitic-hate-woman-spit-on-child-new-york-ging					
83	Buddhist monk, lauded by Martin Luther King, has died	https://edition.cnn.com/travel/article/cruise-industry-omicron/index.html					
84	Cincinnati Bengals and San Francisco 49ers complete stunning NFL playoff upsets	https://edition.cnn.com/travel/article/terminal-cancer-travel-wellness/index.html					
85	Regina King's son dead at 26	https://edition.cnn.com/style/article/british-vogue-february-cover-african-models-lgs-intl/index.html					
86	Palin v. NYT trial will be 'excruciating' for the paper	https://edition.cnn.com/2022/01/21/asia/thich-nhat-hanh-death-intl/index.html					
87	Mysterious ice formations showed up in Chicago this week	https://edition.cnn.com/2022/01/23/sport/cincinnati-bengals-san-francisco-49ers-playoffs-spt-intl/in					
88	SNL' looks back on Biden's first year in office on its version of 'The Ingraham Angle'	https://edition.cnn.com/2022/01/22/entertainment/regina-king-son-death/index.html					
89	I tested negative for Covid but was still sent to a government quarantine facility	https://edition.cnn.com/2022/01/22/media/sarah-palin-new-york-times-trial/index.html					
90	The story behind the $30 million 'Marie Antoinette' watch	https://edition.cnn.com/2022/01/22/weather/midwest-chicago-lake-michigan-ice-pancakes/index.h					
91	Huge superyacht squeezes under bridges	https://edition.cnn.com/2022/01/23/media/snl-saturday-night-live-cold-open-ingraham-angle/index					
92	Striking photographs show 'America in Crisis'	https://edition.cnn.com/travel/article/hong-kong-government-quarantine-camp-opinion-intl-hnk/inde					
93	Make this mood-boosting meal right now	https://edition.cnn.com/style/article/5-of-historys-most-expensive-watches/index.html					
94	Pakistani convicted murderer takes top school score, wins scholarship	https://edition.cnn.com/travel/article/superyacht-squeezes-under-dutch-bridges/index.html					
95	Stunning find discovered hiding in the ocean's 'twilight zone'	https://edition.cnn.com/style/article/america-in-crisis-saatchi-gallery/index.html					
96	See the creative way this man stole a $8,000 guitar	https://edition.cnn.com/2022/01/23/health/chili-winter-mood-boost-recipes-wellness/index.html					
97	The stock market is a 'superbubble' about to burst, top hedge fund manager warns	https://edition.cnn.com/2022/01/22/asia/pakistani-murder-scholarship-win-intl/index.html					

Combined two CSV files into one CSV file in excel sheet

3. Present the headlines on the website for users

First and foremost, I had to figure it out, how I can present python file to html. This was the part what I struggled most time at the end, because my previous projects were HTML-based websites. But this should be dealt with the other way round, so that I had to try different web frameworks and I finally chose *Flask*. *Flask* is a small and lightweight Python-based web framework that provides useful tools and features that make creating web applications in Python easier.

- ***Flask* virtual environment setting with *venv***

For isolating my application, I had to set a virtual environment, which is a directory that contains the software on which its application depends. A virtual environment also changes environment variables to keep the development environment contained. Instead of downloading packages to system-wide or user-wide. This helps it easy to specify which Python binary to use and which dependencies we want to have available on a per project basis. As recommended, I downloaded Flask in virtual environment setting to an isolated directory used only for my current application.

- **Rendering a template and Using Variables**

A first step is to display a message that greets visitors on the index page. For this, *render_template* is one of important Flask function to serve an HTML template as the response, which is used to generate output from a template file based on the Jinja2 engine that is found in the application's templates folder.

First of all, I defined a view function (which is a Python function that returns an HTTP response) called *home()* using the *app.route()* decorator, which converts a regular function into a view function. This view function uses the *render_template()* function to render a template file called index.html.

  - Index.html

```
from flask import Flask, redirect, render_template, request

app = Flask(__name__, template_folder='template')
@app.route("/")
def home():
    return render_template('index.html')
```

After rendering an index page, I made a report.html using the app.route() decorator. Beside of this, I added a funny function, that website print out the name of user for greeting in next page for more interaction. Here I passed a variable called *SearchingBy, resultsNumber, headline_list, today* to the report.html template with each value, so that I can use this values also in html.

- report.html

```
@app.route("/report")
def report():
    name = request.args.get('name')
    # for making as lowercase
    if name:
        name = name.lower()
        #print(headline_list)
    else:
        redirect("/")

    return render_template('report.html', SearchingBy = name, resultsNumber =
           len(df), headline_list = df, today = today_n)

if __name__ =="__main__":
    app.run(host="0.0.0.0", debug=True)
```

- **Using Loops in HTML**

I used for-loops to go through headline lists to display each item in the list.

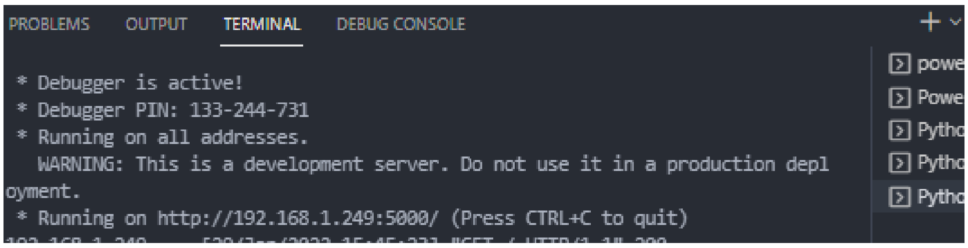
With *Jinja* engine, for-loop in the line {% for comment in comments %} goes through each headline in the headlines list. Also, I used table element in html. For, signal the ending of the for loop using the {% endfor %} keyword. This is different from the way Python for loops are constructed because there is no special indentation in *Jinja* templates. HTML files can be found inside of the template folder.

- **Style with CSS**

For Styling the html, I tried two different methods to adjust CSS on index.html and report.html as following.

- First method: code directly <style> inside of <head>
- Second method: link to the stylesheet file

- **Execute the python file**
  - Run main.py file
  - Click the link, which is given in terminal as following screen shot



```
 * Debugger is active!
 * Debugger PIN: 133-244-731
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production depl
oyment.
 * Running on http://192.168.1.249:5000/ (Press CTRL+C to quit)
```

Result

Everyone talks about Big Data and Business Intelligence. But in the end, what really matters are quality over quantity. You don't need big data, but rather smart data. For this reason, I wanted to scrap the most important headlines of important headlines from each website, which are presented on top of main page.

I got total 91 headlines on this date (29.01.2022) from two different online news websites CNN and BBC using my python program.

The number of headlines differs from times or dates, what time or which date the user get the result, because this is based on a dynamic URL, which is a URL returned through a query on a database-driven website or the URL of a website that runs some processing script such as Java Script.

Through this project, I learned that collecting data can serve an unlimited number of purposes than I expected, because I saw the potential what this technique can achieve in many different areas, since I can do pretty much what I wanted with web scraping. At next time, I would like to develop this idea more and to build argus that is fed from the data I extract. Although in this case, data processing might be a bit tricky since product references are not always the same.