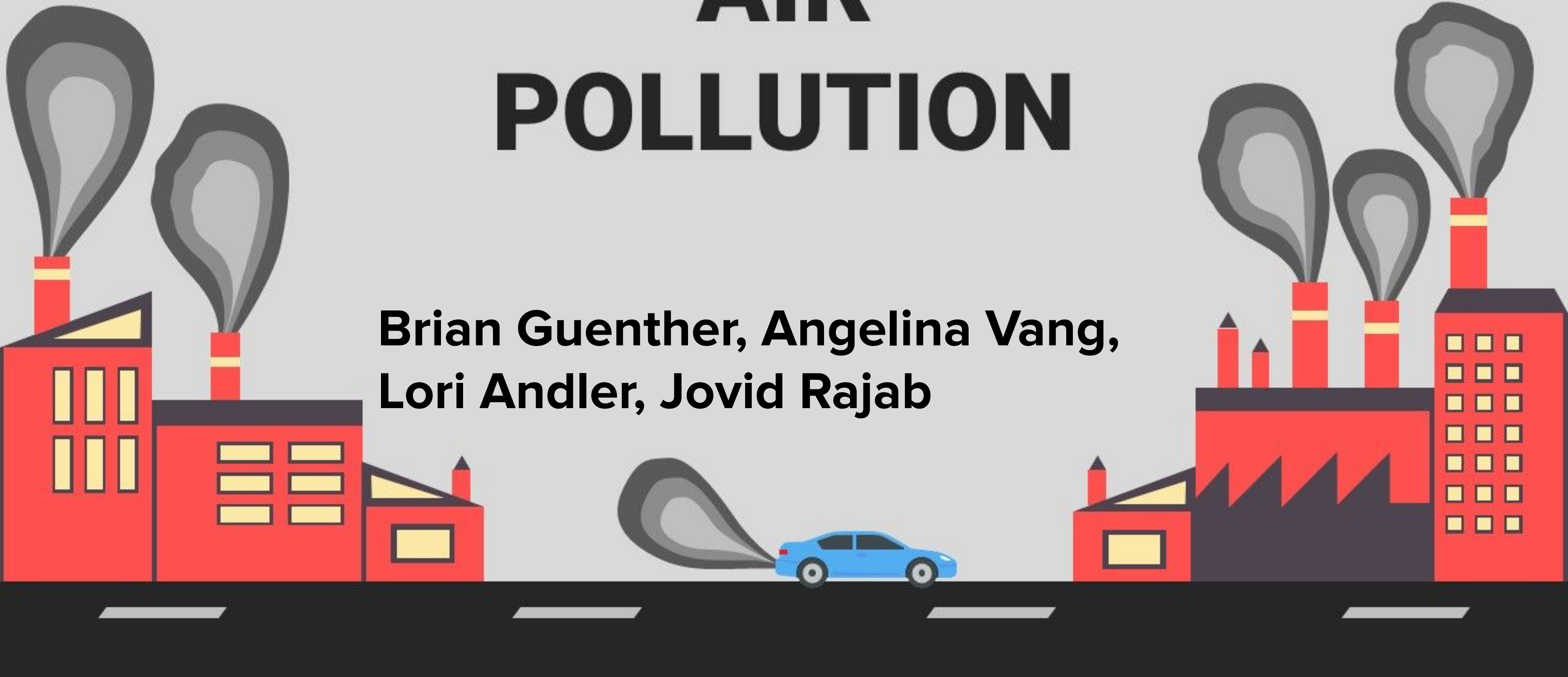
AIR POLLUTION

Brian Guenther, Angelina Vang, Lori Andler, Jovid Rajab



Pollution Data and Information

Start of Project

- Kaggle dataset: US Pollution 2000-2021 that contains a dataset that used data from the US EPA on four major gas pollutants: Carbon Monoxide, Nitrogen Dioxide, Ground-Level Ozone, and Suflur Dioxide leverl in the years 2000-2021.

- EPA API: United States Environmental Protection Agency API on Air Quality. We pulled out data for four more pollutants: Benzene, Formaldehyde, 1,3 - Butadiene, and Lead

Six criteria air pollutants:

Ground-level Ozone (O3)
Particulate matter (PM10 and PM2.5)
Carbon monoxide (CO)
Nitrogen dioxide (NO2)
Sulfur dioxide (SO2)
Lead (Pb)

& Hazardous Air Pollutants (HAPs) - 188
Volatile organic compounds (VOCs) ~126

Concerns:

Chemically reactive
Asthma Heart Disease Chromosomal damage
Infection Cancer Central Nervous System
Developmental
Disproportionately impact certain groups of people with greater severity than others
Causes 200,000 early deaths each year

Data and Project Progression

EPA website
https://aqs.epa.gov/aqsweb/airdata/download_files.html

43 different categories of files for each year
2000 to 2022

Many uncompress to ~ 100 MB

Initial Selection: 8 compounds for four states
Ozone (O3), Carbon monoxide (CO), Nitrogen dioxide (NO2), Sulfur dioxide (SO2), Lead (Pb), Formaldehyde, Benzene, 1,3-Butadiene
California, Minnesota, New York, and Texas

Seasonal changes and maximal levels

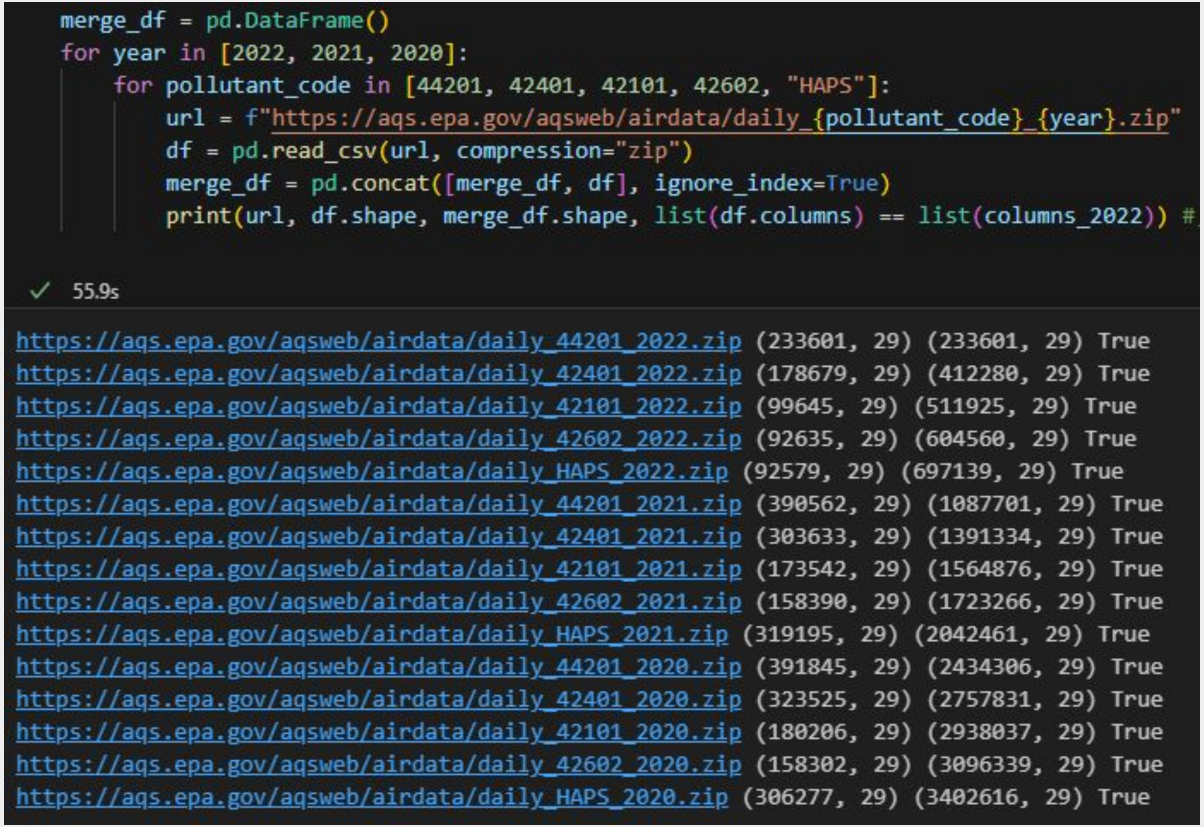
Initial Effort

Download files, characterize columns, identify desired data for analysis

We ran into too much data and size issues (100 MB Github)

Less of an issue when webscraping

Data Cleaning



```
merge_df = pd.DataFrame()
for year in [2022, 2021, 2020]:
    for pollutant_code in [44201, 42401, 42101, 42602, "HAPS"]:
        url = f"https://aqs.epa.gov/aqsweb/airdata/daily_{pollutant_code}_{year}.zip"
        df = pd.read_csv(url, compression="zip")
        merge_df = pd.concat([merge_df, df], ignore_index=True)
        print(url, df.shape, merge_df.shape, list(df.columns) == list(columns_2022)) #
```

✓ 55.9s

```
https://aqs.epa.gov/aqsweb/airdata/daily_44201_2022.zip (233601, 29) (233601, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42401_2022.zip (178679, 29) (412280, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42101_2022.zip (99645, 29) (511925, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42602_2022.zip (92635, 29) (604560, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_HAPS_2022.zip (92579, 29) (697139, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_44201_2021.zip (390562, 29) (1087701, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42401_2021.zip (303633, 29) (1391334, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42101_2021.zip (173542, 29) (1564876, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42602_2021.zip (158390, 29) (1723266, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_HAPS_2021.zip (319195, 29) (2042461, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_44201_2020.zip (391845, 29) (2434306, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42401_2020.zip (323525, 29) (2757831, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42101_2020.zip (180206, 29) (2938037, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_42602_2020.zip (158302, 29) (3096339, 29) True
https://aqs.epa.gov/aqsweb/airdata/daily_HAPS_2020.zip (306277, 29) (3402616, 29) True
```

Takes 40 to 60
seconds to scrape

3 years of data
3402626 rows
29 columns

Filtering (<1 second)
8 compounds
4 states

748021 rows
13 columns

(a 92 MB csv)

Learning Curve

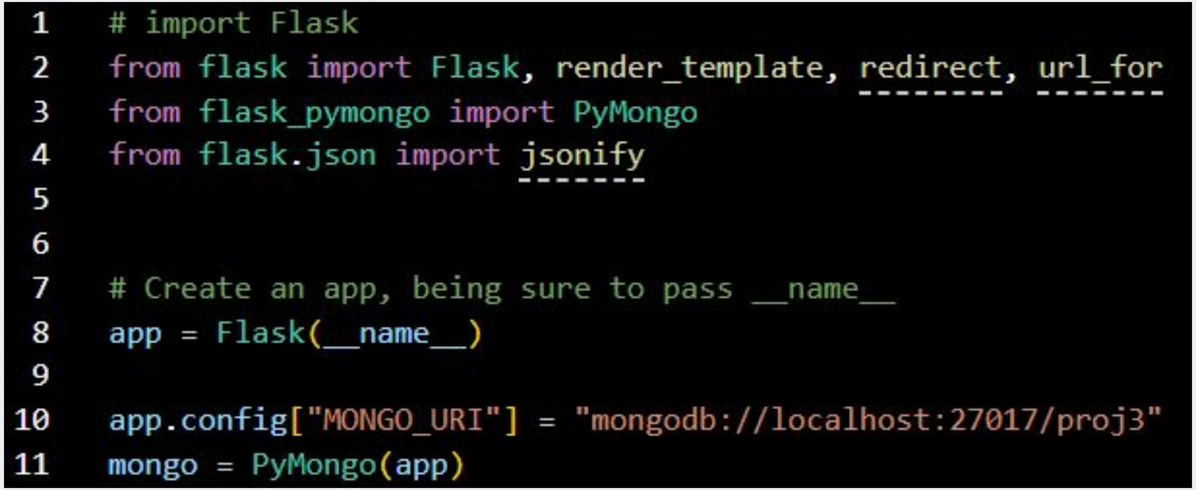
Utilized PyMongo to investigate features of the DB or Flask output

Further reduction of data set for process of getting features working lead to a current data set of four pollutants (Ozone (O3), Carbon monoxide (CO), Nitrogen dioxide (NO2), Sulfur dioxide (SO2) in the same four states for the year 2022.

Filtered csv contains 145698 rows and 13 columns.

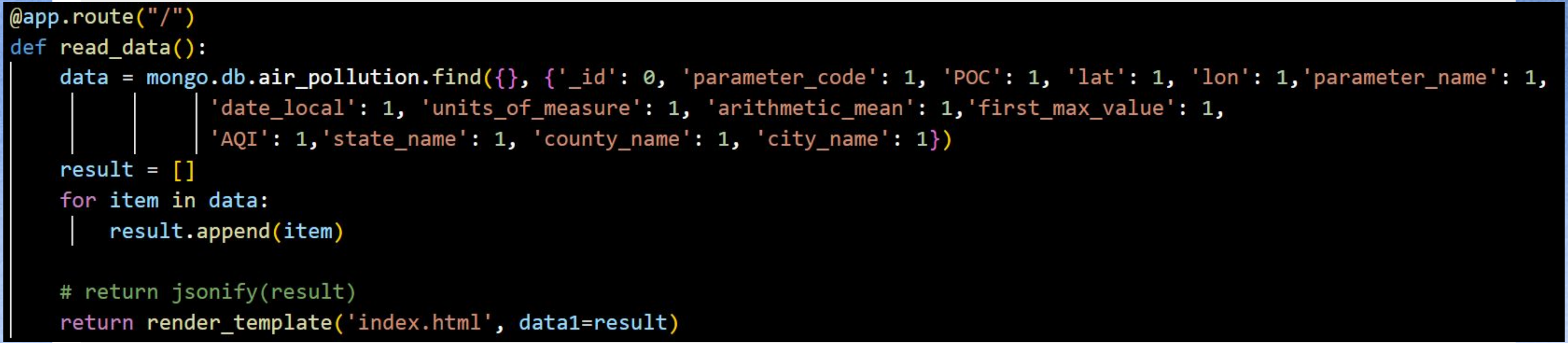
Benefit of teammates example: lon vs. lng for longitude (catching typos)
Creation of MongoDB and writing out a JSON (~250 MB)

Data from MongoDB > Flask app > HTML > JS



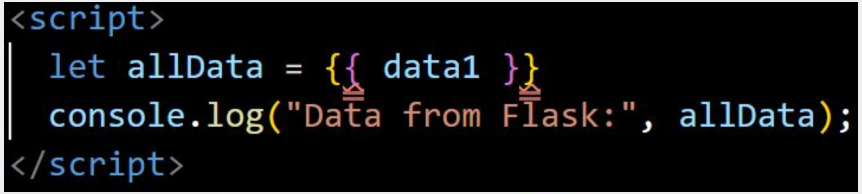
```python
# import Flask
from flask import Flask, render_template, redirect, url_for
from flask_pymongo import PyMongo
from flask.json import jsonify


# Create an app, being sure to pass __name__
app = Flask(__name__)

app.config["MONGO_URI"] = "mongodb://localhost:27017/proj3"
mongo = PyMongo(app)
```

Data from MongoDB > Flask app > HTML > JS

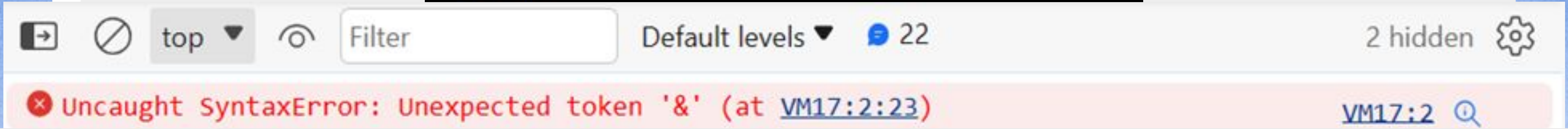


```
@app.route("/")
def read_data():
    data = mongo.db.air_pollution.find({}, {'_id': 0, 'parameter_code': 1, 'POC': 1, 'lat': 1, 'lon': 1,'parameter_name': 1,
                'date_local': 1, 'units_of_measure': 1, 'arithmetic_mean': 1,'first_max_value': 1,
                'AQI': 1,'state_name': 1, 'county_name': 1, 'city_name': 1})
    result = []
    for item in data:
        result.append(item)

    # return jsonify(result)
    return render_template('index.html', data1=result)
```

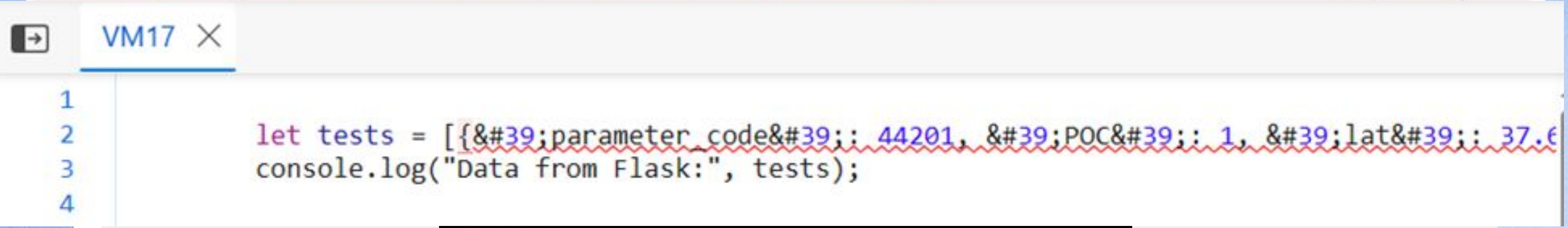
Data from MongoDB > Flask app > HTML > JS



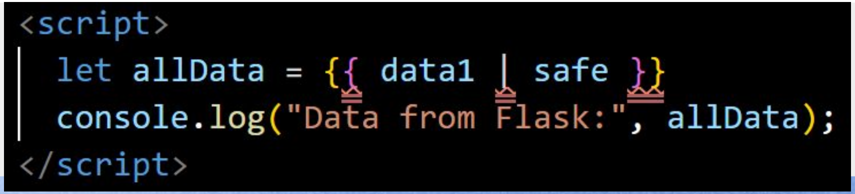
```
<script>
  let allData = {{ data1 }}
  console.log("Data from Flask:", allData);
</script>
```



```
Uncaught SyntaxError: Unexpected token '&' (at VM17:2:23)
```

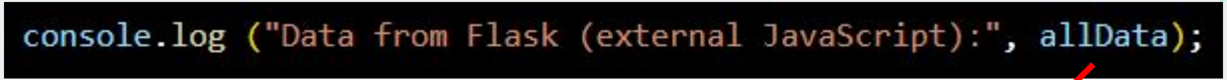


```

    let tests = [{&#39;parameter_code&#39;: 44201, &#39;POC&#39;: 1, &#39;lat&#39;: 37.6
    console.log("Data from Flask:", tests);

```

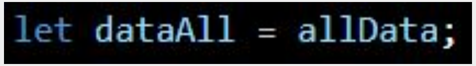


```
<script>
  let allData = {{ data1 | safe }}
  console.log("Data from Flask:", allData);
</script>
```

Data from MongoDB > Flask app > HTML > JS

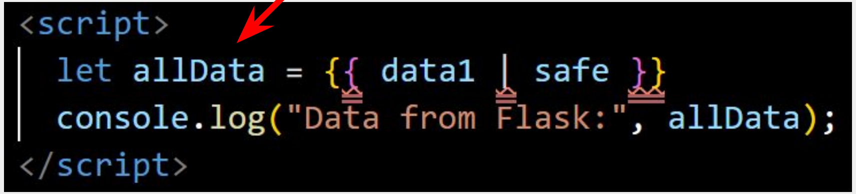


```
console.log ("Data from Flask (external JavaScript):", allData);
```



```
let dataAll = allData;
```

Reminder of our script tag in our HTML code:



```
<script>
  let allData = {{ data1 | safe }}
  console.log("Data from Flask:", allData);
</script>
```

Webpage Demo

If we had more time…

Some of the items we would’ve liked to have expanded upon were:

- Being able to map out more pollutants and/or more states
- Map out over time to see a trend over years
- Add a horizontal bar to the charts to show when the levels are above the “danger levels”
- Change markers so radius changes with pollutant concentration value
- Try to map out whether sites are in urban or rural areas and compare the differences

Thank you!

Thank you to the tutors, instructors and TA's that assisted us in producing our Project 3:

- Kourt Bailey
- Limei Hou
- Steven Thomas
- Hunter Hollis
- Sam & Randy