**Web Scraping Guide**

0. **Before starting:** I recommend creating an environment for the project. All the dependencies and their versions are in the `requirements.txt` in the folder. These are the versions that worked as of February 7th, 2024. The .py files search_results.py and site_types.py must not be moved from the main folder.

Creating an environment:

```
python -m venv myenv
source myenv/bin/activate
pip install -r requirements.txt
```

The original code was restructured in order to save computing and cleaning time. It now runs by Mexican state (Media outlets from certain state + Search terms with municipalities from that state). It is mostly automated.

Basically the web scraping process consists of 3 stages.

1. **Preprocess:** Generating the necessary files for the web scraping. In this stage we pick the *media outlets* that are going to be scraped as well as the *search terms* for the scraping.

   There are two main Python notebooks for this.
   `web_scraping_per_state/code/00_preprocess/`
   - `00_sites_wordpress_v4_generate_per_state` -> Input media outlets from excel, formats them for web scraping code, gives an output per state
   - `00_terms_municipalities_per_state` -> Generates the .txt files with the municipalities + search terms needed for web scraping, outputs file per state

If you do not intend on running the code by state (just run all municipalities in all sites), you can use these notebooks:
- `00_terms_municipalities_simple` -> Generates the .txt file with the municipalities + search terms needed for web scraping, one file for all municipalities
- `00_sites_wordpress_v4_generate_simple` -> Formats list of media outlets for web scraping code

2. **Web-Scraping sample codes:** Scraping the web pages, we use the code developed by José Ramón Enríquez. Changes were made on the last part of the code regarding key words for breaks and time constraints.

   The following files are useful for familiarizing yourself with the code, as well as search_results.py and site_types.py. You can start by running tests with these files, before moving onto automation.

   `web_scraping_per_state/code/code_samples/`
   - `Codigo_WS_1.ipynb` -> Code provided by José Ramón
   - `Codigo_WS_8M_Marzo_2020.ipynb` -> Code used to web scrape news related to feminist protests, searching in all media outlets (key words + all municipalities)

2.1 **Web-Scraping automation:**
After generating the media outlet and term files per state (`00_sites_wordpress_v4_generate_per_state` and

`00_terms_municipalities_per_state` , respectively) you can automate the creation of the necessary notebooks as well as it execution following these steps:

A. Go to `Dropbox/web_scraping_per_state/code/web_scraping_template`
B. Adjust `./code/web_scraping_template/Codigo_WS_8M_template.ipynb` according to your needs. I recommend only changing the directory on the first cell code and the time constraints on the last cell code (there are some examples on how to do this hidden by #)
C. After saving point B., open `./code/web_scraping_template/automate_template.ipynb` , change paths to your directory, and run. It will create one Jupiter Notebook per state. You can limit the Notebooks you run each time.
D. Once you have the Notebooks, go to `./code/web_scraping_template/automate_run.ipynb` change paths and execute. This will create an error.txt file and 2 files for each state (i):
    1. `scrape_newsoutlets_per_state_CVE_ENT_i.csv` -> this file includes the hits from the web scraping
    2. `output_CVE_ENT_i.html` -> this file is basically a copy of the Notebook and its output

E. The estimated run time is …

**3. Cleaning**

To be added. I'm adjusting the previous code to the new structure