Module 6 Challenge

Start Assignment

Due Wednesday by 11:59pm **Points** 100 **Submitting** a text entry box or a website url

Background

Jack loves the PlanMyTrip app. Beta testers love it too. And, as with any new product, they've recommended a few changes to take the app to the next level. Specifically, they recommend adding the weather description to the weather data.

For this challenge, you will use the weather description data you've already retrieved in this module to enhance the PlanMyTrip app. Then, you'll have the beta testers use input statements to filter the data for their weather preferences, which will be used to identify potential travel destinations and nearby hotels. The beta tester will choose four cities from the list of potential travel destinations to create a travel itinerary. Finally, using the Google Maps Directions API, you will create a travel route between the four cities and a marker layer map.

What You're Creating

This new assignment consists of three technical analyses. You will submit the following deliverables:

- **Deliverable 1:** Retrieve Weather Data
- **Deliverable 2:** Create a Customer Travel Destinations Map
- **Deliverable 3:** Create a Travel Itinerary Map

Files

Use the following links to download the Challenge starter codes.

- Download Deliverable 1 starter code

(https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_6/Weather_Database_starter_code.ipynb)

- Download Deliverable 2 starter code

(https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_6/Vacation_Search_starter_code.ipynb)

- Download Deliverable 3 starter code

(https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_6/Vacation_Itinerary_starter_code.ipynb)

- Download backup data for Deliverables 2 and 3

(https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_6/WeatherPy_vacation-backup.csv)

Deliverable 1: Retrieve Weather Data (25 points)

Deliverable 1 Instructions

For this deliverable, you'll generate a set of 2,000 random latitudes and longitudes, retrieve the nearest city, and perform an API call with OpenWeatherMap. In addition to the city weather data you gathered in this module, use your API skills to retrieve the current weather description for each city. Then, create a new DataFrame containing the updated weather data.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 6.2.3:** Make an API call
- **Lesson 6.2.3:** Create a `config.py` file
- **Lesson 6.2.6:** Retrieve city weather data
- **Lesson 6.2.7:** Add the weather data to a DataFrame
- **Lesson 6.2.7:** Export the DataFrame to a CSV file

1. Create a folder called `Weather_Database` to save all the files related with this deliverable.
2. Save the `Weather_Database_starter_code.ipynb` starter code to the `Weather_Database` folder and rename it as `Weather_Database.ipynb`.
3. Use the `np.random.uniform` function to generate a new set of 2,000 random latitudes and 2,000 longitudes.
4. Use the `citipy` module to get the nearest city for each latitude and longitude combination.
5. Import your OpenWeatherMap's API key and assemble the API call URL as a string variable. Recall to edit the `config.py` file to add your API key; also, it's critical to avoid publishing your API key on your GitHub repository.
6. Retrieve the following information from the API call:
    - Latitude and longitude
    - Maximum temperature
    - Percent humidity
    - Percent cloudiness
    - Wind speed
    - Weather description (for example, clouds, fog, light rain, clear sky)
7. Add the weather data to a new DataFrame.
    - Before continuing to the next step, take a moment to confirm that the DataFrame looks similar to the image below (note that cities and countries may vary):

	City	Country	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Current Description
0	Mataura	NZ	-46.19	168.86	39.00	83	2	3.00	clear sky
1	Narsaq	GL	60.92	-46.05	44.60	75	75	2.24	shower rain
2	Aksu	CN	41.12	80.26	82.42	22	6	5.08	clear sky
3	Provideniya	RU	64.38	-173.30	40.84	68	0	0.69	clear sky
4	Bilibino	RU	68.05	166.44	54.82	53	100	4.29	overcast clouds

5	Barrow	US	71.29	-156.79	37.40	89	90	16.11	overcast clouds
6	Ribeira Grande	PT	38.52	-28.70	69.80	73	20	20.80	few clouds
7	Touros	BR	-5.20	-35.46	80.60	74	75	11.41	broken clouds
8	Hasaki	JP	35.73	140.83	66.20	100	75	4.70	light intensity drizzle rain
9	Hoquiam	US	46.98	-123.89	55.00	100	90	0.78	overcast clouds

8. Export the DataFrame as a CSV file, and save it as `WeatherPy_Database.csv` in the `Weather_Database` folder.

Deliverable 1 Requirements

You will earn a perfect score for Deliverable 1 by completing all requirements below:

- Retrieve all of the following information from the API call: **(15 pt)**
  - Latitude and longitude
  - Maximum temperature
  - Percent humidity
  - Percent cloudiness
  - Wind speed
  - Weather description (for example, clouds, fog, light rain, clear sky)

- Add the weather data to a new DataFrame **(5 pt)**
- Export the DataFrame as `WeatherPy_Database.csv` into the `Weather_Database` folder **(5 pt)**

Be sure to double-check that you have the following files in the `Weather_Database` folder:

- The `Weather_Database.ipynb` file.
- The `WeatherPy_Database.csv` file.

Deliverable 2: Create a Customer Travel Destinations Map (35 points)

Deliverable 2 Instructions

In this deliverable, you'll employ input statements to retrieve customer weather preferences. Next, you'll use those preferences to identify potential travel destinations and nearby hotels. Finally, you'll show those destinations on a marker layer map with pop-up markers.

REWIND

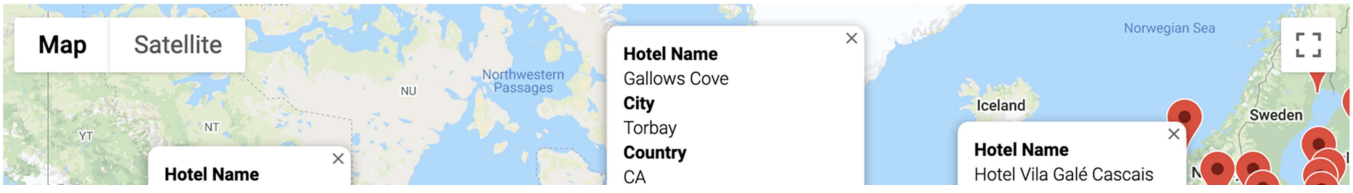
For this deliverable, you've already done the following in this module:

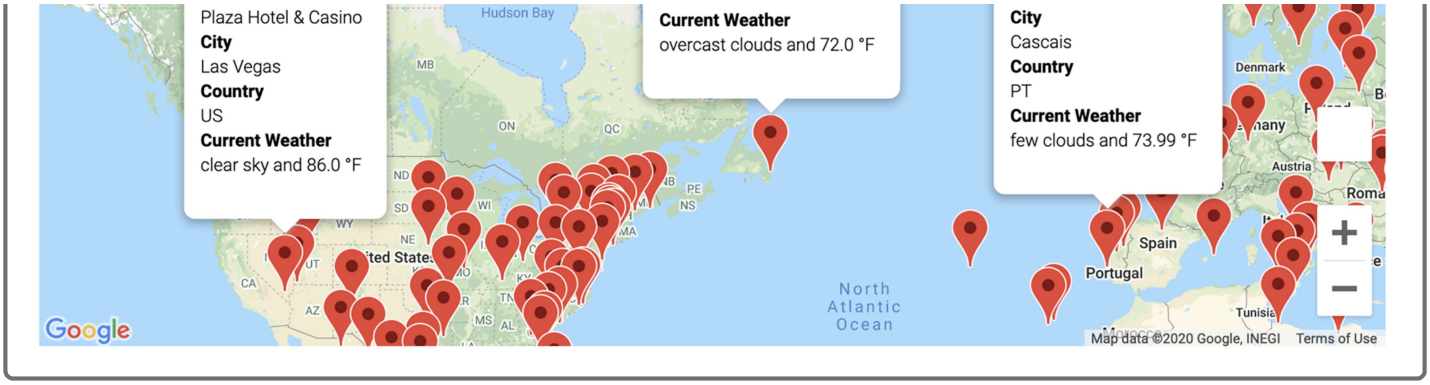
- **Lesson 6.2.7:** Add the weather data to a DataFrame
- **Lesson 6.2.7:** Add the `config.py` file to the `.gitignore` file
- **Lesson 6.5.3:** Create input statements
- **Lesson 6.5.3:** Filter a DataFrame using the `loc` method
- **Lesson 6.5.4:** Retrieve hotel data
- **Lesson 6.5.4:** Add data to a pop-up marker
- **Lesson 6.5.4:** Create a marker layer map

1. Create a folder called `Vacation_Search` to save all the files related with this deliverable.
2. Download the `Vacation_Search_starter_code.ipynb` Jupyter notebook,save it into your `Vacation_Search folder`, and rename it as `Vacation_Search.ipynb`.
3. In the `Vacation_Search.ipynb` file, ensure that the dependencies and the Google API key is imported correctly.
4. From the `Weather_Database` folder you created in the "Deliverable 1," import the `WeatherPy_Database.csv` file as a Pandas DataFrame named `city_data_df`.
5. Write two input statements that prompt the user to enter their minimum and maximum temperature criteria for their vacation.
6. Create a new Pandas DataFrame by using the `loc` Pandas method to filter the `city_data_df` DataFrame for temperature criteria collected. Name the DataFrame as `preferred_cities_df`.
7. Create a new Pandas DataFrame named `clean_travel_cities` by using the Pandas `dropna` function on the `preferred_cities_df` to drop any empty rows.
8. Use the `copy` Pandas function to create a new DataFrame, called `hotel_df`, by copying the following columns from the `clean_travel_cities` DataFrame: "City", "Country", "Max Temp", "Current Description", "Lat", "Lng".
9. Add a new empty column named `Hotel Name` to the `hotel_df` DataFrame.
10. Review the hotel search parameters provided. These parameters are the same we used in this module; you'll use them to search for a hotel for each city.
11. Use a for loop to iterate through the `hotel_df` DataFrame, retrieve the latitude and longitude of each city to find the nearest hotel based on the search parameters provided, then add the hotel name to the `hotel_df` DataFrame. If a hotel isn't found, skip to the next city.
12. Drop any rows in the `hotel_df` DataFrame where a hotel name is not found and store the resulting data into a new DataFrame named `clean_hotel_df`.
    - Take a moment to confirm that your `clean_hotel_df` DataFrame looks similar to the image below (note that the data may vary).

	City	Country	Max Temp	Current Description	Lat	Lng	Hotel Name
2	Aksu	CN	82.42	clear sky	41.12	80.26	Pudong Holiday Hotel
7	Touros	BR	80.60	broken clouds	-5.20	-35.46	INN NEW HORIZON
10	Morehead	US	75.20	clear sky	37.27	-87.18	CCI Express Inn
11	Port Elizabeth	ZA	84.20	clear sky	-33.92	25.57	39 On Nile Guest House
16	Northam	GB	82.40	few clouds	51.03	-4.22	Durrant House Hotel
18	Hyderabad	IN	87.01	haze	17.38	78.47	Taj Krishna, Hyderabad
20	Port Alfred	ZA	82.00	clear sky	-33.59	26.89	The Halyards Hotel
22	Atuona	PF	79.72	clear sky	-9.80	-139.03	Villa Enata
23	Kapaa	US	73.40	heavy intensity rain	22.08	-159.32	Sheraton Kauai Resort at Coconut Beach
25	Nikolskoye	RU	88.00	clear sky	59.70	30.79	Tourist House - Sablino

13. Create an CSV file to store the `clean_hotel_df` DataFrame as `WeatherPy_vacation.csv` in the Vacation_Search folder.
14. Review the formatting template provided that you'll use to add an information box to each marker in the map. In the pop-up for each city you'll add:
    1. The city name.
    2. The country code.
    3. The weather description and maximum temperature for the city.
15. Review the provided list comprehension code to retrieve the city data from each row, which will then be added to the formatting template and saved in the `hotel_info` list.
16. Use the provided code snippet to retrieve the latitude and longitude from each row and store them in a new DataFrame called `locations`.
17. Refactor your previous marker layer map code to create a marker layer map that will have pop-up markers for each city on the map.
18. Take a screenshot of your map and save it to the `Vacation_Search` folder as `WeatherPy_vacation_map.png`.
    - The marker layer map with a pop-up marker for each city should look similar to the following image:





Deliverable 2 Requirements

You will earn a perfect score for Deliverable 2 by completing all requirements below:

- Input statements are written to prompt the customer for their minimum and maximum temperature preferences. **(5 pt)**
- A new DataFrame is created based on the minimum and maximum temperature, and empty rows are dropped. **(5 pt)**
- The hotel name is retrieved and added to the DataFrame, and the rows that don't have a hotel name are dropped. **(10 pt)**
- The DataFrame is exported as a CSV file into the Vacation_Search folder and is saved as `WeatherPy_vacation.csv`. **(5 pt)**
- A marker layer map with pop-up markers for the cities in the vacation DataFrame is created, and it is uploaded as a PNG. Each marker has the following information: **(5 pt)**
  - Hotel name
  - City
  - Country
  - Current weather description with the maximum temperature
- The marker layer map is saved and uploaded to the Vacation_Search folder as `WeatherPy_vacation_map.png`. **(5 pt)**

Be sure to double-check that you have the following in the Vacation_Search folder:

- The `Vacation_Search.ipynb` file.
- The `WeatherPy_vacation.csv` file.
- The `WeatherPy_vacation_map.png` image.

Deliverable 3: Create a Travel Itinerary Map (40 points)

Deliverable 3 Instructions

For this deliverable, you'll use the Google Directions API to create a travel itinerary that shows the route between four cities chosen from the customer's possible travel destinations. Then, you'll create a marker layer map with a pop-up

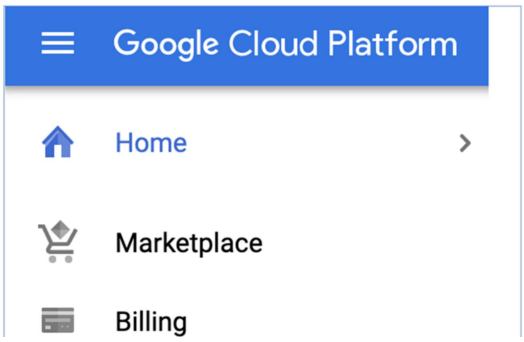
cities chosen from the customer's possible travel destinations. Then, you'll create a marker layer map with a pop-up marker for each city on the itinerary.

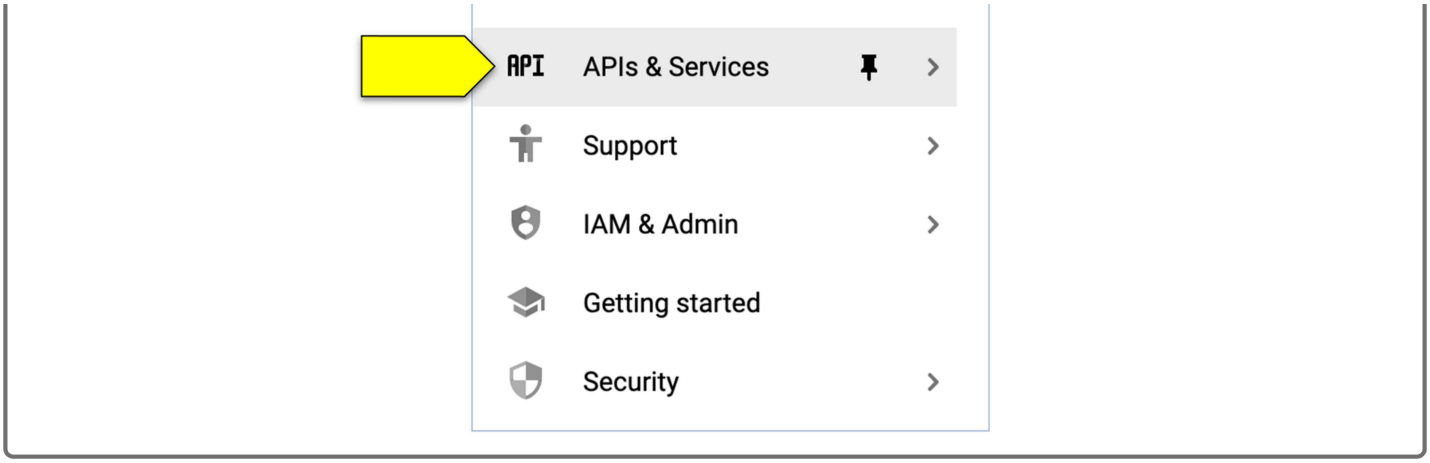
REWIND

For this deliverable, you've already done the following in this module:

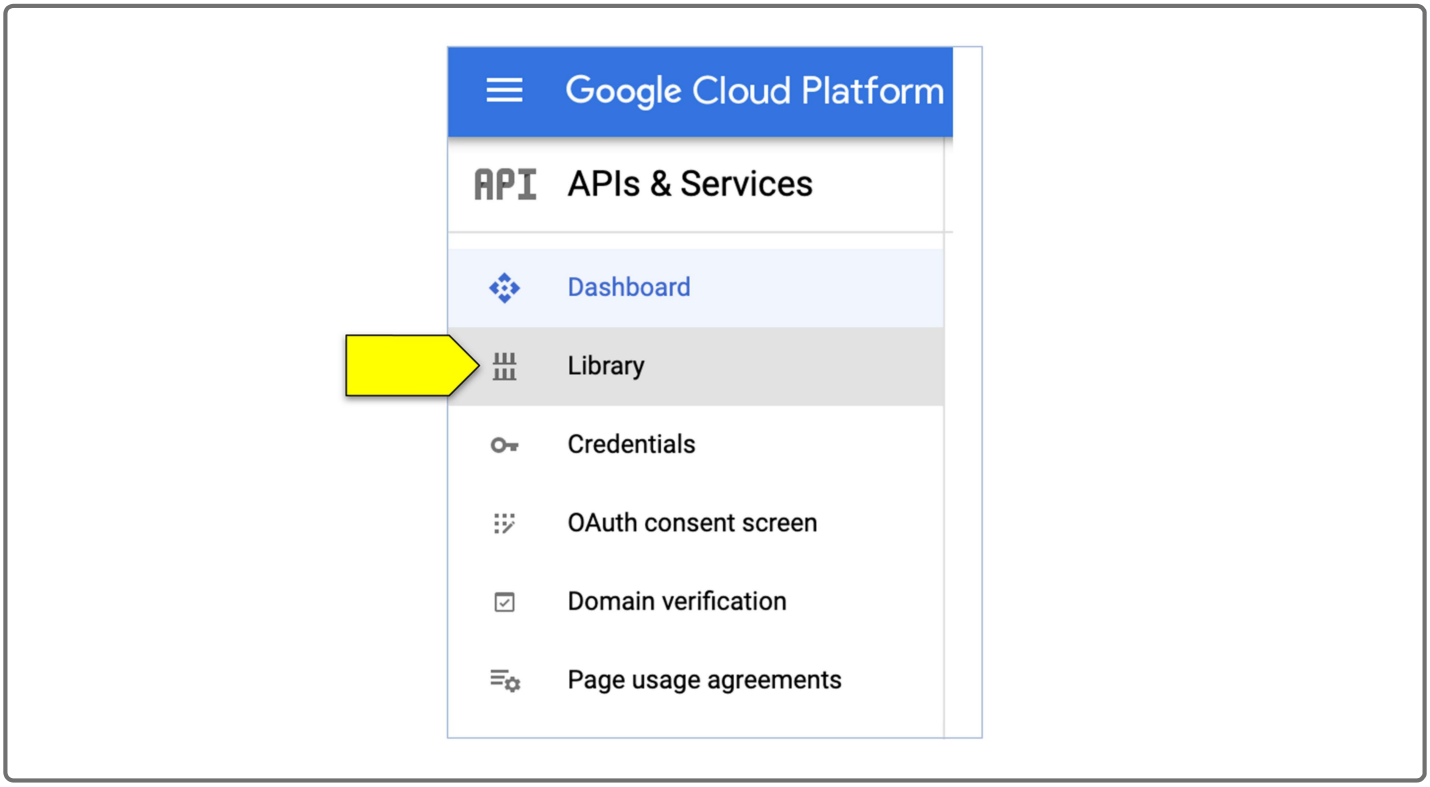
- **Lesson 6.5.4:** Add data to a pop-up marker
- **Lesson 6.5.4:** Create a marker layer map

1. Enable the "Directions API" in your Google account for your API key.
    - On the Google Cloud Platform, select "APIs & Services" from the left-hand side

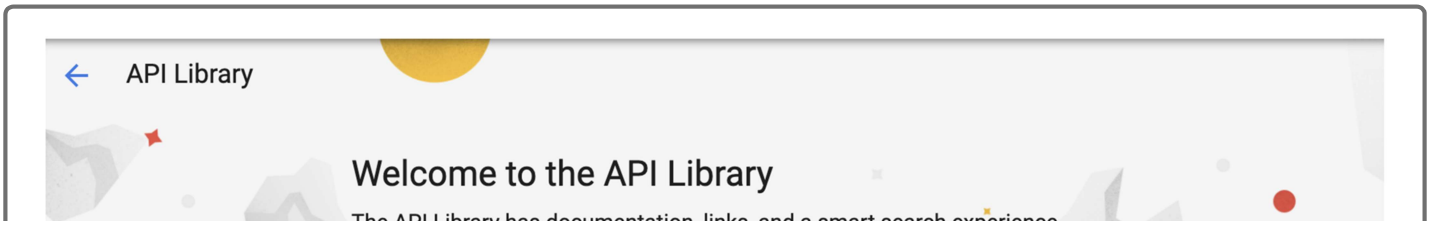


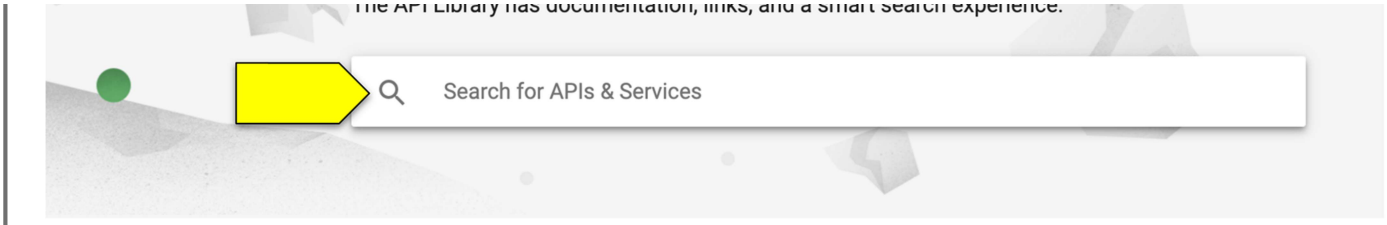


- Then, select "Library".

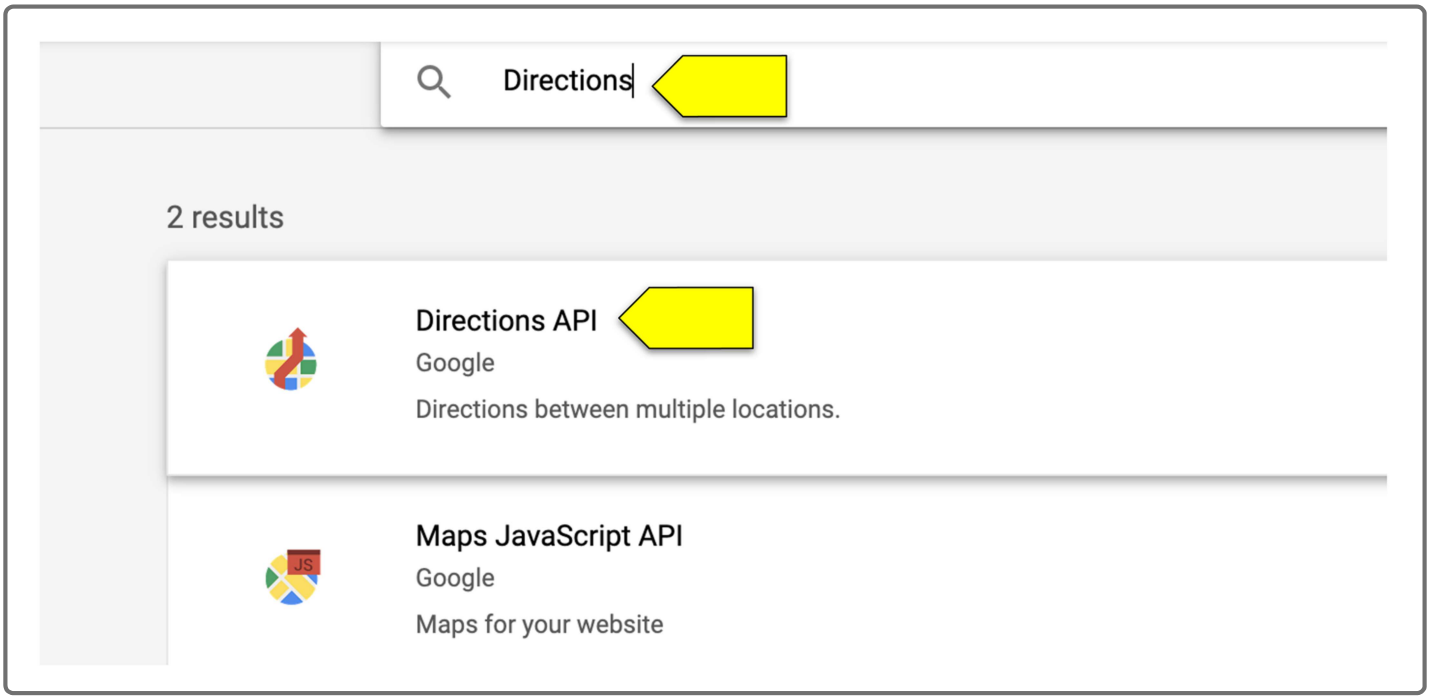


- In the Search field, type "Directions".

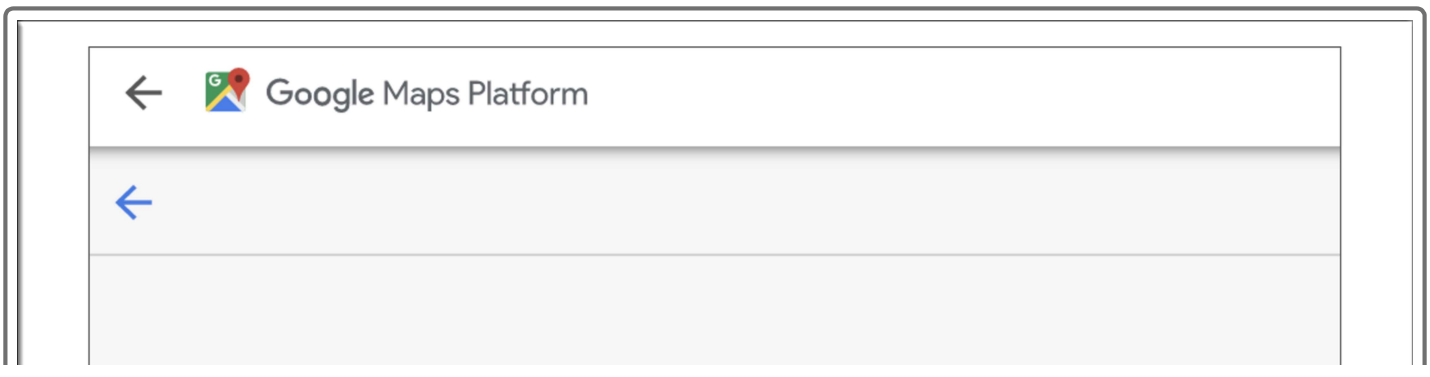


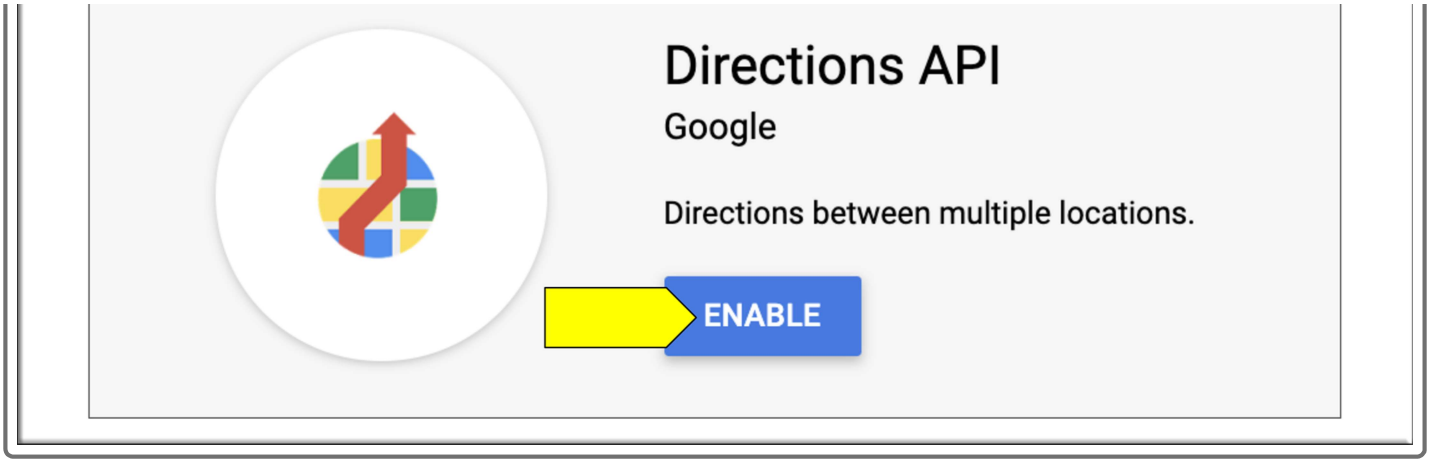


- Select "Directions API".



- Click "Enable" to activate the Directions API.





2. Create a folder called `Vacation_Itinerary` to store all the files for this deliverable.
3. Download the `Vacation_Itinerary_starter_code.ipynb` file into your `Vacation_Itinerary` folder and rename it `Vacation_Itinerary.ipynb`.
4. Make sure the initial dependencies and the Google API key are imported.
5. From your `Vacation_Search` folder from Deliverable 2, import the `WeatherPy_vacation.csv` file as a DataFrame named `vacation_df`.
6. In this step, you will set-up the pop-up markers. Review the code to create a marker layer map of the vacation search results. This code is the same as in Deliverable 2.
7. From the vacation search map, *choose four cities* that a customer might want to visit. They should be close together and in the same country. Use the variables we have provided and the `loc` method to create separate DataFrames for each city on the travel route.

HIDE HINT

You will start and end the route in the same city, so the `vacation_start` and `vacation_end` DataFrames will be t same city.

8. Use the `to_numpy()` function and list indexing to write code to retrieve the latitude-longitude pairs as tuples from each city DataFrame.

   If you'd like a hint on using the `to_numpy()` function with list indexing, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

HIDE HINT

The `to_numpy()` function can be used on a DataFrame column to convert the column data into an array. List indexing can be used on the array to retrieve each item in the array.

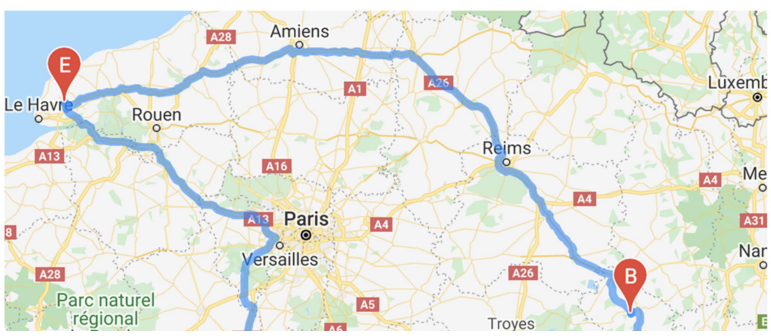
Check out this Pandas documentation

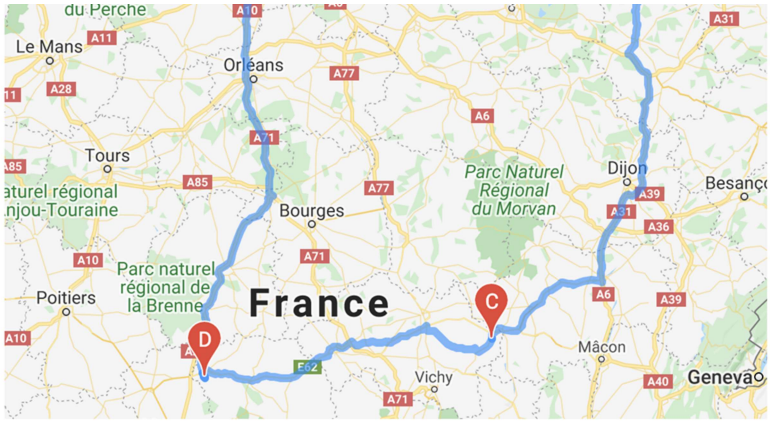
(https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.to_numpy.html#pandas.DataFrame.to_num for an explanation of how to use `to_numpy()` function.

9. Use the gmaps documentation

(https://jupyter-gmaps.readthedocs.io/en/latest/tutorial.html#directions-layer) to create a directions layer map using the variables from the previous step. Where the starting and ending city are the same city, the `waypoints` are the three other cities, and the `travel_mode` is either `"DRIVING"`, `"BICYCLING"`, or `"WALKING"`.

10. Take a screenshot of your map and save it to the `Vacation_Itinerary` folder as `WeatherPy_travel_map.png`.
    - The directions layer map should look similar to the following image:





11. Use the provided `concat()` function code snippet to combine the four separate city DataFrames into one DataFrame.

    If you'd like a hint on combining DataFrames into one DataFrame using the `concat()` function, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

HIDE HINT

The `concat()` function can be used to combine multiple DataFrames that have the same columns (axis). In addition, Pandas concatenation preserves indices. To reset the index in the concatenated DataFrame, we can us the `ignore_index` parameter and set it equal to `True`.

Check out this Pandas documentation

(https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.concat.html) for an explanation of how to use the `concat()` function.

12. Refactor the code from Step 6 to create a new marker layer map of the cities on the travel route.
13. Take a screenshot of your map and save it to the `Vacation_Itinerary` folder as `WeatherPy_travel_map_markers.png`.
    - The marker layer map with a pop-up marker for each city should look similar to the following image:

Deliverable 3 Requirements

You will earn a perfect score for Deliverable 3 by completing all requirements below:

- Four DataFrames are created, one for each city on the itinerary. **(10 pt)**
- The latitude and longitude pairs for each of the four cities are retrieved. **(5 pt)**
- A directions layer map between the cities and the travel map is created and uploaded as `WeatherPy_travel_map.png`. **(10 pt)**
- A DataFrame that contains the **four** cities on the itinerary is created. **(10 pt)**
- A marker layer map with a pop-up marker for the cities on the itinerary is created, and it is uploaded as `WeatherPy_travel_map_markers.png`. Each marker has the following information: **(5 pt)**
  - Hotel name
  - City
  - Country
  - Current weather description with the maximum temperature

Be sure to double-check that you have the following in the Vacation_Itinerary folder:

- The `Vacation_Itinerary.ipynb` file.
- The `WeatherPy_travel_map.png` image.
- The `WeatherPy_travel_map_markers.png` image.

Submission

Once you're ready to submit, make sure to check your work against the rubric to ensure you are meeting the requirements for this Challenge one final time. It's easy to overlook items when you're in the zone!

As a reminder, the deliverables for this Challenge are as follows:

- Deliverable 1: Retrieve Weather Data
- Deliverable 2: Create a Customer Travel Destinations Map
- Deliverable 3: Create a Travel Itinerary Map

Upload the following to your WeatherPy GitHub repository:

- The Weather_Database folder with the following:
  - The `Weather_Database.ipynb` file
  - The `WeatherPy_Database.csv` file

- The Vacation_Search folder with the following:
  - The `Vacation_Search.ipynb` file
  - The `WeatherPy_vacation.csv` file
  - The `WeatherPy_vacation_map.png` image

- The Vacation_Itinerary folder with the following:
  - The `Vacation_Itinerary.ipynb` file
  - The `WeatherPy_travel_map.png` image
  - The `WeatherPy_travel_map_markers.png` image

- A README.md that describes the purpose of the repository. Although there is no graded written analysis for this challenge, it is encouraged and good practice to add a brief description of your project.

IMPORTANT

Do not include your `config.py` file in your submission.

If you'd like a hint on how to not include the `config.py` file when adding your files to your GitHub repository, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

HIDE HINT

To prevent GitHub from tracking and adding the `config.py` file to your GitHub repository, revisit Lesson 6.2.7:

To prevent GitHub from tracking and adding the `config.py` file to your GitHub repository, revisit **Lesson 6.2.7**.

Create a DataFrame of city weather data.

To submit your challenge assignment for grading in Bootcamp Spot, click Start Assignment, click the Website URL tab, then provide the URL of your WeatherPy GitHub repository, and then click Submit. Comments are disabled for graded submissions in BootCampSpot. If you have questions about your feedback, please notify your instructional staff or the Student Success Manager. If you would like to resubmit your work for an improved grade, you can use the **Re-Submit Assignment** button to upload new links. You may resubmit up to 3 times for a total of 4 submissions.

IMPORTANT

Once you receive feedback on your Challenge, make any suggested updates or adjustments to your work. Then, add this week's Challenge to your professional portfolio.

NOTE

You are allowed to miss up to two Challenge assignments and still earn your certificate. If you complete all Challenge assignments, your lowest two grades will be dropped. If you wish to skip this assignment, click Next, and move on to the next Module.