*# Dependencies and Setup*

**import** **matplotlib.pyplot** **as** **plt**

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**import** **requests**

**import** **gmaps**

**import** **os**

**import** **json**

**from** **pprint** **import** pprint

*# Import API key*

**from** **api\_keys** **import** g\_key

**Store Part I results into DataFrame**

* Load the csv exported in Part I to a DataFrame

*# load output\_data\_file from WeatherPy located in the folder output data*

output\_data\_file = 'output\_data/cities.csv'

cities\_df = pd.read\_csv(output\_data\_file)

weather\_data\_frame = cities\_df.drop(columns=["Unnamed: 0"])

weather\_data\_frame

# Humidity Heatmap

* Configure gmaps.
* Use the Lat and Lng as locations and Humidity as the weight.
* Add Heatmap layer to map.

In [4]:

*# Configure gmaps key*

gmaps.configure(api\_key=g\_key)

In [5]:

*# Store latitude and longitude in locations*

locations = weather\_data\_frame[["Lat", "Lng"]]

*# Used humidity as the weight*

*# Fill NaN values and convert to float*

humidity = weather\_data\_frame["Humidity"].astype(float)

*# Figure Layout*

figure\_layout={

'width': '400px',

'height': '300px',

'border': '1px solid black',

'padding': '1px'

}

*# Plot the Heatmap*

fig = gmaps.figure()

*# create de heat layer*

heatmap\_layer = gmaps.heatmap\_layer(locations, weights=humidity,

dissipating=**False**, max\_intensity=5,

point\_radius=1)

*# Add the layer to the map*

fig.add\_layer(heatmap\_layer)

*# Display figure*

fig

Figure(layout=FigureLayout(height='420px'))

### Create new DataFrame fitting weather criteria

* Narrow down the cities to fit weather conditions.
* Drop any rows will null values.

In [6]:

*# Narrow down the DataFrame to find your ideal weather condition. For example:*

*# A max temperature lower than 80 degrees but higher than 70.*

*# Wind speed less than 10 mph.*

*# Zero cloudiness.*

*# Drop any rows that don't contain all three conditions.*

*# You want to be sure the weather is ideal*

*# Create ideal cities data frame with the desirable variables*

ideal\_cities\_df = pd.DataFrame(weather\_data\_frame, columns = ["City", "Cloudiness", "Max Temp", "Wind Speed"])

max\_temp = (ideal\_cities\_df["Max Temp"] <= 80) & (ideal\_cities\_df["Max Temp"] > 70)

wind\_speed = ideal\_cities\_df["Wind Speed"] < 10

cloudiness = ideal\_cities\_df["Cloudiness"] == 0

*# Ideals Cities data frame with max\_temp, wind\_speed and cloudiness]*

ideal\_cities\_df[cloudiness & max\_temp & wind\_speed]

**Hotel Map**

* Store into variable named hotel\_df.
* Add a "Hotel Name" column to the DataFrame.
* Set parameters to search for hotels with 5000 meters.
* Hit the Google Places API for each city's coordinates.
* Store the first Hotel result into the DataFrame.
* Plot markers on top of the heatmap.

In [7]:

*# Store into variable named hotel\_df .*

hotel\_df = pd.DataFrame(weather\_data\_frame, columns=["City", "Country", "Lat", "Lng"])

*# Add a "Hotel Name" column to the weather\_data\_frame.*

hotel\_df["Hotel Name"] = ""

*# see table*

hotel\_df

*# Set parameters to search for hotels with 5000 meters.*

coordinates = f"**{hotel\_df['Lat'][0]}**,**{hotel\_df['Lng'][0]}**"

search = "hotel"

radius = 5000

target\_type = "hotel"

*# set up a parameters dictionary*

params = {

"location": coordinates,

"keyword": search,

"radius": radius,

"type": target\_type,

"key": g\_key

}

*# Build URL using the Google Maps API*

*# base url*

base\_url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json"

*# run a request using our params dictionary*

response = requests.get(base\_url, params=params)

cities\_ideal = response.json()

pprint(response.json(), depth=1)

{'html\_attributions': [], 'results': [...], 'status': 'OK'}

In [9]:

hotel\_df.iloc[0][0]

Out[9]:

'jamestown'

# Using Google Places API to find the first hotel for each city located within 5000 meters of your coordinates

In [10]:

*# Define a list to storage the hotel names*

hotel\_name = []

*# Assign Geo Coordinates*

radious = '5000'

keyword = 'hotel'

types = 'hotel'

*# set up a parameters dictionary*

params = {

"keyword": keyword,

"radius": radius,

"types": types,

"key": g\_key

}

*# loop through*

**for** index, row **in** hotel\_df.iterrows():

*# get lat, lng from df*

lat = row["Lat"]

lng = row["Lng"]

*# change location each iteration while leaving original params in place*

params["location"] = f"**{lat}**,**{lng}**"

base\_url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json"

*# make request and print url*

name\_address = requests.get(base\_url, params=params)

*# convert to json*

name\_address = name\_address.json()

*# print(json.dumps(name\_address, indent=4, sort\_keys=True))*

**try**:

hotel\_name.append(name\_address['results'][0]['name'])

**except** **IndexError**:

hotel\_name.append(np.nan)

In [11]:

hotel\_name

*# export file*

hotel\_df['Hotel Name']= hotel\_name

hotel\_df.dropna()

hotel\_df.to\_csv('output\_data/hotel\_export.csv')

# Plot the hotels on top of the humidity heatmap with each pin containing the \*\*Hotel Name\*\*, \*\*City\*\*, and \*\*Country\*\*.

In [13]:

*# NOTE: Do not change any of the code in this cell*

*# Using the template add the hotel marks to the heatmap*

info\_box\_template = """

<dl>

<dt>Name</dt><dd>{Hotel Name}</dd>

<dt>City</dt><dd>**{City}**</dd>

<dt>Country</dt><dd>**{Country}**</dd>

</dl>

"""

*# Store the DataFrame Row*

*# NOTE: be sure to update with your DataFrame name*

hotel\_info = [info\_box\_template.format(\*\*row) **for** index, row **in** hotel\_df.iterrows()]

locations = hotel\_df[["Lat", "Lng"]]

In [14]:

*# Add marker layer ontop of heat map*

markers = gmaps.marker\_layer(locations)

*# Add the layer to the map*

fig.add\_layer(markers)

*# Display Map*

fig

Figure(layout=FigureLayout(height='420px'))

In [15]:

*# Add marker layer ontop of heat map*

markers\_layer = gmaps.marker\_layer(locations, info\_box\_content=hotel\_info)

*# Add the layer to the map*

fig.add\_layer(markers\_layer)

*# Display Map*

fig