Group 3

COVID-19 DATASET

The Number of Confirmed cases, Deaths and Recovered cases every day across the globe

Adam, Ben, Ekta, William

The Dataset

Kaggle

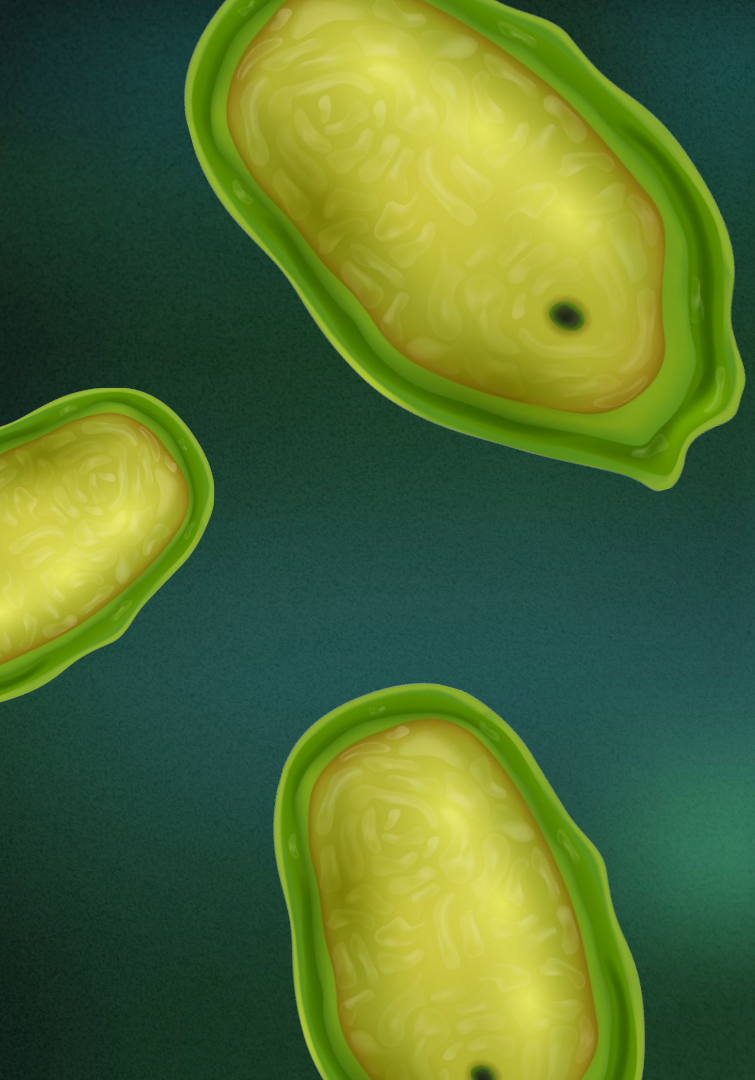
Devakumar K. P.
Data Set Updated 3 Years Ago

6 CSV files:

- Country_wise_latest
- Covid_19_clean_complete
- Day_wise
- Full_Grouped
- USA_county_wise
- Worldometer

Content

- Human-to-human transmission
- Confirmed
  - Cases
  - Deaths
  - Recovered
- Around the world



01.

Data Cleaning

```python
import pandas as pd
import sqlite3
import csv
```

```python
# Specify the path to your CSV file
csv_file_path = 'covid_19_clean_complete.csv'

# Open the CSV file
with open(csv_file_path, 'r') as file:
    # Create a CSV reader object
    csv_reader = csv.reader(file)

#     # Iterate over each row in the CSV file
#     for row in csv_reader:
#         # Each row is a list where each element corresponds to a column in the CSV
#         print(row)
```

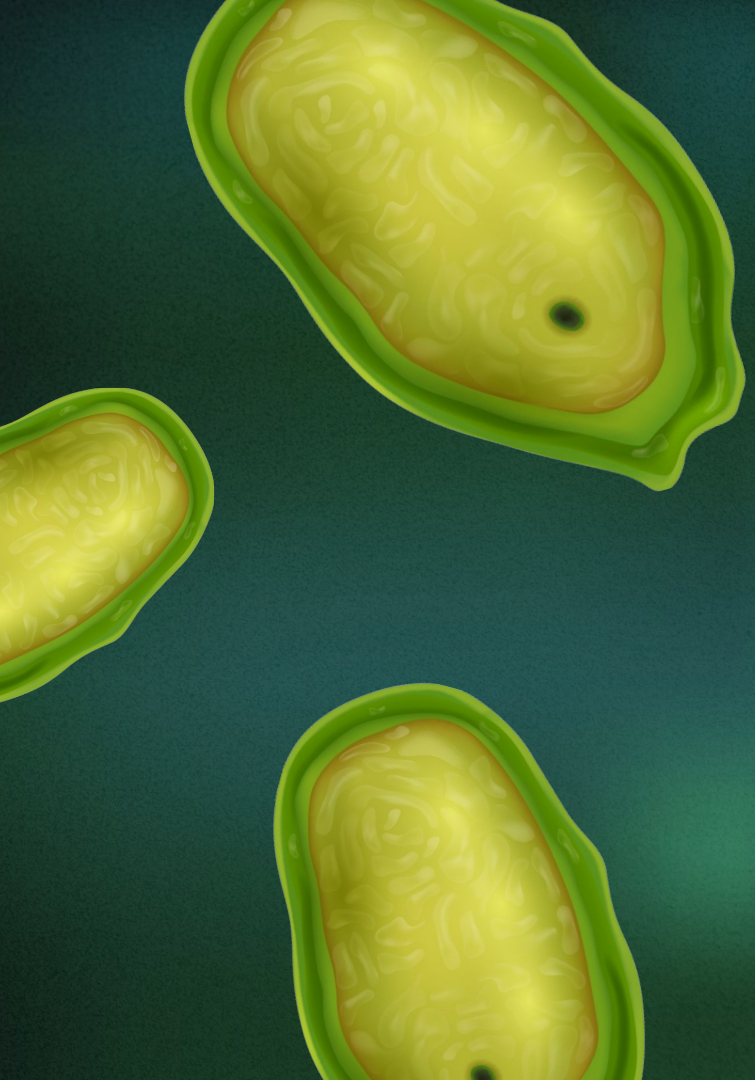
```python
# Load covid report data into a DataFrame
df = pd.read_csv("covid_19_clean_complete.csv")

# Filter and clean the DataFrame
df_transform = df[(df['Confirmed'] >1) &
                  (df['Deaths'] >= 0)]


df_transform = df_transform.copy()
df_transform.rename(columns={'Province/State': 'State', 'Country/Region': 'Country','WHO Region':'WHORegion'}, inplace=True)
#df_transform = df.dropna()
df_transform
```

```python
#Create/connect to a SQLite database
conn = sqlite3.connect('covid.db')
cursor = conn.cursor()
```

```python
create_sql = '''
    DROP TABLE IF EXISTS country
'''
cursor.execute(create_sql)
```

```
<sqlite3.Cursor at 0x2cf8bf0dcc0>
```

```python
create_sql = '''
    CREATE TABLE IF NOT EXISTS country (
        "RowID" INTEGER PRIMARY KEY AUTOINCREMENT,
        "Country" TEXT,
        "State" TEXT,
        "Lat" REAL,
        "Long" REAL,
        "Date" TEXT,
        "Confirmed" INTEGER,
        "Deaths" INTEGER,
        "Recovered" INTEGER,
        "Active" INTEGER,
        "WHORegion" TEXT

    )
'''
cursor.execute(create_sql)
```



02.

Data Extraction

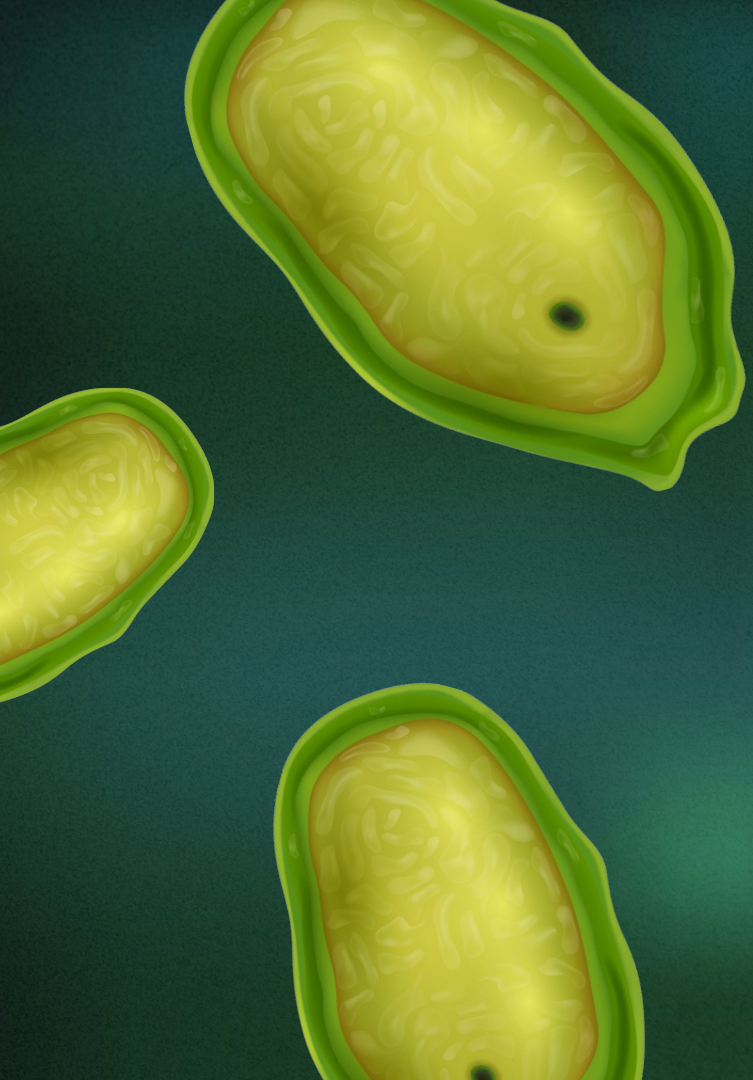
Loaded SQL Data into JSON Format using:

- SqlAlchemy
- Flask

Created 4 Api links:
1 for Geomap,
1 for Bubble chart
2 for Bar chart

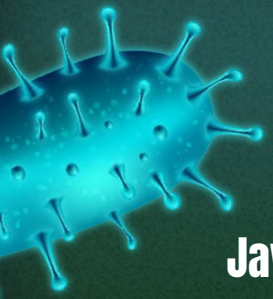
```
engine = create_engine("sqlite:///covid.db")


# reflect an existing database into a new model
Base = automap_base()
# reflect the tables
Base.prepare(engine, reflect=True)



# Save reference to the table
country = Base.classes.country
```

```
#geomap that shows total cases worldwide
@app.route("/api/geomap")
def for_map():

    session = Session(engine)

    results = session.query(country.Country, country.Sta
                            country.Confirmed, country.De

    results = [list(r) for r in results]

    countryResults = [result[0] for result in results]
    state = [result[1] for result in results]
    lat = [result[2] for result in results]
    long = [result[3] for result in results]
    confirmed = [result[4] for result in results]
    deaths = [result[5] for result in results]
    recovered = [result[6] for result in results]


    map_results = {
        "country": countryResults,
        "state": state,
        "lat": lat,
        "long": long,
        "confirmed": confirmed,
        "deaths": deaths,
        "recovered": recovered
    }

    session.close()

    return jsonify(map_results)
```



03.

Javascript



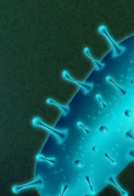
Javascript file contains:

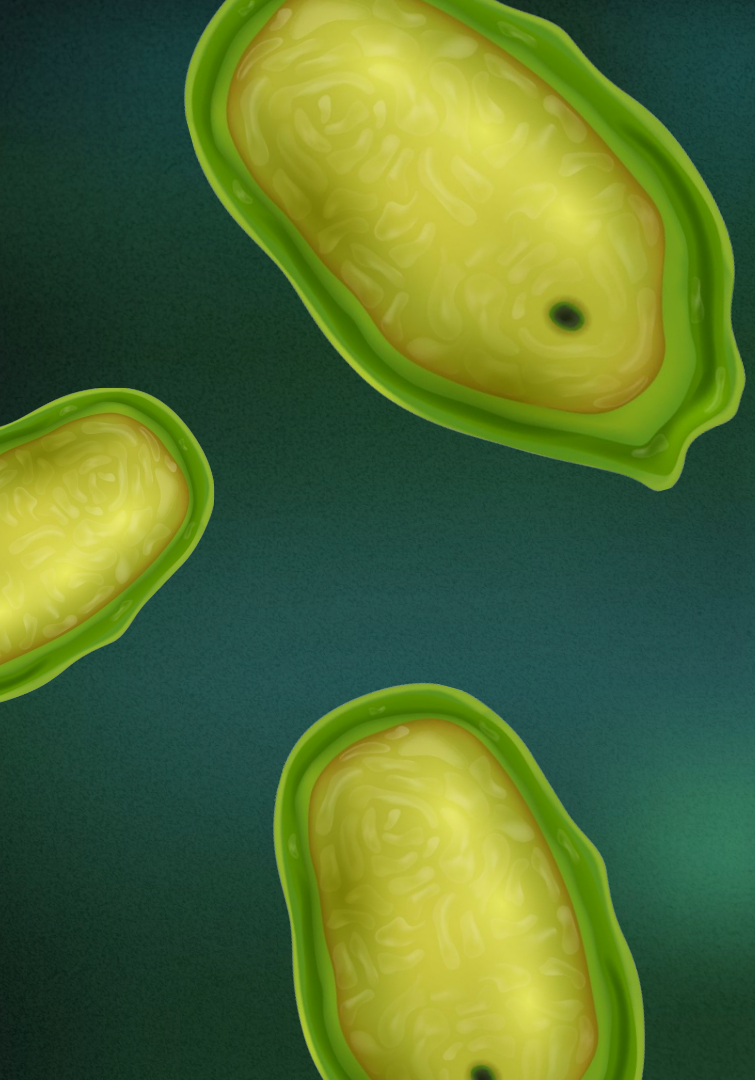
- Geomap code
- Leaflet Plugin: Conditional Markers
- Bubble Chart
- Bar Chart

```
// Bubble Chart
function buildBubbleChart(region) {

    url = 'api/bubble/'+region

    d3.json(url).then((data) => {

    let square = data['confirmed'].map((item) => {
        return Math.log2(item);
    });

    const colors = [];
    for (let i = 0; i < data['country'].length; i++) {
      const randomColor = "#" + Math.floor(Math.random() * 16777215).toSt
      colors.push(randomColor);
    };

    let bubbleData = [{
        x: data['country'],
          y: data['confirmed'],
          text: data['state'],
          mode: 'markers',
          marker: {
              color: colors,
              size: square
          }
    }];
```



```
// Get the data with d3.
let conLayer = L.conditionalMarkers([], {maxMarkers: 50});
d3.json("/api/geomap").then(function(response) {

  // Loop through the data.
  for (let i = 0; i < response['country'].length; i++) {
      // Add a new marker and bind a popup
      conLayer.addLayer(
        L.marker([response['lat'][i], response['long'][i]])
        .bindPopup("<h1>" + response['country'][i]+"</h1>" + "<hr>" +
        "<strong> State: </strong>" + response['state'][i] + "<br>" +
```





04.

HTML

HTML

Interactive Visualisation Setup:

- Utilises HTML and JavaScript to create an interactive web page.
- Includes dynamic charts for exploring COVID-19 cases, such as a conditional map, bubble chart, and stacked bar chart.

Map Integration:

- Integrates Leaflet library for mapping functionalities.
- Retrieves and displays geospatial data using D3.js and Leaflet to represent COVID-19 cases across different countries.

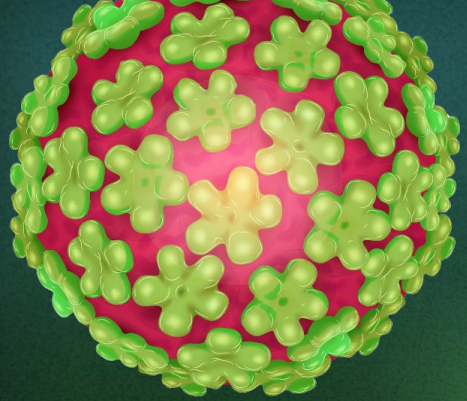
User Interaction:

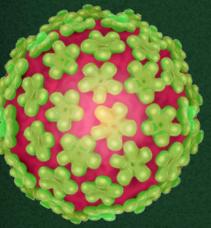
- Provides a dropdown menu allowing users to select a specific country.
- Updates the displayed map and charts dynamically based on the user's selected country, enhancing user engagement and customisation.

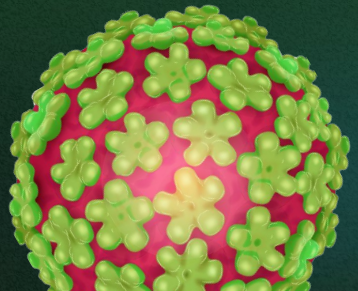
Data Visualisation with Plotly:

- Utilises Plotly library to create interactive and visually appealing charts (bubble chart and stacked bar chart).
- Presents COVID-19 statistics, including confirmed cases, recoveries, and deaths, with detailed information for each selected country.

```
123     <div class="header"> Use the interactive charts below to explore the dataset
124     </div>
125
126     <div class="container">
127         <div class="column-2">
128             <div class="dropdown">
129                 <div class="dropdown-content">
130                 <h5>Select Country:</h5>
131                <!---- <select id="selDataset" onchange="optionChanged(this.value)"></select>-->
132             </div>
133         </div>
134     </div>
135
136     <div class="column-3">
137         <!-- Your map content goes here -->
138         <div id="map" style="height: 100%"></div>
139         <script src="{{ url_for('static', filename='js/leaflet.conditionalLayer.js') }}"></script>
140     </div>
141 </div>
142
143 <div class="container">
144     <div class="column-4">
145         <div class="dropdown">
146             <label for="regions">Select WHO Region:</label>
147             <select name="region" id="region" onchange="optionRegionChanged(this.value)">
148                 <option value="Western Pacific">Western Pacific</option>
149                 <option value="Europe">Europe</option>
150                 <option value="Americas">Americas</option>
```





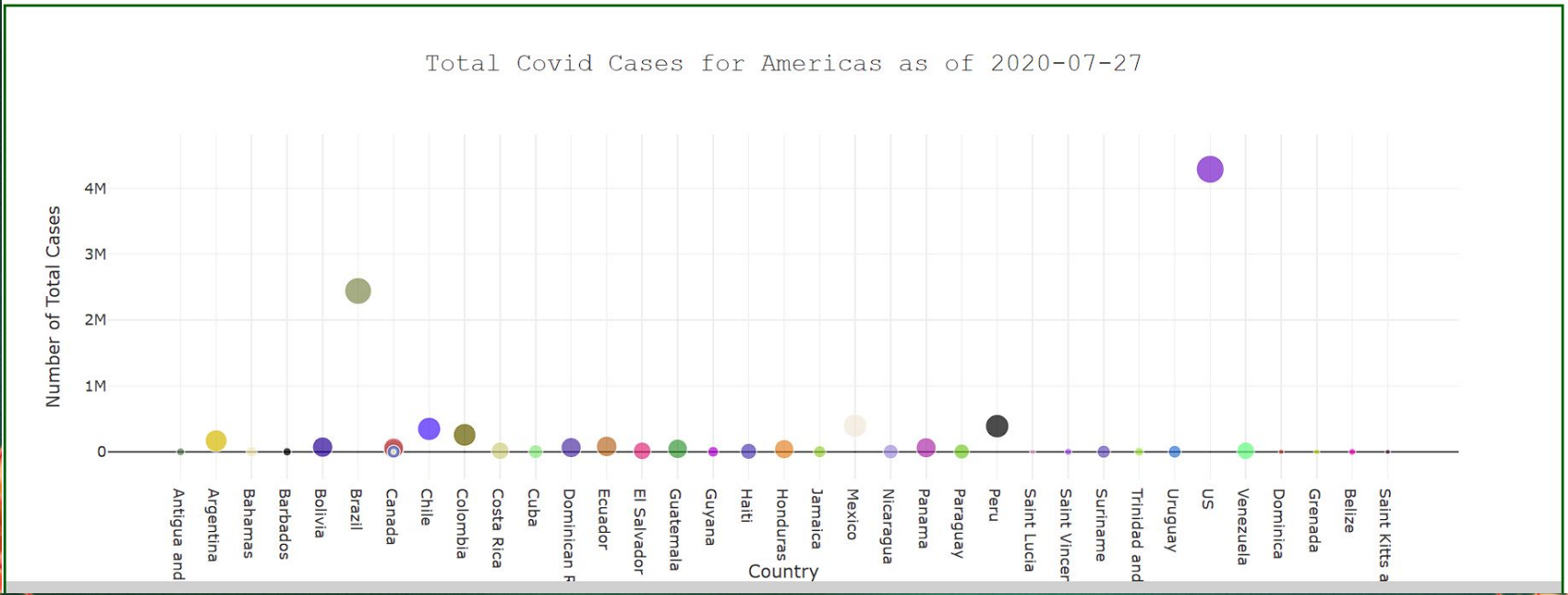


SHALL WE SEE THE WEBSITE?

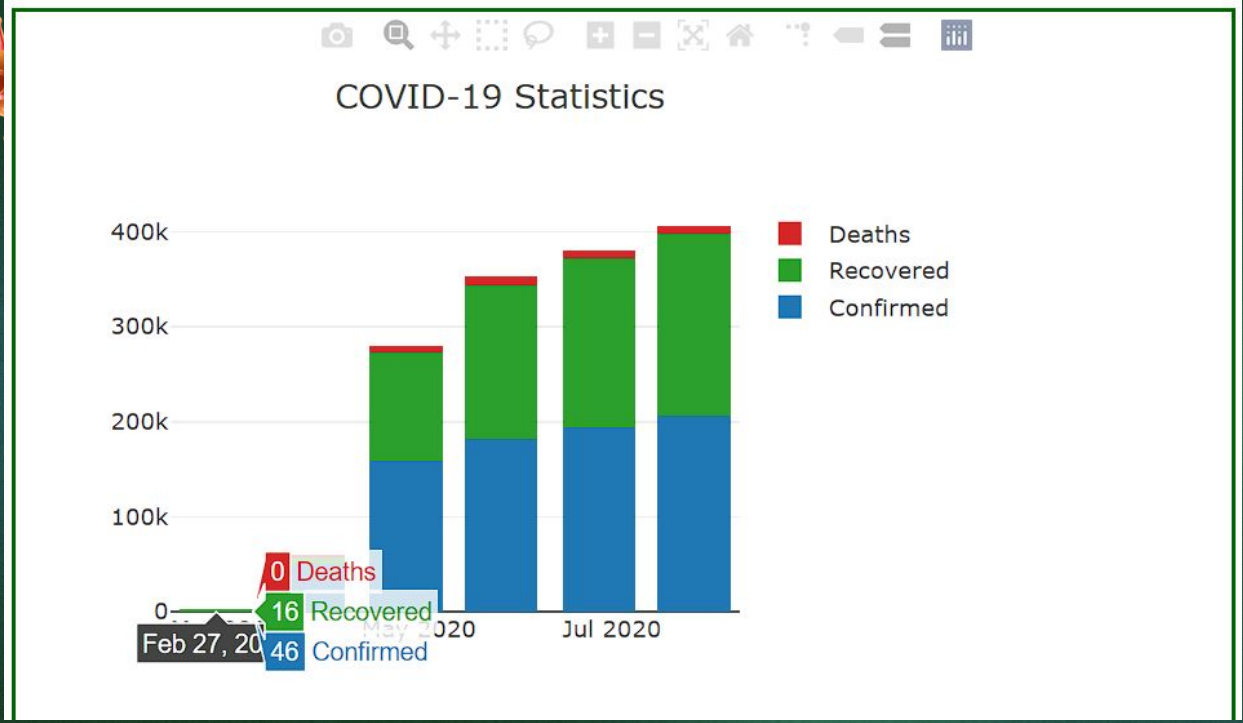
GEO MAP


+
−
Leaflet | Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

Bubble Chart



Bar Chart


COVID-19 Statistics
400k
300k
200k
100k
0
Deaths
Recovered
Confirmed
Jul 2020
0 Deaths
16 Recovered
46 Confirmed
Feb 27, 20

Q & A

THANKS!

Do you have any questions?