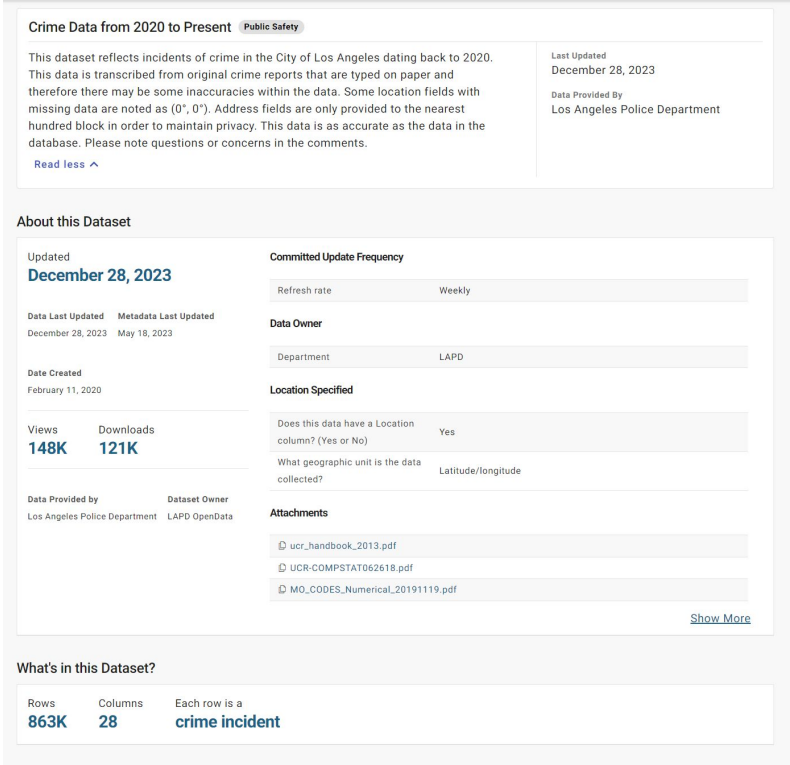
LOS ANGELES CRIME DATA ANALYTICS DASHBOARD

Los Angeles Crime Dataset: What is it?

- https://data.lacity.org/
- Los Angeles Police department
- Over 850k rows of crime incidents
- Updated weekly (2020 - 2023)



Los Angeles Crime Dataset: Target Audience

- LA Police Department
  - Hotspots
  - Trends
- Policy Makers
  - Preventative measures
  - Measure a policy's effectiveness
- LA Citizens
  - Aware of surroundings
  - Steps to ensure their safety

Los Angeles Crime Dataset: Extraction

- Using sodapy and pandas to extract raw data
- Results was then converted into a pandas dataframe for cleaning

```
import pandas as pd
from sodapy import Socrata

# Get my LA City App Token from the api_keys module file
from api_keys import lacity_app_token


# Reference: https://data.lacity.org/Public-Safety/Crime-Data-from-2020-to-Present/2nrs-mtv8/about_data
client = Socrata("data.lacity.org", lacity_app_token)

# Approx: 7-8 Minutes
results = client.get_all("2nrs-mtv8")

# Convert to pandas DataFrame
crime_df = pd.DataFrame.from_records(results)

crime_df
```

Los Angeles Crime Dataset: Extraction

- Experimented with using JSON api
- Extraction approximately took 7-8 mins to download

create_db_json.py > ...

```python
import requests
import json
import pandas as pd
from pathlib import Path
from datetime import datetime

# Define the base URL and initial parameters
base_url = "https://data.lacity.org/resource/2nrs-mtv8.json"
params = {
    "$limit": 50000,  # Number of records per request
    "$offset": 0     # Initial offset
}

#print url
print(base_url, params["$limit"])

#create empty list
all_records = []

while True:
    response = requests.get(base_url, params=params)

    # Check if the response is successful
    if response.status_code == 200:
        data = response.json()
```

Los Angeles Crime Data: Raw Dataset (Pandas DataFrame)

	dr_no	date_rptd	date_occ	time_occ	area	area_name	rpt_dist_no	part_1_2	crm_cd	crm_cd_desc	...	status	status_desc	crm_cd_1	location	lat	lon	crm_cd_2	cross_street	crm_cd_3	crm_cd_4
0	010304468	2020-01-08T00:00:00.000	2020-01-08T00:00:00.000	2230	03	Southwest	0377	2	624	BATTERY - SIMPLE ASSAULT	...	AO	Adult Other	624	1100 W 39TH PL	34.0141	-118.2978	NaN	NaN	NaN	NaN
1	190101086	2020-01-02T00:00:00.000	2020-01-01T00:00:00.000	0330	01	Central	0163	2	624	BATTERY - SIMPLE ASSAULT	...	IC	Invest Cont	624	700 S HILL ST	34.0459	-118.2545	NaN	NaN	NaN	NaN
2	200110444	2020-04-14T00:00:00.000	2020-02-13T00:00:00.000	1200	01	Central	0155	2	845	SEX OFFENDER REGISTRANT OUT OF COMPLIANCE	...	AA	Adult Arrest	845	200 E 6TH ST	34.0448	-118.2474	NaN	NaN	NaN	NaN
3	191501505	2020-01-01T00:00:00.000	2020-01-01T00:00:00.000	1730	15	N Hollywood	1543	2	745	VANDALISM - MISDEAMEANOR ($399 OR UNDER)	...	IC	Invest Cont	745	5400 CORTEEN PL	34.1685	-118.4019	998	NaN	NaN	NaN
4	191921269	2020-01-01T00:00:00.000	2020-01-01T00:00:00.000	0415	19	Mission	1998	2	740	VANDALISM - FELONY ($400 & OVER, ALL CHURCH VA...	...	IC	Invest Cont	740	14400 TITUS ST	34.2198	-118.4468	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
857653	231606525	2023-03-22T00:00:00.000	2023-03-22T00:00:00.000	1000	16	Foothill	1602	1	230	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT	...	IC	Invest Cont	230	12800 FILMORE ST	34.279	-118.4116	NaN	NaN	NaN	NaN
857654	231210064	2023-04-12T00:00:00.000	2023-04-12T00:00:00.000	1630	12	77th Street	1239	1	230	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT	...	IC	Invest Cont	230	6100 S VERMONT AV	33.9841	-118.2915	NaN	NaN	NaN	NaN
857655	230115220	2023-07-02T00:00:00.000	2023-07-01T00:00:00.000	0001	01	Central	0154	1	352	PICKPOCKET	...	IC	Invest Cont	352	500 S MAIN ST	34.0467	-118.2485	NaN	NaN	NaN	NaN
857656	230906458	2023-03-05T00:00:00.000	2023-03-05T00:00:00.000	0900	09	Van Nuys	0914	2	745	VANDALISM - MISDEAMEANOR ($399 OR UNDER)	...	IC	Invest Cont	745	14500 HARTLAND ST	34.1951	-118.4487	NaN	NaN	NaN	NaN
857657	230319786	2023-11-10T00:00:00.000	2023-11-09T00:00:00.000	2300	03	Southwest	0395	1	331	THEFT FROM MOTOR VEHICLE - GRAND ($950.01 AND ...	...	IC	Invest Cont	331	4100 S HOBART BL	34.0091	-118.3078	NaN	NaN	NaN	NaN

857658 rows × 28 columns

Los Angeles Crime Data: Cleaning The Dataset

- Jupyter Notebook
- Convert to the correct data types
- Removed unnecessary columns
- Replaced Codes to meaningful data (Eg: M = Male)
- Create new summarised Crime Category column

```
import numpy as np

# Sex Code to Actual:
# M - Male
# F - Female
# X - Unknown
replace_sex_code = {"M": "Male",
                    "F": "Female",
                    "X": "Unknown",
                    "-": None}

final_crime_df = final_crime_df[~final_crime_df["vict_sex"].isin(["H"])]
final_crime_df["vict_sex"] = final_crime_df["vict_sex"].replace({np.nan: None})
final_crime_df["vict_sex"] = final_crime_df["vict_sex"].replace(replace_sex_code)



unique_victim_sex = final_crime_df["vict_sex"].unique()

print("Unique Victim Sex:")
print()
print(unique_victim_sex)
```

```
Unique Victim Sex:

['Female' 'Male' 'Unknown' None]
```

Los Angeles Crime Data: Adding 'Crime Categories'

UC

UCR REPORTING – Return A
(Based on date of reporting)

Part I – Violent Crimes

HOMICIDE	110 (Homicide)
	113 (Manslaughter)
RAPE	121 (Rape)
	122 (Attempted Rape)
	815 (Sexual Penetration w/ Foreign Object)
	820 (Oral Copulation)
	821 (Sodomy)
ROBBERY	210 (Robbery)
	220 (Robbery - attempted)
AGG. ASSAULTS	230 (ADW)
	231 (ADW against LAPD Police Officer)
	235 (Child beating)
DV*	236 (Spousal beating)
	250 (Shots Fired)
	251 (Shots fired inhabited dwelling)
	761 (Brandishing)
	926 (Train Wrecking)

```python
# Rape
rape_codes = [121, 122, 815, 820, 821]

# Robbery
robbery_codes = [210, 220]

# Aggravated Assault
agg_assault_codes = [230, 231, 235]

# Domestic Violence
domestic_violence_codes = [236, 250, 251, 761, 926,
                           626, 627, 647, 763, 928, 930]

# Simple Assault
simple_assault_codes = [435, 436, 437, 622, 623, 624, 625]

# Burglary
burglary_codes = [310, 320]

# Grand Theft Auto (Motor Vehicle Theft)
gta_codes = [510, 520, 433]

# Burglary or Theft from Vehicle
btfv_codes = [330, 331, 410, 420, 421]

# Personal Theft
personal_theft_codes = [350, 351, 352, 353, 450, 451, 452, 453]

# Other Theft
other_theft_codes = [341, 343, 345, 440, 441, 442, 443, 444, 445,
                     470, 471, 472, 473, 474, 475, 480, 485, 487, 491]

ucr_dict = {"Homicide": homicide_codes,
            "Rape": rape_codes,
            "Robbery": robbery_codes,
            "Aggravated Assault": agg_assault_codes,
            "Domestic Violence": domestic_violence_codes,
            "Simple Assault": simple_assault_codes,
            "Burglary": burglary_codes,
            "Grand Theft Auto": gta_codes,
            "Burglary or Theft from Vehicle": btfv_codes,
            "Personal Theft": personal_theft_codes,
            "Other Theft": other_theft_codes}
```

Los Angeles Crime Data: Final Dataset

Reduced to:

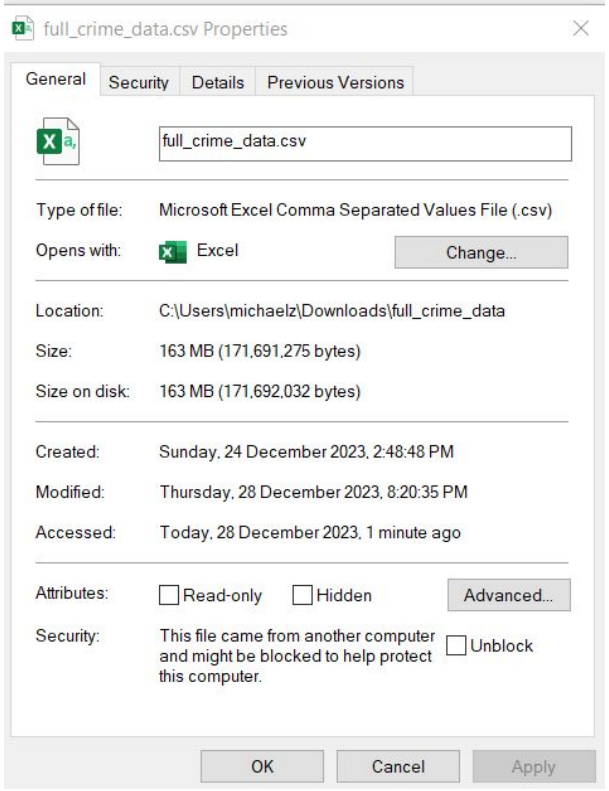
- 16 Columns
- 800k records
- Converted to appropriate data types
- New Crime Category field
- Translated:
- Descent Code
- Sex Code

```
dr_no
date_rptd
date_occ
time_occ
area_name
crime_category
crm_cd
crm_cd_desc
vict_age
vict_sex
vict_descent
premis_desc
location
cross_street
lat
lon
```

Los Angeles Crime Data: Final Dataset

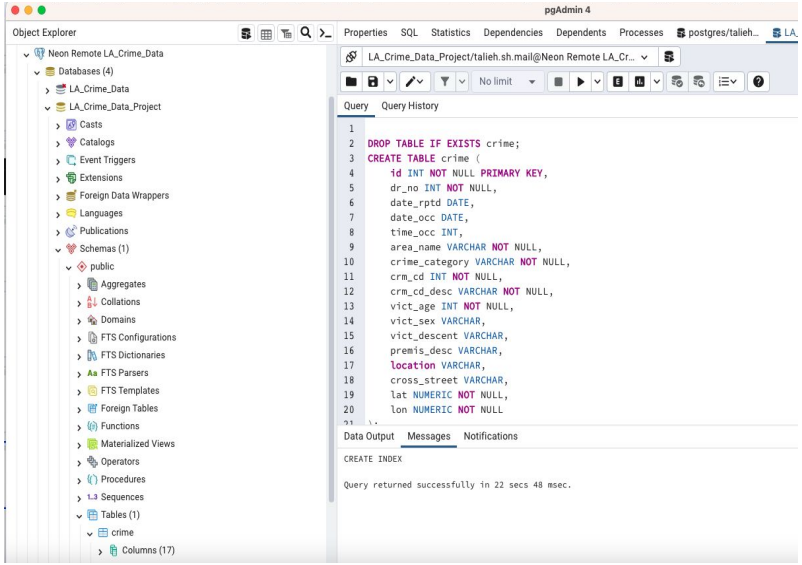
Original raw csv file size: 250MB

Once cleaned: **163MB**

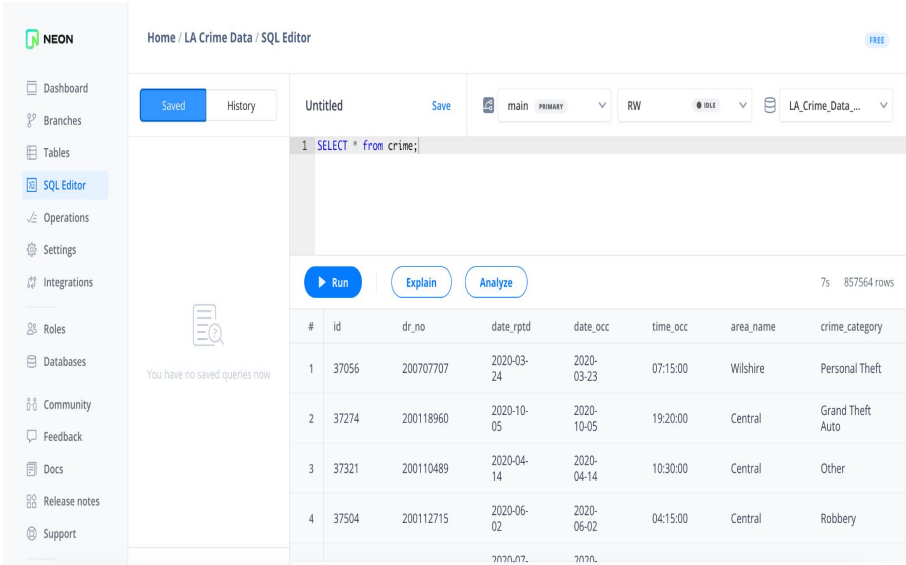


PostgreSQL Database: Creation & Cloud Hosting

pgAdmin

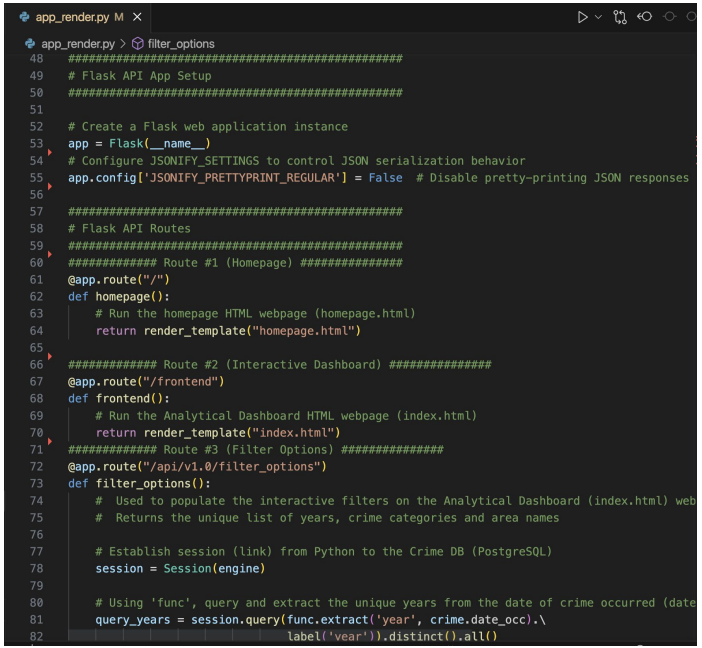


NEON

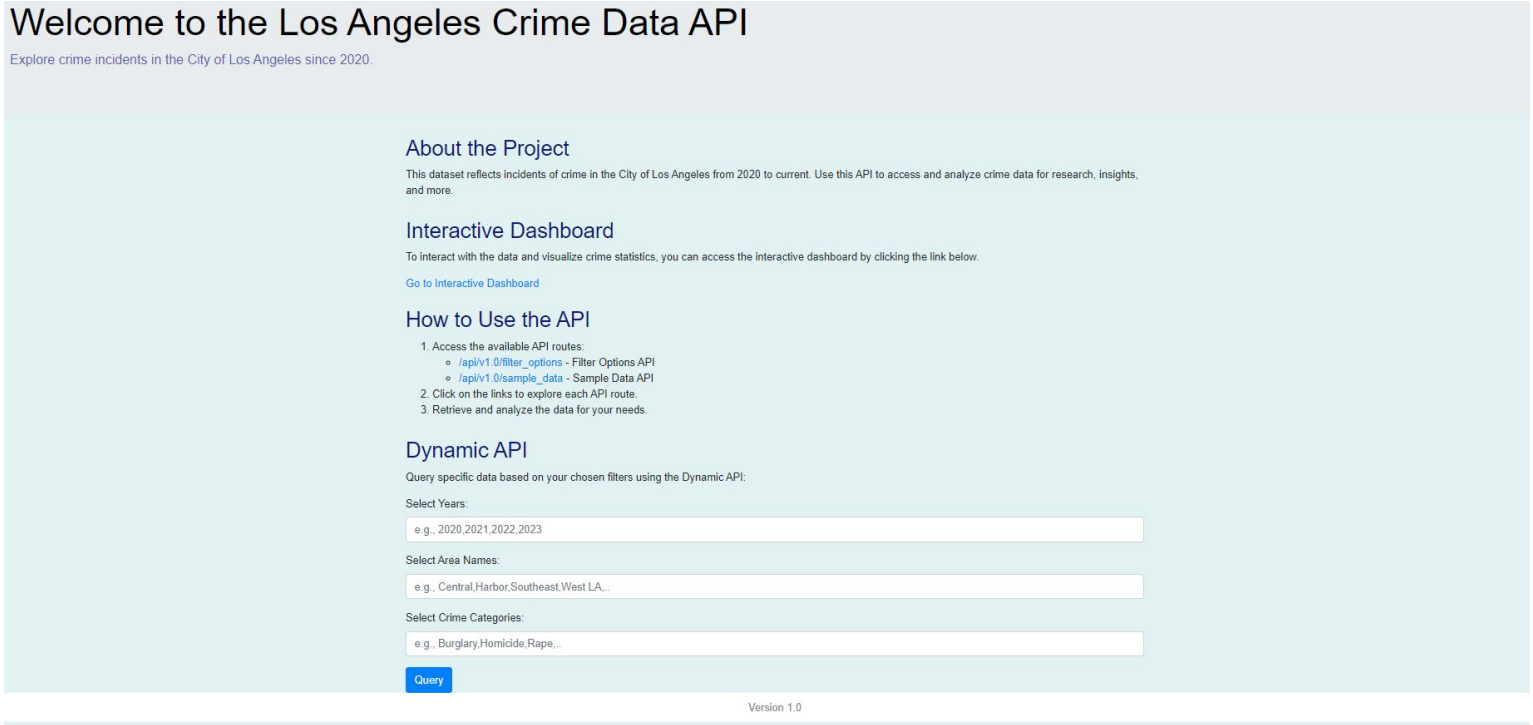


Flask Web Application: Crime Dataset API

- SQL Database Connection
- Data Querying and Response
- Route Definitions:
  - Homepage
  - Interactive Dashboard
  - API Endpoints



Flask Web Application:
Crime Dataset API Landing Page

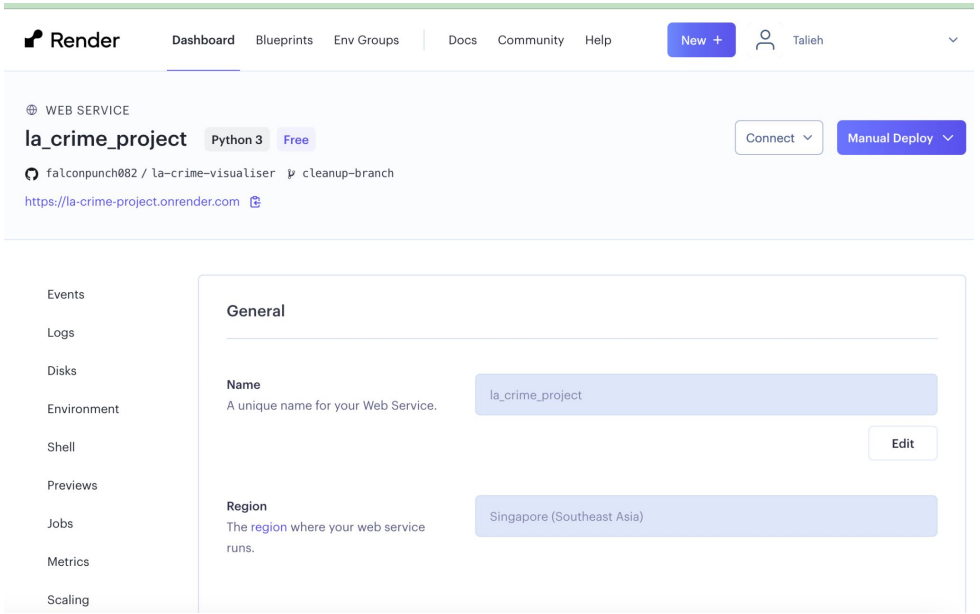


Flask Application:
Cloud Hosting with Render

render

Why Render?

- Automated Deployment
- Ease of Use
- Automatic Scaling
- Developer-Friendly
- Cost-Effective

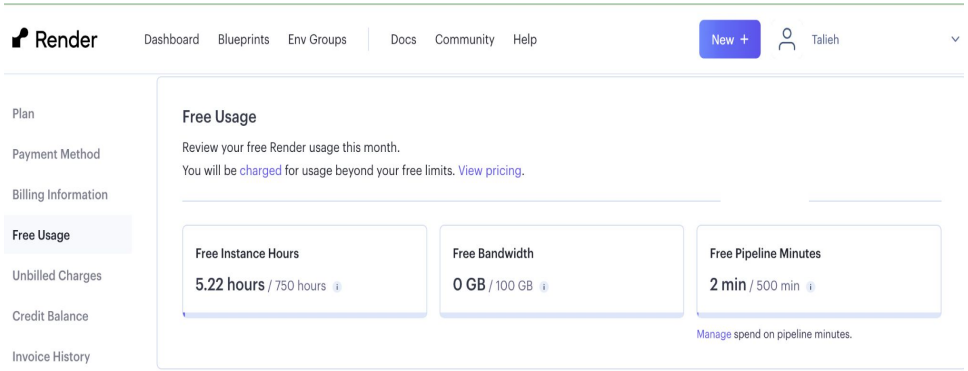


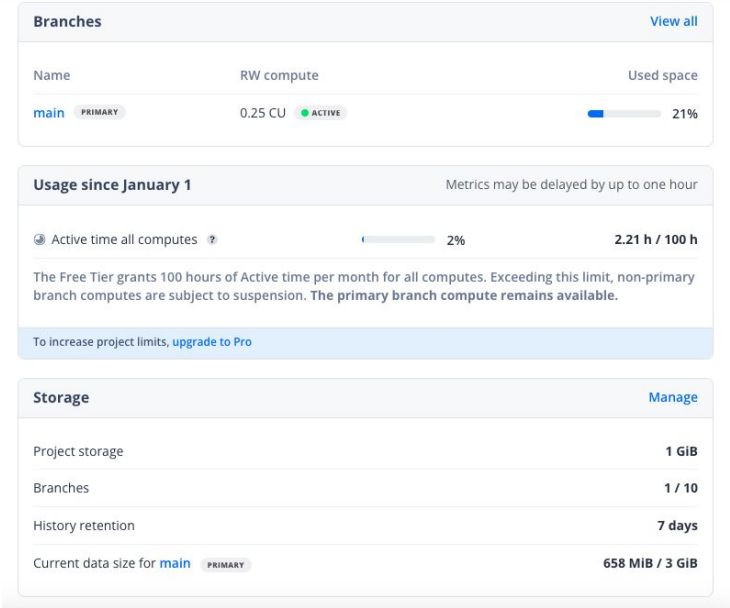
Cloud Hosting:
Challenges & Solutions

- **Database hosting on cloud**
- **Web application hosting on cloud**

render

NEON





Interactive Filters: Populating For All Options


```
{
  "years": [
    "2020",
    "2021",
    "2022",
    "2023"
  ],
  "area_names": [
    "77th Street",
    "Central",
    "Devonshire",
    "Foothill",
    "Harbor",
    "Hollenbeck",
    "Hollywood",
    "Mission",
    "N Hollywood",
    "Newton",
    "Northeast",
    "Olympic",
    "Pacific",
    "Rampart",
    "Southeast",
    "Southwest",
    "Topanga",
    "Van Nuys",
    "West LA",
    "West Valley",
    "Wilshire"
  ],
  "crime_categories": [
    "Aggravated Assault",
    "Burglary",
    "Burglary or Theft from Vehicle",
    "Domestic Violence",
    "Grand Theft Auto",
    "Homicide",
    "Other",
    "Other Theft",
    "Personal Theft",
    "Rape",
    "Robbery",
    "Simple Assault"
  ]
}
```


Dynamic API Route:
Query Filtered Data from PostgreSQL Table

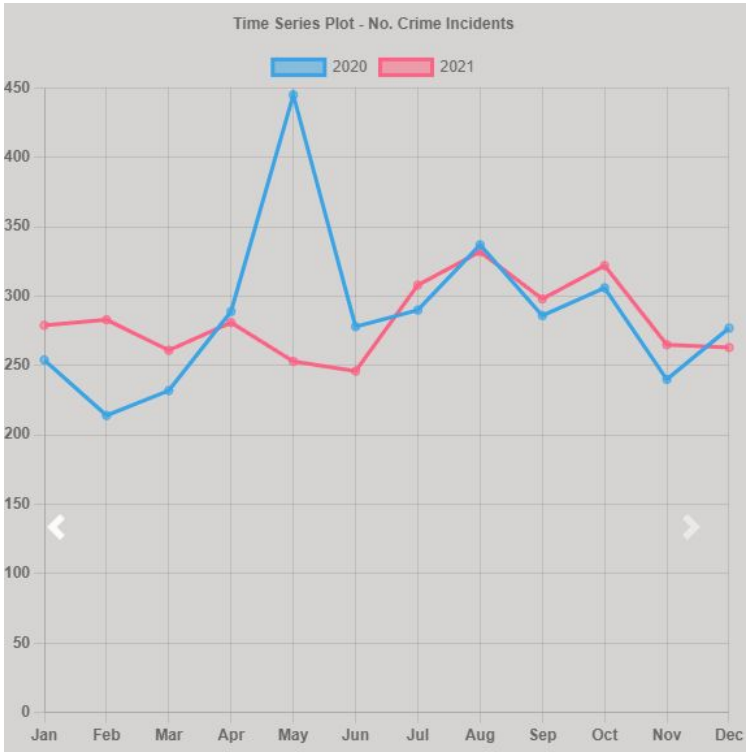

```
@app.route("/api/v1.0/<years_str>/<area_names_str>/<crime_categories_str>")
```


```
{
  "years": [
    2020,
    2021
  ],
  "area_names": [
    "Central"
  ],
  "crime_categories": [
    "Burglary"
  ],
  "crime_data": [
    {
      "id": 619,
      "dr_no": 200105263,
      "date_rptd": "Sun, 19 Jan 2020 00:00:00 GMT",
      "date_occ": "Sat, 18 Jan 2020 00:00:00 GMT",
      "time_occ": "19:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 36,
      "vict_sex": "Female",
      "vict_descent": "White",
      "premis_desc": "MULTI-UNIT DWELLING (APARTMENT, DUPLEX, ETC)",
      "location": "700 S  BROADWAY",
      "cross_street": null,
      "lat": "34.0452",
      "lon": "-118.2534"
    },
```

Visualisation #1: Time Series Plot (Chart.js)


```
{
  "years": [
    2020,
    2021
  ],
  "area_names": [
    "Central"
  ],
  "crime_categories": [
    "Burglary"
  ],
  "crime_data": [
    {
      "id": 619,
      "dr_no": 200105263,
      "date_rptd": "Sun, 19 Jan 2020 00:00:00 GMT",
      "date_occ": "Sat, 18 Jan 2020 00:00:00 GMT",
      "time_occ": "19:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 36,
      "vict_sex": "Female",
      "vict_descent": "White",
      "premis_desc": "MULTI-UNIT DWELLING (APARTMENT, DUPLEX, ETC)",
      "location": "700 S  BROADWAY",
      "cross_street": null,
      "lat": "34.0452",
      "lon": "-118.2534"
    },
    {
      "id": 687,
      "dr_no": 200105370,
      "date_rptd": "Mon, 20 Jan 2020 00:00:00 GMT",
      "date_occ": "Mon, 20 Jan 2020 00:00:00 GMT",
      "time_occ": "17:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 30,
      "vict_sex": "Female",
      "vict_descent": "Other",
      "premis_desc": "SINGLE FAMILY DWELLING",
      "location": "800    ALPINE                    ST",
      "cross_street": null,
      "lat": "34.0637",
      "lon": "-118.2440"
    },
```

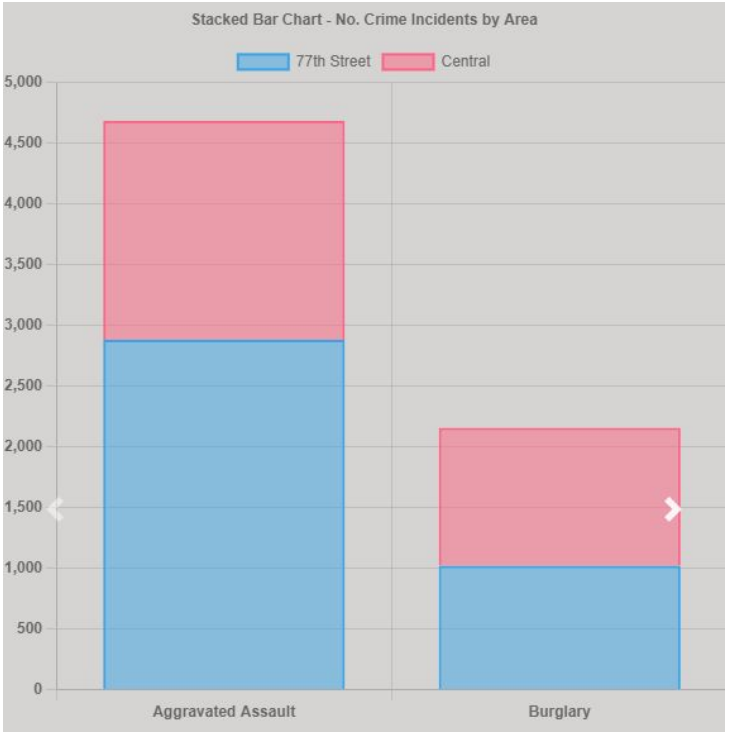

```
// Using the Chart JS library, this callback function initialis
const init_TimeSeries = (thisDataset) => {
    // Create an array of Calendar Months
    const monthLabels = ['Jan', 'Feb', 'Mar', 'Apr','May', 'Jun

    // From the Queried JSON Dataset, store the uniquely select
    // In the event only one year is selected e.g. 2020,2020
    const uniqueYearsSet = new Set(thisDataset.years);

    // Convert the Set() object into an array
    const uniqueYearsArray = [...uniqueYearsSet];

    // From the array of Years, get the min and max values
    const minYear = Math.min(...uniqueYearsArray);
    const maxYear = Math.max(...uniqueYearsArray);

    // Return a new array containing all years e.g. 2020, 2021,
    // Length of array determined by difference between max and
    // For every element in the array, return the addition of t
    // '_' parameter not used but passed to run the function
    const finalYears = Array.from({ length: maxYear - minYear +



    // Return an array of dataset objects where each represents
    // For every year in the array...
    const tsData = finalYears.map(Year => {
        // Return the JSON dataset with records for the current
        const currentYearData = thisDataset.crime_data.filter(i

        // Return a new array containing the count for each Cal
        // Length of array is 12 (Calendar Months)
        // Current Month in iteration is determined by its inde
        // From the already filtered JSON Dataset, return the l
        const getMonthCounts = Array.from({ length: 12 }, (_, m
          const currentMonth = monthIndex + 1;
          return currentYearData.filter(item => new Date(item.d
        });
```



Visualisation #2: Stacked Bar Chart (Chart.js)


```
{
  "years": [
    2020,
    2021
  ],
  "area_names": [
    "Central"
  ],
  "crime_categories": [
    "Burglary"
  ],
  "crime_data": [
    {
      "id": 619,
      "dr_no": 200105263,
      "date_rptd": "Sun, 19 Jan 2020 00:00:00 GMT",
      "date_occ": "Sat, 18 Jan 2020 00:00:00 GMT",
      "time_occ": "19:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 36,
      "vict_sex": "Female",
      "vict_descent": "White",
      "premis_desc": "MULTI-UNIT DWELLING (APARTMENT, DUPLEX, ETC)",
      "location": "700 S  BROADWAY",
      "cross_street": null,
      "lat": "34.0452",
      "lon": "-118.2534"
    },
    {
      "id": 687,
      "dr_no": 200105370,
      "date_rptd": "Mon, 20 Jan 2020 00:00:00 GMT",
      "date_occ": "Mon, 20 Jan 2020 00:00:00 GMT",
      "time_occ": "17:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 30,
      "vict_sex": "Female",
      "vict_descent": "Other",
      "premis_desc": "SINGLE FAMILY DWELLING",
      "location": "800    ALPINE                    ST",
      "cross_street": null,
      "lat": "34.0637",
      "lon": "-118.2440"
    },
```

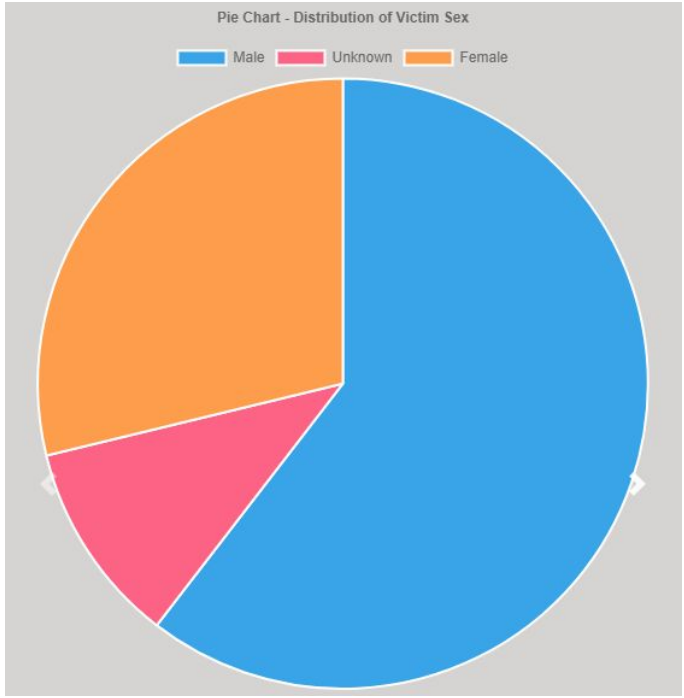

```
// Using the Chart JS library, this callback function initi
const init_StackedBarChart = (thisDataset) => {

    // From the Queried JSON Dataset, store the selected op
    const uniqueCrimes = thisDataset.crime_categories;
    const uniqueAreas = thisDataset.area_names;

    // Create a new object to store the labels and datasets
    // Used for when creating the Chart
    let barData = {
        labels: uniqueCrimes,
        datasets: []
    };

    // For every Area Name from the array...
    uniqueAreas.forEach(Area => {
        // Create a new object that represents a new column
        let columnData = {
            label: Area,     // Column Label = Current Area
            data: [],        // Dataset = count of Area Name
            borderWidth: 2  // Column Width = 2
        };

        // For every Crime Category from its respective arr
        // Tally up the count (length) of the current Area
        // Then, append the column object with the count
        uniqueCrimes.forEach(Crime => {
            let getCount = thisDataset.crime_data.filter(it
            columnData.data.push(getCount);
        });

        // Append the dataset key in the main bar chart obj
        barData.datasets.push(columnData);

    });
```



Visualisation #3: Pie Chart (Chart.js)


```
{
  "years": [
    2020,
    2021
  ],
  "area_names": [
    "Central"
  ],
  "crime_categories": [
    "Burglary"
  ],
  "crime_data": [
    {
      "id": 619,
      "dr_no": 200105263,
      "date_rptd": "Sun, 19 Jan 2020 00:00:00 GMT",
      "date_occ": "Sat, 18 Jan 2020 00:00:00 GMT",
      "time_occ": "19:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 36,
      "vict_sex": "Female",
      "vict_descent": "White",
      "premis_desc": "MULTI-UNIT DWELLING (APARTMENT, DUPLEX, ETC)",
      "location": "700 S  BROADWAY",
      "cross_street": null,
      "lat": "34.0452",
      "lon": "-118.2534"
    },
    {
      "id": 687,
      "dr_no": 200105370,
      "date_rptd": "Mon, 20 Jan 2020 00:00:00 GMT",
      "date_occ": "Mon, 20 Jan 2020 00:00:00 GMT",
      "time_occ": "17:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 30,
      "vict_sex": "Female",
      "vict_descent": "Other",
      "premis_desc": "SINGLE FAMILY DWELLING",
      "location": "800    ALPINE                       ST",
      "cross_street": null,
      "lat": "34.0637",
      "lon": "-118.2440"
    },
```

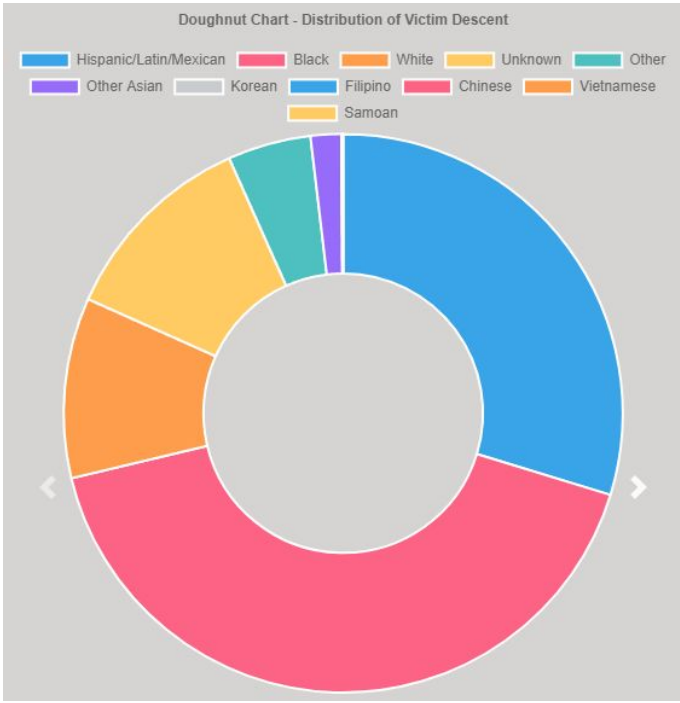

```
// Using the Chart JS library, this callback function initialise
const init_PieChart = (thisDataset) => {
    // New constant with null; used as condition
    const nullValue = null;

    // From the array of all vict_sex values from the dataset,
        // '...' spread operator used to convert the following
        // 'new Set()' is used to create a new Set instance to
        // using map() on crime_data (nested in thisDataset), ge
    let uniqueSexes = [...new Set(thisDataset.crime_data.map(ite
    uniqueSexes = uniqueSexes.filter(item => item !== nullValue

    // Create an empty array; used to store the count for every
    let sliceData = []

    // For every unique vict_sex value...
    // Get the length (count) for that vict_sex value in the Dat
    uniqueSexes.forEach(Sex => {
        const getCount = thisDataset.crime_data.filter(item =>
        sliceData.push(getCount);
    });


    // Destroy canvas in the <div> container and reinitialise to
    // Necessary in order to get the chart to utilise data from
    d3.select("#chartContainer3").html("");
    d3.select("#chartContainer3").html("<canvas id='jsChart_Pie'

    // select the Canvas HTML element for the Chart
    const pieCanvas = document.getElementById("jsChart_Pie").get

    // Initialise new Pie Chart where...
    const pieChart = new Chart(pieCanvas, {
        type: 'pie',
        data: {
            labels: uniqueSexes,
            datasets: [{
                data: sliceData}]
        },
```



Visualisation #4: Doughnut Chart (Chart.js)


```
{
  "years": [
    2020,
    2021
  ],
  "area_names": [
    "Central"
  ],
  "crime_categories": [
    "Burglary"
  ],
  "crime_data": [
    {
      "id": 619,
      "dr_no": 200105263,
      "date_rptd": "Sun, 19 Jan 2020 00:00:00 GMT",
      "date_occ": "Sat, 18 Jan 2020 00:00:00 GMT",
      "time_occ": "19:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 36,
      "vict_sex": "Female",
      "vict_descent": "White",
      "premis_desc": "MULTI-UNIT DWELLING (APARTMENT, DUPLEX, ETC)",
      "location": "700 S  BROADWAY",
      "cross_street": null,
      "lat": "34.0452",
      "lon": "-118.2534"
    },
    {
      "id": 687,
      "dr_no": 200105370,
      "date_rptd": "Mon, 20 Jan 2020 00:00:00 GMT",
      "date_occ": "Mon, 20 Jan 2020 00:00:00 GMT",
      "time_occ": "17:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 30,
      "vict_sex": "Female",
      "vict_descent": "Other",
      "premis_desc": "SINGLE FAMILY DWELLING",
      "location": "800    ALPINE                    ST",
      "cross_street": null,
      "lat": "34.0637",
      "lon": "-118.2440"
    },
```

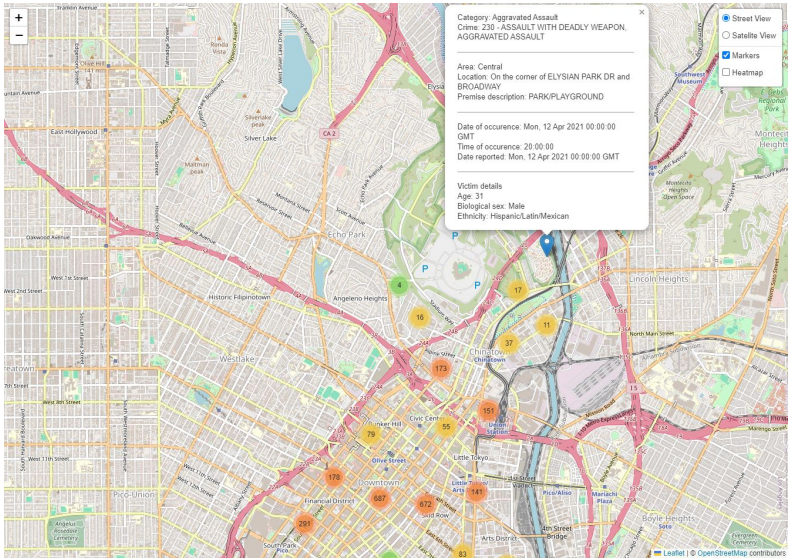

```
// Using the Chart JS library, this callback function initial
const init_DonutChart = (thisDataset) => {
    // New constant with null; used as condition
    const nullValue = null;

    // From the array of all vict_descent values from the data
        // '...' spread operator used to convert the following
        // 'new Set()' is used to create a new Set instance to
        // using map() on crime_data (nested in thisDataset),
    let uniqueDescents = [...new Set(thisDataset.crime_data.ma

    // Return the same array of unique vict_descent values whe
    uniqueDescents = uniqueDescents.filter(item => item !== nu

    // Create an empty array; used to store the count for ever
    let sliceData = []

    // For every unique vict_descent value...
    // Get the length (count) for that vict_descent value in t
    uniqueDescents.forEach(Descent => {
        const getCount = thisDataset.crime_data.filter(item =>
        sliceData.push(getCount);
    });


    // Destroy canvas in the <div> container and reinitialise
    // Necessary in order to get the chart to utilise data fro
    d3.select("#chartContainer4").html("");
    d3.select("#chartContainer4").html("<canvas id='jsChart_Do

    // select the Canvas HTML element for the Chart
    const donutCanvas = document.getElementById("jsChart_Donut

    // Initialise new Donut Chart where...
    const donutChart = new Chart(donutCanvas, {
        type: 'doughnut',
        data: {
            labels: uniqueDescents
```

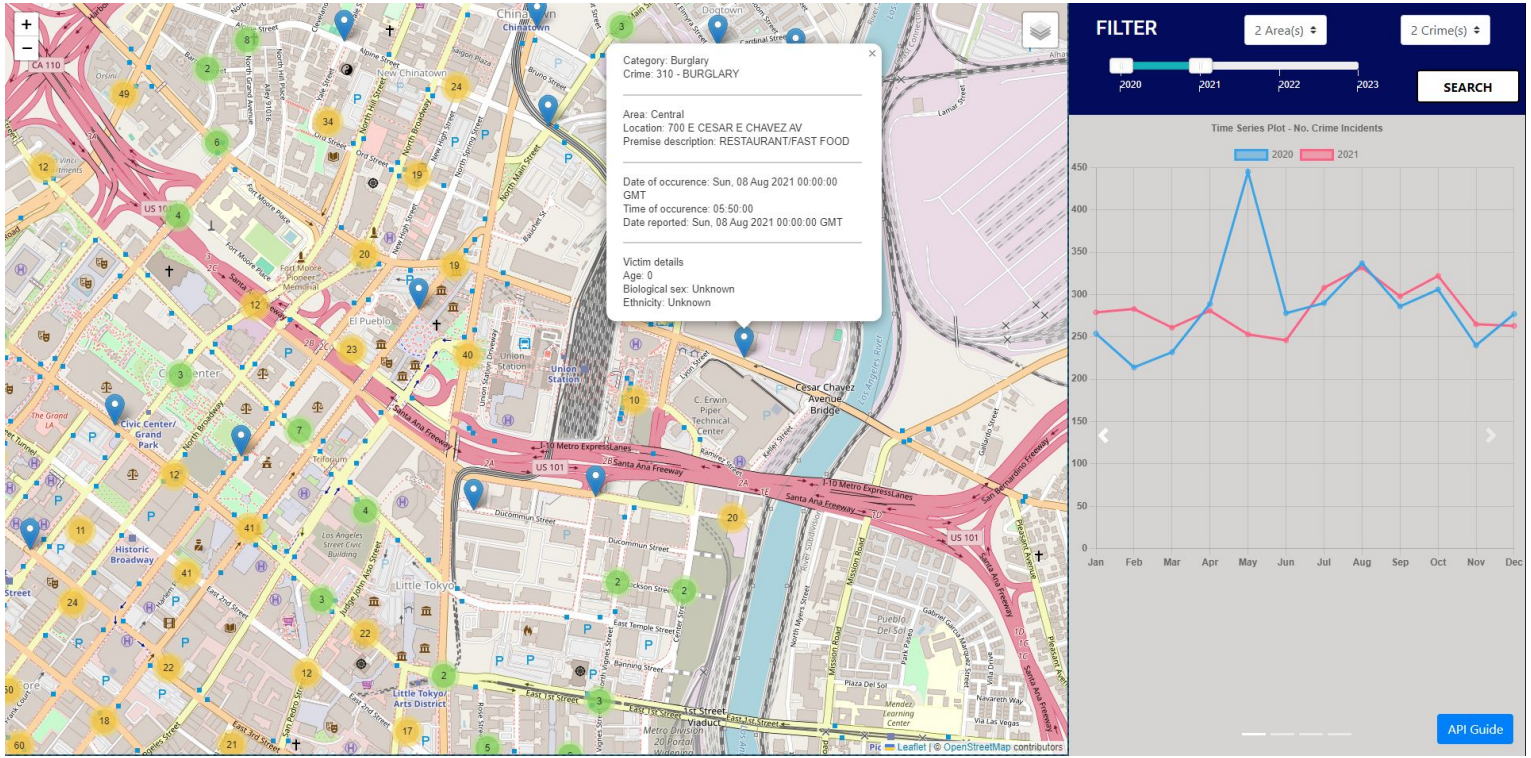
Visualisation #5: Interactive Map (Leaflet.js)


```
{
  "years": [
    2020,
    2021
  ],
  "area_names": [
    "Central"
  ],
  "crime_categories": [
    "Burglary"
  ],
  "crime_data": [
    {
      "id": 619,
      "dr_no": 200105263,
      "date_rptd": "Sun, 19 Jan 2020 00:00:00 GMT",
      "date_occ": "Sat, 18 Jan 2020 00:00:00 GMT",
      "time_occ": "19:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 36,
      "vict_sex": "Female",
      "vict_descent": "White",
      "premis_desc": "MULTI-UNIT DWELLING (APARTMENT, DUPLEX, ETC)",
      "location": "700 S  BROADWAY",
      "cross_street": null,
      "lat": "34.0452",
      "lon": "-118.2534"
    },
    {
      "id": 687,
      "dr_no": 200105370,
      "date_rptd": "Mon, 20 Jan 2020 00:00:00 GMT",
      "date_occ": "Mon, 20 Jan 2020 00:00:00 GMT",
      "time_occ": "17:00:00",
      "area_name": "Central",
      "crime_category": "Burglary",
      "crm_cd": 310,
      "crm_cd_desc": "BURGLARY",
      "vict_age": 30,
      "vict_sex": "Female",
      "vict_descent": "Other",
      "premis_desc": "SINGLE FAMILY DWELLING",
      "location": "800    ALPINE                    ST",
      "cross_street": null,
      "lat": "34.0637",
      "lon": "-118.2440"
    },
```


```
// Using Leaflet Library, this callback function initialises the
const init_Map = (thisDataset) => {
        // Declaring resultData as the contents of pulled data's
        resultData = thisDataset.crime_data;

        // Declaring empty marker cluster group variable to be fi
        let marks = L.markerClusterGroup();
        // Declaring empty dataset to be used to generate heatmap
        let hm_pts = [];

        // For every single datapoint gained from API call...
        for(i = 0; i < resultData.length; i++) {
            // Declaring variable that changes every iteration
            let datapoint = resultData[i];

            // Declaring datapoints in dataset
            let coords = [datapoint["lat"], datapoint["lon"]];
            let area = datapoint["area_name"];
            let category = datapoint["crime_category"];
            let code = datapoint["crm_cd"];
            let descp = datapoint["crm_cd_desc"];
            let street = datapoint["location"];
            let x_str = datapoint["cross_street"];
            let occ_date = datapoint["date_occ"];
            let occ_time = datapoint["time_occ"]
            let rep = datapoint["date_rptd"];
            let premise = datapoint["premis_desc"];
            let age = datapoint["vict_age"];
            let sex = datapoint["vict_sex"];
            let descent = datapoint["vict_descent"];

            // Popup function to generate text seen when clicking
            // Legend:
            // cat - category of crime
            // c_c - crime code
            // c_d - description of crime
            // ar - area where crime occured
            // st - street where crime occured
            // x - intersection
            // p - description of premise where crime occured
```



Interactive Dashboard Demonstration



Thanks For Listening!

References

- *https://github.com/falconpunch082/la-crime-visualiser?tab=readme-ov-file#target-audience*
- *https://data.lacity.org/Public-Safety/Crime-Data-from-2020-to-Present/2nrs-mtv8/about_data*