**Steps for setting up the project.**

1. Create a new folder and copy all the project files within the same directory.
2. Download Apache Spark from <https://spark.apache.org/downloads.html> and extract it in the folder where you want to set up the installation.
3. Download PostgreSQL JDBC driver for JAVA 8 from <https://jdbc.postgresql.org/download/> and copy it in the **jars** folder inside the spark installation directory created in step 2.
4. Create a virtual environment .venv within that directory.
5. Run the **Data-tables.ipynb** for setting up the PostgreSQL Database tables and insert data into reference tables used in the project.Make sure to update the DB credentials required to create tables in PostgreSQL accordingly as per your installation.
6. Update the directory path in the **app.py** script according to the location of your local machine for the spark-submit command and the location for the JDBC driver.
7. Similarly update the directory path for the jar and other script locations in **final.py.** Additionally, update the DB credentials required to the DF data write to PostgreSQL accordingly as per your installation.
8. Set up the cron jobs for the scheduled execution of the data pipeline using below commands. (Note: update the path as per your local machine’s location).

0 9 \* \* \* /Library/Frameworks/Python.framework/Versions/3.10/bin/python3 /Users/sakshimehta/Documents/Coursework/Assignments/Managing\_Data/Assignments/Final\_Project/app.py >> /Users/sakshimehta/Documents/Coursework/Assignments/Managing\_Data/Assignments/Final\_Project/logs/log\_`date "+%Y-%m-%d"`.log 2>&1

10 9 \* \* \* curl <http://127.0.0.1:5001/load>

The above cron will publish the API and then kick-off the spark ETL job using the flask API. Replace the URL above in the curl command with your setup.

**Flask App**

Copy the **interactive\_flask\_app.ipynb** inside the code folder created above and create a new folder named templates within the same code directory and copy the **index\_0.html** file inside the templates folder.

The app is designed to take restaurant names e.g. “DUN HUANG” or “SHAKE SHACK”. The input is not case sensitive.

It is only necessary to input a restaurant name into the app before clicking “Add markers” to show which branches of the restaurant in New York have recorded violations. For example, you could enter DOMINO’S in the name field and leave the other fields blank. This will show results for all branches of Domino’s in New York. By specifying zip code or borough, results shown on the map can be limited to branches within the specified zip code or borough.

**Please Note: We have attached final.ipynb and app.ipynb as a part of the project submission. But both these files were converted to python scripts (final.py and app.py) for setting up the cron jobs.**