The project was divided into **3 fundamental modules**.
1. NCBI scrapping for species vs txid
2. AntWiki scrapping page/species and corresponding data across all 10 given cargo tables and sorting duplicate species with different data among them into a single species
3. Merging 1 & 2

For the same purposes**, 4 different python scripts** are attached.

For 1 --> 2 python scripts: scrape_ncbi.py & scrape_ncbi_01.py
The difference between them: scrape_ncbi.py is the raw script that scrapes the given NCBI website and gets all the species. But it is somewhat near the number 15k+, meaning this script requests to scrape 15k+ individual links one by one. Once completed, it writes a json file with the given 'file_name' (at the end of the script) in the current working directory. The downside is that: it takes approximately 6.5 hours to complete. Which is why it is prone to get error due to connection error (device network issue/website not responding/somehow device got off) resulting in losing all the data that it already scraped, so re-running this script means starting from the beginning which is time consuming and not efficient enough. Still, being the raw script, I added it if necessary.

To address this issue, added scrape_ncbi_01.py: this script generates 2 json files. First one's name is "ncbi_dataset_file" (at the beginning of main of the script). Whenever it runs, it checks for this file, then starts scraping the NCBI website. If some error is thrown, it stores the already scraped data into the same file in the current working directory. All it needs is, re-run the script, again. The script will read the file at the beginning, will get the stored data and starts scraping from where it stopped due to error. Often the NCBI website throws connection errors (4XX or 5XX errors), so if the script is stopped, re-run it each time. When it is completed, it sorts the whole dataset and writes another json file in the current working directory named “file_name” (at the end of the script).

Summary, after completion, both scripts should do the same thing: write a json file named “file_name” in the current working directory where data is sorted.

For 2 → scrape_antwiki.py
It scrapes for the 10 given cargo tables from the AntWiki website and writes a json file named “file_name” (at the end of the script) in the current working directory. Data is sorted.

For 3 → merging.py
This script takes in two json files from the same current working directory: i) the json file that was the output of scrape_ncbi.py or scrape_ncbi_01.py ii) the json file that was the output of scrape_antwiki.py. The names of the files must match. Otherwise, it would throw an error saying “No such file not found”. Then, it merges the NCBI dataset with AntWiki dataset, basically adding the txid to AntWiki dataset on the basis of species/page name, if no match found, the NCBI data is appended into the AntWiki dataset. Finally, the Merged AntWiki dataset will be written in a json file in the current working directory named

“file_name” (at the end of the script) and a report of the merger will be given in the console.

Characteristics of the final merged dataset:

1. It has one single key value pair. The key is “antwiki” and the value is an array of different species/pages.
2. Each page is a key value pair itself. The keys are the fields coming from the different tables and the values are corresponding values of those fields.
3. If the same species/page name came multiple times from the same cargo table, there would be one single key value pair for the page name and the multiple values for each field will be stored in arrays. So just getting the length of these arrays should provide the idea of how many times the same page name came up in the same cargo table.
4. The different field names from different tables are renamed by default. Like for ‘Notes’ for table “A”, the field name gets changed into “A_Notes”. So, there is absolutely no chance, loss/misplacement of data due to the same field name across two given tables.
5. Page name with no NCBI match, has “txid” with value of 0 by default.

Notes:

1. The json files should be in the same place as the scripts as they work on the current working directory. So just create a new folder, and run the scripts from there. That will do just fine.
2. I used an additional OS module to write and read json files from the device which is applicable for Windows operating systems only (probably).
3. If the client wants, for additional prices, I can create exe format and batch files of these scripts to run/rerun from command prompt easily.
4. Attached all the corresponding json files that were generated in the last test run. Used scrape_ncbi_01.py.
5. Feel free to ask questions if facing any confusion.

Thanks for the project, it has been a wonderful experience for me. **Appreciate it, a lot.**