

Covid Criminals

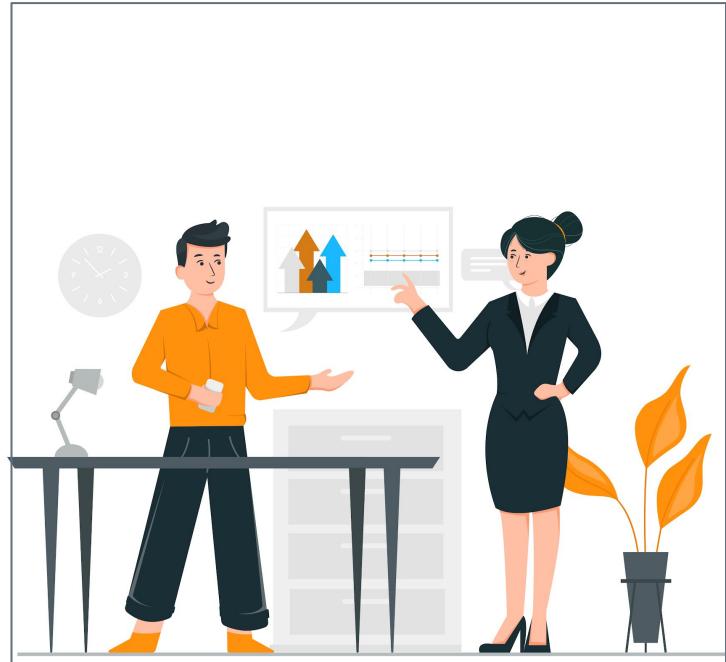
PROJECT

Covid Criminals Team
Presents



Welcome

Let's visualize the relationship
between COVID-19 cases and
criminal activities in New York
City!



The project workflow in one glance



Datasets

The Sources



- All datasets are obtained from NYC open data sources.
- All data were available in CSV and JSON formats.

NYC Covid-19 Dataset Link:

<https://www1.nyc.gov/site/doh/covid/covid-19-data-boroughs.page>

NYC Crimes Dataset Link:

<https://data.cityofnewyork.us/Social-Services/NYPD/fjn5-bxwg>

COVID-19: Data  OpenData



Why was it difficult?

It was challenging to find appropriate datasets that included all elements required such as Latitude and Longitude.

Jupyter

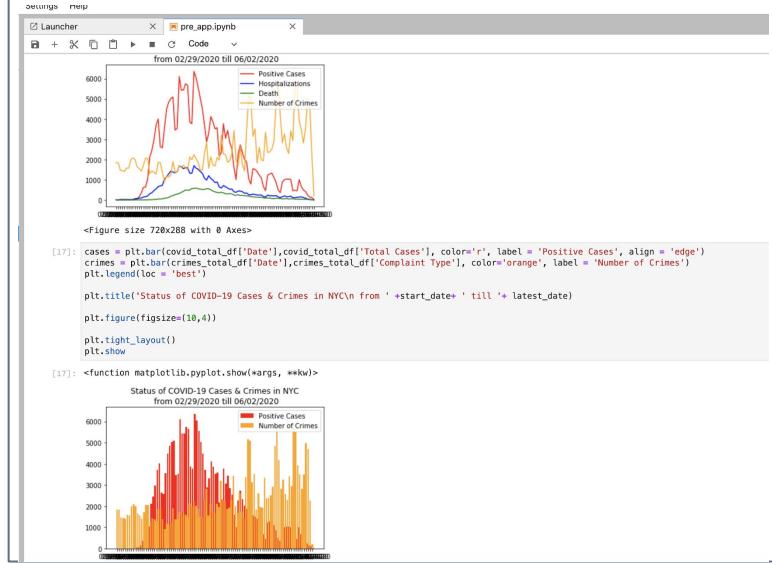
Preliminary Analysis

Level of Difficulty



What was done?

- Read CSV files.
- Removed unnecessary data.
- Merged rows, tables and columns.
- Grouped and aggregated.
- Preliminary plotting.



Why?

- Ensure our datasets are informative
- Preliminary clean up to plan ETL
- Initial plotting for early study

Design

Front-end planning

Level of Difficulty



What was done?

- Based on our datasets and plots, the expected front-end was sketched in Adobe Photoshop.

Why?

- Simplifying the output requirements, providing a layout for the wireframe of the HTML
- To have clear roadmap for expected results



Jupyter

ETL



What was done?

- Jupyter app was created to extract, transform and load data into PostgreSQL.

Why?

- Data was required to be fed into SQL to further serve the front end application.

The screenshot shows a Jupyter Notebook cell with the title "Dependencies". It contains Python code for importing various libraries such as pandas, matplotlib, sqlalchemy, requests, time, datetime, scipy.stats, and weather_api_key. Below this, there's a section titled "DataFrames Related to COVID-19" with code for reading CSV files for COVID-19 cases, hospitalizations, death, and NYC boroughs. It also includes code for connecting to a PostgreSQL database, creating engines, and writing final DataFrames to the database. The code uses f-strings and various pandas methods like to_sql and map.

```
import pandas as pd
import matplotlib.pyplot as plt
from sqlalchemy import create_engine
from config import password

import requests
import time
from datetime import datetime

from scipy.stats import linregress

# Import API key
from config import weather_api_key

# Reading csv files for COVID-19(Cases, Hospitalizations, Death) & NYC Boroughs
bronx_covid_data = "data/covid/Bronx-data-GHyLp.csv"

connection_string = f"postgres:{password}@localhost:5432/NYC_COVID19_CRIMES_DB"
engine = create_engine(f'postgresql://{{connection_string}}')

# Final DF for NYC-COVID-19
covid_final_df.to_sql(name='covid', con=engine, if_exists='replace', index=True)

# Final DF for NYC-CRIMES
nypd_final_df.to_sql(name='crime', con=engine, if_exists='replace', index=True)

# Final DF for total info map
total_info_map.to_sql(name='summary', con=engine, if_exists='replace', index=True)
```

Why was it difficult?

- DataFrames were changed multiple times to obtain correct data structure and aggregated values.
- Indexing for loading to SQL.

PostgreSQL

ETL

Level of Difficulty



What was done?

- Database was created.
- Data was loaded into SQL.
- Ran scripts to assign primary keys.

Why?

- To provide data to the Flask App.



```
1 /* Adding Index as PK for Tables */
2 ALTER TABLE covid ADD PRIMARY KEY (index);
3 ALTER TABLE crime ADD PRIMARY KEY (index);
4 ALTER TABLE summary ADD PRIMARY KEY (index);
5
6 /* Checking each table to be sure every thing looks good */
7 SELECT * FROM covid;
8 SELECT * FROM crime;
9 SELECT * FROM summary;
10
```

Data Output Notifications Explain Messages

ALTER TABLE

Query returned successfully in 251 msec.

Flask

Flask App

Level of Difficulty

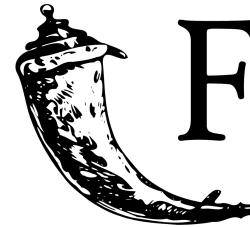


What was done?

- Googling and searching stack overflow to connect to Postgres
- Created API routes
- Provided JSON output data

Why?

Provide a bridge between SQL data and the front-end. Flask was the data server.



Flask

web development,
one drop at a time

```
app.py > x
UT-TOR-DATA-PT-01-2020-U-C > Project 02 > git > Project_2 > app.py > ...
1 import os
2 from flask import Flask, render_template, jsonify
3 from sqlalchemy import create_engine
4 import numpy as np
5 import sqlalchemy
6 from sqlalchemy.ext.automap import automap_base
7 from sqlalchemy.orm import Session
8 from sqlalchemy import create_engine, func
9 from config import password
10 from flask_cors import CORS
11
12
13 #Key Things, added a few line to jupyter lab
14 #in SQL everyone has to run the following to add primary key
15 #ALTER TABLE covid ADD PRIMARY KEY (index);
16 #ALTER TABLE crime ADD PRIMARY KEY (index);
17
18 ##### Navid added for Mapping
19 config = {
20     'ORIGINS': [
21         'http://localhost:5000', # React
22         'http://127.0.0.1:5000', # React
23     ],
24
25     'SECRET_KEY': '....'
26 }
```

Why was it difficult?

Multiple API routes were created due to the large size of data and filtering requirements

Front-end

Java Script, HTML, CSS



What was done?

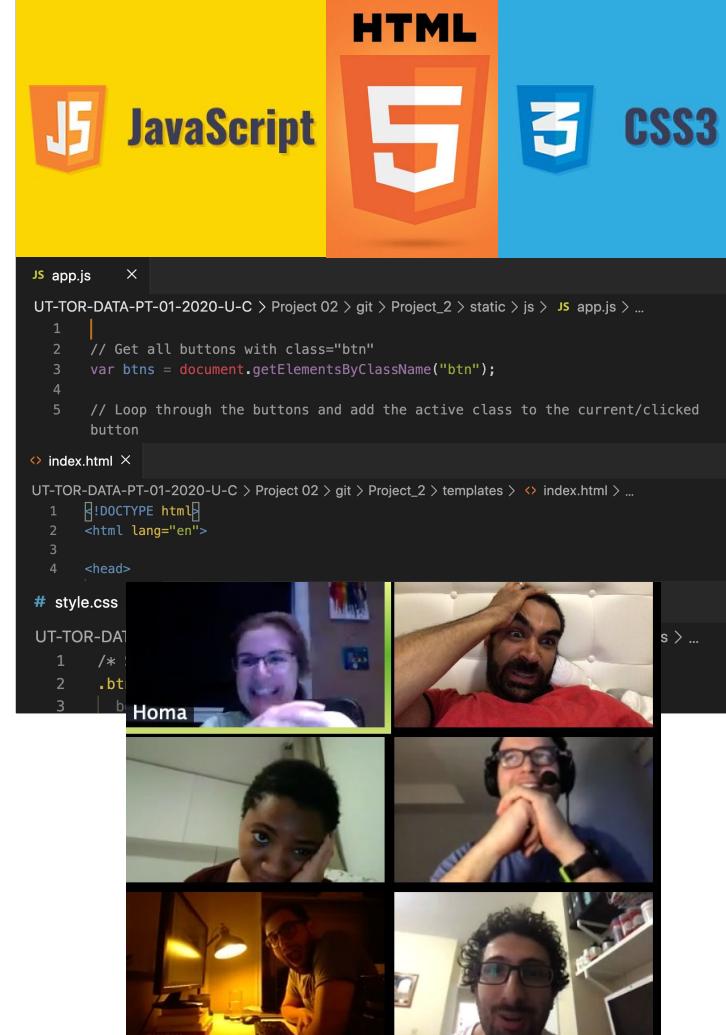
- All JSON data called from Flask API using D3
- Plots, charts, maps and dropdowns created
- Leaflet, plotly, Geojson libraries utilized
- Finalized styling via CSS

Why?

Dan said so!

Why was it difficult?

Beyond words!



Extra JS Library

Front end

CHART.JS



<https://chartjs.org>

Level of Difficulty



What was done?

- Polar Area chart created to show proportions of COVID cases with a scale of value for context

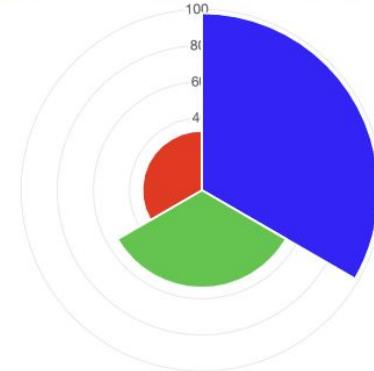
Why?

- Created to explore the functionalities of another JS library



Chart.js

Positive Cases Hospitalizations Deaths



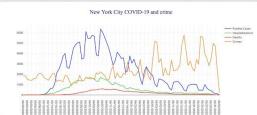
Why was it difficult?

We had trouble deleting the initial data layer and updating the chart with new data



Covid-19 & Crime in New York City:

This page is a visualization of Covid-19 correlation with crime in New York City from February 2020 and the present time.



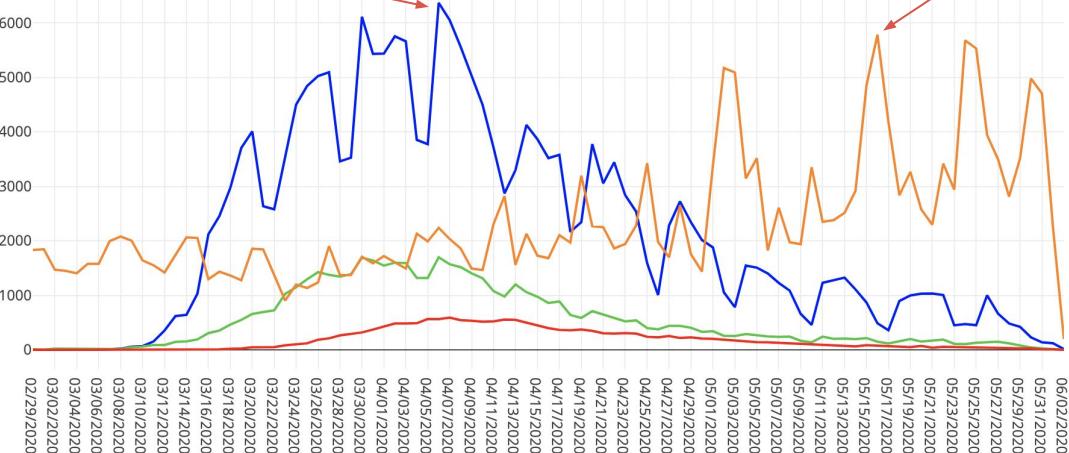
Final Application



High Covid-19
Low Crime

New York City COVID-19 and crime

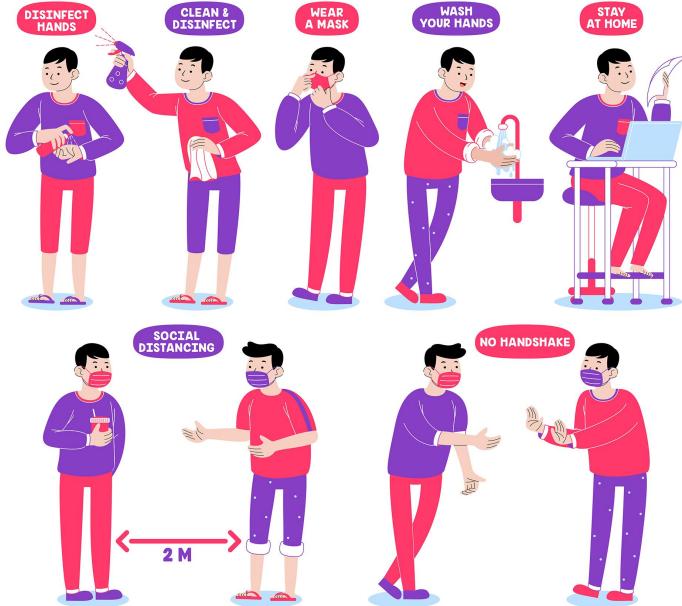
High Crime
Low Covid-19



Let's see the project

Discussion Panel

CORONAVIRUS PREVENTION



Thank you!