```
#!/usr/bin/env python

# In[1]:


# Dependencies and Setup
get_ipython().run_line_magic('matplotlib', 'inline')
import pandas as pd
import requests
import gmaps
import gmaps.datasets
import time

# Import API key
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("WeatherPy_database.csv")
city_data_df.head()


# In[4]:


# 2. Prompt the user to enter minimum and maximum temperature criteria
min_temp = float(input("What is your desired minimum temperature for your trip?
"))
max_temp = float(input("What is your desired maximum temperature for your trip?
"))


# 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.head(10)
```

```
# In[6]:


# 4a. Determine if there are any empty rows.
preferred_cities_df.isnull().sum()


# In[7]:


# 4b. Drop any empty rows and create a new DataFrame that doesn't have empty
rows.
clean_df = preferred_cities_df.dropna()
clean_df


# In[8]:


# 4b.1 Recheck if there are any empty rows.
clean_df.isnull().sum()


# In[9]:


# 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)


# In[10]:


# 6a. Set parameters to search for hotels with 5000 meters.
import json
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}"

    # 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("Hotel not found...Skipping.")



# In[11]:


# 7. Drop the rows where there is no Hotel Name.
clean_hotel_df = hotel_df.loc[(hotel_df["Hotel Name"]!='')]
clean_hotel_df.head(10)


# In[12]:


# 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[13]:
```

```
# 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>Current Weather</dt><dd>{Current Description}</dd>
<dt>Max Temp</dt><dd>{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[15]:


# 11a. Add a marker layer for each city to the map.
marker_layer = gmaps.marker_layer(locations, info_box_content=hotel_info)
fig = gmaps.figure(center=(30.0, 31.0), zoom_level=1.5)
fig.add_layer(marker_layer)

# 11b. Display the figure
fig


# In[ ]:
```