PROJECT : REAL-TIME AQI ANALYSIS AND VISUALIZATION OF INDIA

- IMPORTING NECESSARY LIBRARIES :

In [1]:
```
import pandas as pd
import folium
from folium.plugins import HeatMap
```

###-STEP 1 DOWNLOAD DATA

In [2]:
```
# Details of API at:- https://aqicn.org/api/
base_url = "https://api.waqi.info"
```

Got a special User token from:- https://aqicn.org/data-platform/token/#/ by registering myself on this site.

In [3]:
```
tok = "0976e0617860ee99e9fbbfb1e3f84c9e44fd4fa2"
```

(lat, long)-> bottom left, (lat, lon)-> top right

Location of India is 8N 61E to 37N, 97E approx

In [4]:
```
latlngbox = "8.0000,61.0000,37.0000,97.0000" # For India
```

In [5]:
```
trail_url=f"/map/bounds/?latlng={latlngbox}&token={tok}"
```

Joining the parts of URL:

In [6]:
```
my_data = pd.read_json(base_url + trail_url)
```

Printing 2 cols ‘status’ and ‘data’

In [7]:
```
print('columns->', my_data.columns)
```

```
columns-> Index(['status', 'data'], dtype='object')
```

###-STEP 2:- Create table like DataFrame

In [8]:
```
all_rows = []
for each_row in my_data['data']:
    all_rows.append([each_row['station']['name'],each_row['lat'],each_row['lon'],each_
    df = pd.DataFrame(all_rows,
    columns=['station_name', 'lat', 'lon', 'aqi'])
```

### -STEP 3:- Cleaning the DataFrame#

Converting Invalid parse to NaN

In [9]:
```
df['aqi'] = pd.to_numeric(df.aqi,
errors='coerce')
```

Printing Values with NaN :

In [10]:
```
print('with NaN->', df.shape)
```

```
with NaN-> (206, 4)
```

Remove NaN (Not a Number) entries in column:

In [11]:
```
df1 = df.dropna(subset = ['aqi'])
```

Printing Values Without NaN:

In [12]:
```
print('without NaN->', df1.shape)
```

```
without NaN-> (197, 4)
```

###-STEP 4:- Making folium heat map

In [13]:
```
df2 = df1[['lat', 'lon', 'aqi']]
```

To Print Our DataFrame:

In [14]:
```
print(df2.head)
```

```
<bound method NDFrame.head of            lat        lon    aqi
0    17.349694  78.451437  162.0
1    25.204762  85.514960  185.0
2    26.664451  87.195171  134.0
3    26.980660  87.343920   85.0
4    26.630860  84.900510  319.0
..         ...        ...    ...
201  16.987287  81.736318  157.0
202  19.252920  73.142019  155.0
203  22.431000  75.521300  110.0
204  22.410802  73.097923   59.0
205  24.584344  80.854941   53.0

[197 rows x 3 columns]>
```

Giving Central Location:

In [15]:
```
init_loc = [23, 77] # Approx over Bhopal
```

Getting and Printing Max_Aqi of Locations:

In [16]:
```
max_aqi = int(df1['aqi'].max())
print('max_aqi->', max_aqi)
```

```
max_aqi-> 694
```

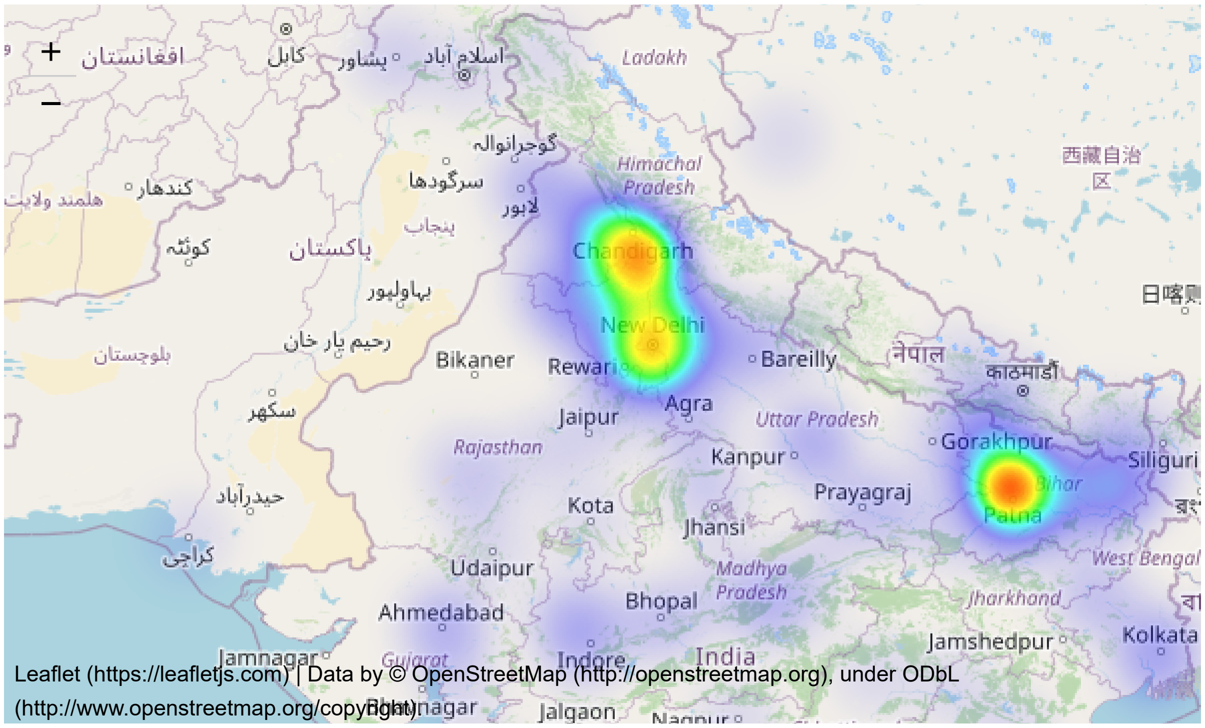
Visualization Of Live_HeatMap of India:

In [17]:
```
m = folium.Map(location = init_loc, zoom_start = 5)

heat_aqi = HeatMap(df2, min_opacity = 0.1, max_val = max_aqi,
radius = 20, blur = 20, max_zoom = 2)
m.add_child(heat_aqi)
m # Show the map
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_5872\2687615664.py:3: UserWarning: The `max_
val` parameter is no longer necessary. The largest intensity is calculated automatica
lly.
  heat_aqi = HeatMap(df2, min_opacity = 0.1, max_val = max_aqi,
```

Out[17]:

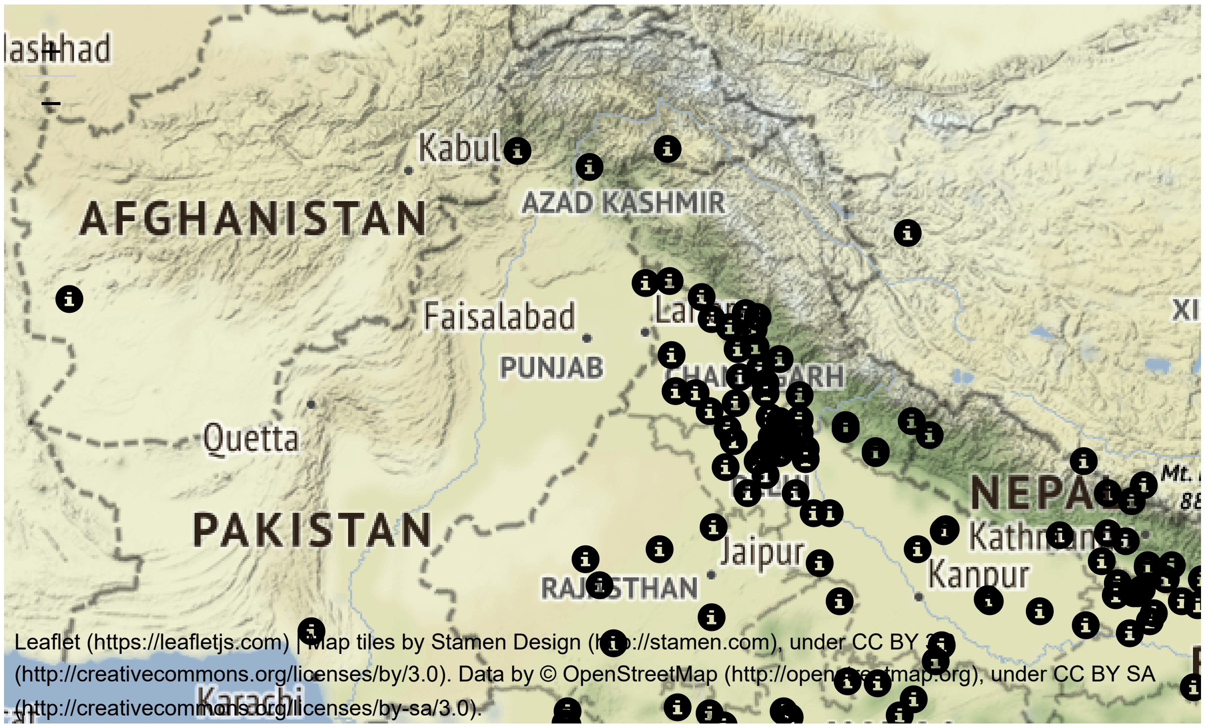


###-STEP 5 : Ploting stations on map

In [18]:

```
centre_point = [23.25, 77.41] # Approx over Bhopal
m2 = folium.Map(location = centre_point,
tiles = 'Stamen Terrain',
zoom_start= 6)
for idx, row in df1.iterrows():
    lat = row['lat']
    lon = row['lon']
    station = row['station_name'] + ' AQI=' + str(row['aqi'])
    station_aqi = row['aqi']
    if station_aqi > 300: ## Red for very bad AQI
        pop_color = 'red'
    elif station_aqi > 200:
        pop_color = 'orange' ## Orange for moderate AQI
    else:
        pop_color = 'green' ## Green for good AQI
    folium.Marker(location= [lat, lon],
    popup = station,
    icon = folium.Icon(color = pop_color)).add_to(m2)
m2 # Display map
```

Out[18]:



Leaflet (https://leafletjs.com) | Map tiles by Stamen Design (http://stamen.com), under CC BY 3.0 (http://creativecommons.org/licenses/by/3.0). Data by © OpenStreetMap (http://openstreetmap.org), under CC BY SA (http://creativecommons.org/licenses/by-sa/3.0).

In [19]:

```
import requests
import json
import pandas as pd
import re
import datetime
import time
import base64
from itertools import product

stationsData = pd.read_csv("C:/Users/hp/OneDrive/Desktop/PROJECT 3rd Sem/station.csv")

def getData(api, filters):
    url1 = "https://api.data.gov.in/resource/3b01bcb8-0b14-4abf-b6f2-c1bfd384ba69?api-
    criteriaAll = [[(k, re.sub(r'\s+', '%20', v)) for v in criteria[k]] for k in crite
    url2 = [url1 + ''.join(f'&filters[{ls}]={value}' for ls, value in p) for p in proc

    pollutionDfAll = pd.DataFrame()
    for i in url2:
        response = requests.get(i, verify=True)
        response_dict = json.loads(response.text)
        pollutionDf = pd.DataFrame(response_dict['records'])
        pollutionDfAll = pd.concat([pollutionDfAll, pollutionDf])

    return pollutionDfAll



api ="579b464db66ec23bdd000001fcbeb272b328454e41d1cd46d77298ba"
```

In the code below, there are two arguments that we needs to input - API Key Filter criteria. Filter criteria can have "state", "city",

"station", "pollutant_id". To see the unique values of state, city and station, you can download and refer the dataset shown above. Distinct values of pollutant_id are as follows -"PM2.5" "PM10" "NO2" "NH3" "SO2" "CO" "OZONE"

In [20]:
```
criteria = {'city':["Rupnagar","Punjab"], 'pollutant_id': ["PM10", "PM2.5","NO2","NH3"
mydata = getData(api, criteria)
mydata
```

Out[20]:

	id	country	state	city	station	last_update	pollutant_id	pollutant_min	pollutant_m
0	1417	India	Punjab	Rupnagar	Ratanpura, Rupnagar - Ambuja Cements	09-11-2022 08:00:00	PM10	89	2.
0	1416	India	Punjab	Rupnagar	Ratanpura, Rupnagar - Ambuja Cements	09-11-2022 08:00:00	PM2.5	93	3
0	1418	India	Punjab	Rupnagar	Ratanpura, Rupnagar - Ambuja Cements	09-11-2022 08:00:00	NO2	6	
0	1419	India	Punjab	Rupnagar	Ratanpura, Rupnagar - Ambuja Cements	09-11-2022 08:00:00	SO2	3	
0	1420	India	Punjab	Rupnagar	Ratanpura, Rupnagar - Ambuja Cements	09-11-2022 08:00:00	CO	39	1

To find AQI score of station(s) which is the most granular level of information. We can club it with the pollutant ID to narrow down Our search result.

In [21]:
```
criteria = {"station":["Anand Vihar, Delhi - DPCC", "Okhla Phase-2, Delhi - DPCC"], "p
mydata = getData(api, criteria)
mydata
```

Out[21]:

	id	country	state	city	station	last_update	pollutant_id	pollutant_min	pollutant_max	pollut
0	297	India	Delhi	Delhi	Anand Vihar, Delhi - DPCC	09-11-2022 08:00:00	PM10	140	418	
0	443	India	Delhi	Delhi	Okhla Phase-2, Delhi - DPCC	09-11-2022 08:00:00	PM10	123	416	

In [22]:

```
criteria={"station": ["Sector 22, Chandigarh - CPCC"],'pollutant_id': ["PM10", "PM2.5"
mydata = getData(api, criteria)
mydata
```

Out[22]:

	id	country	state	city	station	last_update	pollutant_id	pollutant_min	polluta
0	264	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	PM10	107	
0	263	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	PM2.5	80	
0	265	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	NO2	33	
0	266	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	NH3	4	
0	267	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	SO2	6	
0	268	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	CO	24	
0	269	India	Chandigarh	Chandigarh	Sector 22, Chandigarh - CPCC	09-11-2022 08:00:00	OZONE	10	

In [ ]: