Deliverable 2. Create a Customer Travel Destinations Map.

In [2]:
```
# Dependencies and Setup
import pandas as pd
import requests
import gmaps

# Import API key
import sys
sys.path.append("../")
from config import g_key

# Configure gmaps API key
gmaps.configure(api_key=g_key)
```

In [3]:
```
# 1. Import the WeatherPy_database.csv file.
city_data_df = pd.read_csv("../Weather_Database/WeatherPy_database.csv")
city_data_df.head()
```

Out[3]:

	City_ID	City	Country	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Current Description	Date
0	0	Kalmunai	LK	7.4167	81.8167	77.18	79	93	4.92	overcast clouds	2021-03-28 22:48:13
1	1	Talnakh	RU	69.4865	88.3972	-14.22	99	59	4.16	broken clouds	2021-03-28 22:48:14
2	2	Snasa	NO	64.2457	12.3778	35.67	98	100	3.18	light rain	2021-03-28 22:48:14
3	3	Ushuaia	AR	-54.8000	-68.3000	46.40	61	75	14.97	broken clouds	2021-03-28 22:48:14
4	4	Bredasdorp	ZA	-34.5322	20.0403	60.80	88	100	11.50	overcast clouds	2021-03-28 22:48:14

In [4]:
```
# 2. Prompt the user to enter minimum and maximum temperature criteria
# Ask the customer to add a minimum and maximum temperature value.
min_temp = float(input("What is the minimum temperature you would like for your trip? "))
max_temp = float(input("What is the maximum temperature you would like for your trip? "))
```

```
What is the minimum temperature you would like for your trip? 65
What is the maximum temperature you would like for your trip? 80
```

In [5]:
```
# 3. Filter the city_data_df DataFrame using the input statements to create a new DataFrame using the loc method.
preferred_cities_df = city_data_df.loc[(city_data_df["Max Temp"] <= max_temp) &
                                       (city_data_df["Max Temp"] >= min_temp)]
preferred_cities_df
```

Out[5]:

	City_ID	City	Country	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Current Description	Date
0	0	Kalmunai	LK	7.4167	81.8167	77.18	79	93	4.92	overcast clouds	2021-03-28 22:48:13

	City_ID	City	Country	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Current Description	Date
5	5	Hilo	US	19.7297	-155.0900	71.60	78	90	4.61	overcast clouds	2021-03-28 22:48:14
7	7	Sampit	ID	-2.5333	112.9500	74.10	97	100	0.72	moderate rain	2021-03-28 22:48:15
8	8	Victoria	HK	22.2855	114.1577	75.99	87	23	6.98	few clouds	2021-03-28 22:48:09
9	9	Carnarvon	AU	-24.8667	113.6333	71.60	94	40	5.75	scattered clouds	2021-03-28 22:48:15
...	...	...	...	...	...	...	...	...	...	...	...
660	660	Kidal	ML	18.4411	1.4078	73.53	10	0	11.59	clear sky	2021-03-28 22:57:36
661	661	Veraval	IN	20.9000	70.3667	76.89	71	6	11.32	clear sky	2021-03-28 22:57:37
662	662	Cacheu	GW	12.2706	-16.1658	77.00	69	0	4.61	clear sky	2021-03-28 22:57:37
665	665	Geraldton	AU	-28.7667	114.6000	66.20	77	14	4.61	few clouds	2021-03-28 22:57:38
666	666	San Cristobal	VE	7.7669	-72.2250	78.80	73	40	9.22	scattered clouds	2021-03-28 22:57:38

217 rows × 11 columns

In [7]:
```
# 4a. Determine if there are any empty rows.
preferred_cities_df.isnull().sum() #Country -2
preferred_cities_df.count()
```

Out[7]:
```
City_ID                217
City                   217
Country                215
Lat                    217
Lng                    217
Max Temp               217
Humidity               217
Cloudiness             217
Wind Speed             217
Current Description    217
Date                   217
dtype: int64
```

In [10]:
```
# 4b. Drop any empty rows and create a new DataFrame that doesn't have empty rows.
clean_df = preferred_cities_df.dropna() #Remove records with missing Countries, etc...
```

In [14]:
```
# 5a. Create DataFrame called hotel_df to store hotel names along with city, country, max temp, and coordinates.
hotel_df = clean_df[["City", "Country", "Max Temp", "Current Description", "Lat", "Lng"]].copy()

# 5b. Create a new column "Hotel Name"
hotel_df["Hotel Name"] = ""
hotel_df.head(10)
```

Out[14]:

	City	Country	Max Temp	Current Description	Lat	Lng	Hotel Name
0	Kalmunai	LK	77.18	overcast clouds	7.4167	81.8167	
5	Hilo	US	71.60	overcast clouds	19.7297	-155.0900	
7	Sampit	ID	74.10	moderate rain	-2.5333	112.9500	
8	Victoria	HK	75.99	few clouds	22.2855	114.1577	
9	Carnarvon	AU	71.60	scattered clouds	-24.8667	113.6333	
10	Cape Town	ZA	68.00	clear sky	-33.9258	18.4232	
12	Pisco	PE	73.40	clear sky	-13.7000	-76.2167	
14	Souillac	MU	75.20	few clouds	-20.5167	57.5167	
17	Mizan Teferi	ET	66.72	overcast clouds	6.9833	35.5833	
18	Padang	ID	73.40	light rain	-0.9492	100.3543	

In [15]:

```
# 6a. Set parameters to search for hotels with 5000 meters.
params = {
    "radius": 5000,
    "type": "lodging",
    "key": g_key
}

# 6b. Iterate through the hotel DataFrame.
for index, row in hotel_df.iterrows():
    # 6c. Get latitude and longitude from DataFrame
    lat = row["Lat"]
    lng = row["Lng"]
    params["location"] = f"{lat},{lng}"

    city_location = row["City"] + ", " + row["Country"]
    geolocation = params["location"]

    # 6d. Set up the base URL for the Google Directions API to get JSON data.
    base_url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json"

    # 6e. Make request and retrieve the JSON data from the search.
    hotels = requests.get(base_url, params=params).json()

    # 6f. Get the first hotel from the results and store the name, if a hotel isn't found skip the city.
    try:
        hotel_df.loc[index, "Hotel Name"] = hotels["results"][0]["name"]
    except (IndexError):
        print(f"Hotel not found for {city_location}, {geolocation}... skipping.")
```

```
Hotel not found for Bongandanga, CD, 1.5,21.05... skipping.
Hotel not found for Pitmoaga, BF, 12.2397,-1.8767... skipping.
Hotel not found for Gamba, GA, -2.65,10.0... skipping.
Hotel not found for Ati, TD, 13.2154,18.3353... skipping.
Hotel not found for Beloha, MG, -25.1667,45.05... skipping.
Hotel not found for Goundam, ML, 16.4145,-3.6708... skipping.
Hotel not found for Filingue, NE, 14.3521,3.3168... skipping.
Hotel not found for Koindu, GN, 8.4386,-10.3253... skipping.
Hotel not found for Elko, US, 41.0002,-115.5012... skipping.
Hotel not found for Linay, PH, 8.5167,123.1333... skipping.
Hotel not found for Aripuana, BR, -9.1667,-60.6333... skipping.
Hotel not found for Amapa, BR, 1.0,-52.0... skipping.
Hotel not found for Tessalit, ML, 20.1986,1.0114... skipping.
Hotel not found for Umm Kaddadah, SD, 13.6017,26.6876... skipping.
Hotel not found for Mayor Pablo Lagerenza, PY, -19.9309,-60.7718... skipping.
Hotel not found for Lahij, YE, 13.1667,44.5833... skipping.
Hotel not found for Twin Falls, US, 42.3505,-114.6445... skipping.
Hotel not found for Raga, SS, 8.4596,25.678... skipping.
Hotel not found for Yashan, CN, 22.1975,109.9419... skipping.
Hotel not found for Kidal, ML, 18.4411,1.4078... skipping.
Hotel not found for Cacheu, GW, 12.2706,-16.1658... skipping.
```

In [36]:
```
# 7. Drop the rows where there is no Hotel Name.
hotel_df.count()
```

Out[36]:
```
City                  215
Country               215
Max Temp              215
Current Description   215
Lat                   215
Lng                   215
Hotel Name            215
dtype: int64
```

In [46]:
```
import numpy as np
hotel_df.loc[(hotel_df["Hotel Name"] == ""),"Hotel Name"] = np.nan
hotel_df
```

Out[46]:

	City	Country	Max Temp	Current Description	Lat	Lng	Hotel Name
0	Kalmunai	LK	77.18	overcast clouds	7.4167	81.8167	VS Villa
5	Hilo	US	71.60	overcast clouds	19.7297	-155.0900	Hilo Hawaiian Hotel
7	Sampit	ID	74.10	moderate rain	-2.5333	112.9500	Aquarius Boutique Hotel Sampit
8	Victoria	HK	75.99	few clouds	22.2855	114.1577	Mini Hotel Central
9	Carnarvon	AU	71.60	scattered clouds	-24.8667	113.6333	Hospitality Carnarvon
...	...	...	...	...	...	...	...
660	Kidal	ML	73.53	clear sky	18.4411	1.4078	NaN

	City	Country	Max Temp	Current Description	Lat	Lng	Hotel Name
661	Veraval	IN	76.89	clear sky	20.9000	70.3667	Lords Inn Somnath
662	Cacheu	GW	77.00	clear sky	12.2706	-16.1658	NaN
665	Geraldton	AU	66.20	few clouds	-28.7667	114.6000	Broadwater Mariner Resort
666	San Cristobal	VE	78.80	scattered clouds	7.7669	-72.2250	Posada Villaven C.A.

215 rows × 7 columns

In [48]:
```
hotel_df.isnull().sum() # 21 empty hotels
#hotel_df.count()
```

Out[48]:
```
City                  215
Country               215
Max Temp              215
Current Description   215
Lat                   215
Lng                   215
Hotel Name            194
dtype: int64
```

In [51]:
```
clean_hotel_df = hotel_df.dropna() #Remove 21 empty hotels
#clean_hotel_df
```

In [52]:
```
# 8a. Create the output File (CSV)
output_data_file = "WeatherPy_vacation.csv"
# 8b. Export the City_Data into a csv
clean_hotel_df.to_csv(output_data_file, index_label="City_ID")
```

In [63]:
```
# 9. Using the template add city name, the country code, the weather description and maximum temperature for the city.
info_box_template = """
<dl>
<dt>Hotel Name</dt><dd>{Hotel Name}</dd>
<dt>City</dt><dd>{City}</dd>
<dt>Country</dt><dd>{Country}</dd>
<dt>Weather</dt><dd>{Current Description}, {Max Temp} °F</dd>
</dl>
"""

# 10a. Get the data from each row and add it to the formatting template and store the data in a list.
hotel_info = [info_box_template.format(**row) for index, row in clean_hotel_df.iterrows()]

# 10b. Get the latitude and longitude from each row and store in a new DataFrame.
locations = clean_hotel_df[["Lat", "Lng"]]
```

In [64]:
```
# 11a. Add a marker layer for each city to the map.
```

```
locations = clean_hotel_df[["Lat", "Lng"]]
max_temp = clean_hotel_df["Max Temp"]
fig = gmaps.figure(center=(30.0, 31.0), zoom_level=1.5)
heat_layer = gmaps.heatmap_layer(locations, weights=max_temp,dissipating=False,
             max_intensity=300, point_radius=4)
marker_layer = gmaps.marker_layer(locations, info_box_content=hotel_info)
fig.add_layer(heat_layer)
fig.add_layer(marker_layer)
# 11b. Display the figure
fig
```