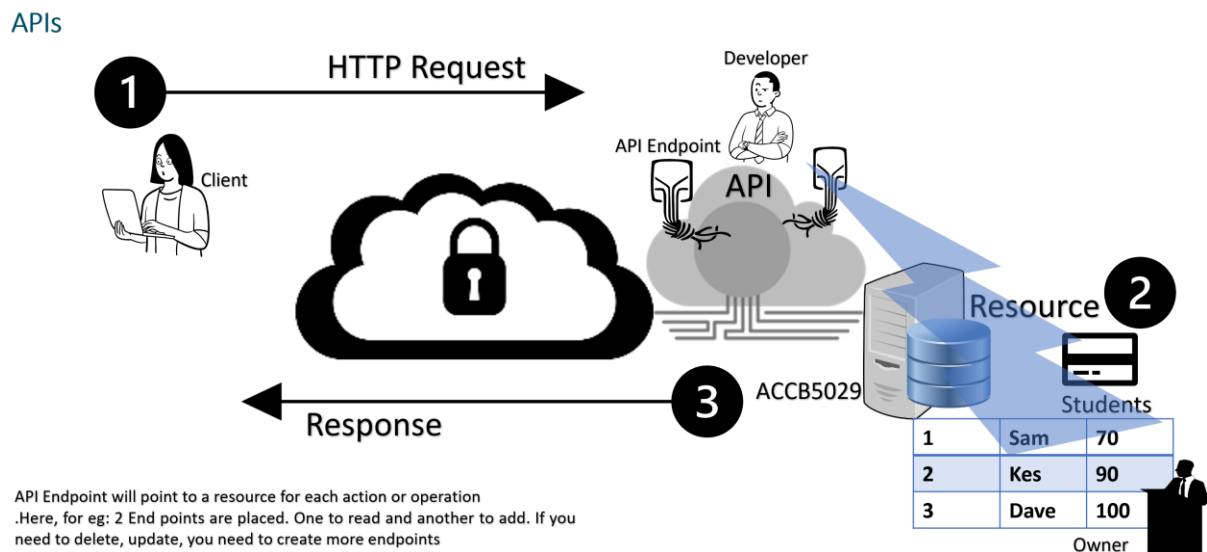
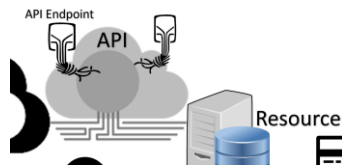


Integrating API with Python Code – Demo and In-Class Activity

Note: The links provided in this document may no longer be active. As part of best practices for industry security, links are regularly updated. If a link is unavailable, please try searching using the relevant keywords provided.

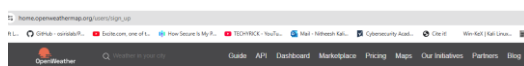


1. Setting up API



In this example, we are using the OpenWeather app (<https://openweathermap.org>). To access its features, you'll need to create an account. However, you are welcome to use any other open-source or freely available applications.

+ Create OpenWeather Account and API Keys



Create New Account

Username

Email

☐ I am 16 years old and over

☒ I agree with Privacy Policy, Terms and conditions of use and Cookies Policy

I consent to receive communications from OpenWeather Group of Companies and their partners:

☐ System news (API usage alert, system update, temporary system shutdown, etc)

☐ Product news (change to price, new product features, etc)

☐ Corporate news (our life, the launch of a new service, etc)

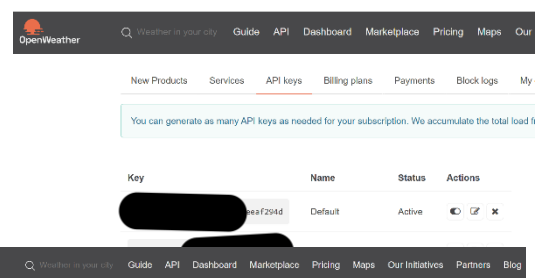
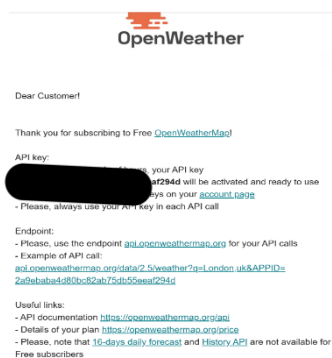
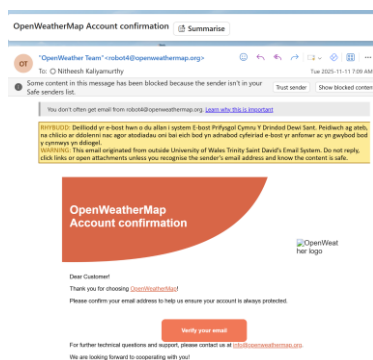
How and where will you use our API?

Hi! We are doing some housekeeping around thousands of our customers. Your impact will be much appreciated. All you need to do is to choose in which exact area you use our services.

Company: University of Wales TSD

Purpose: Education/Science

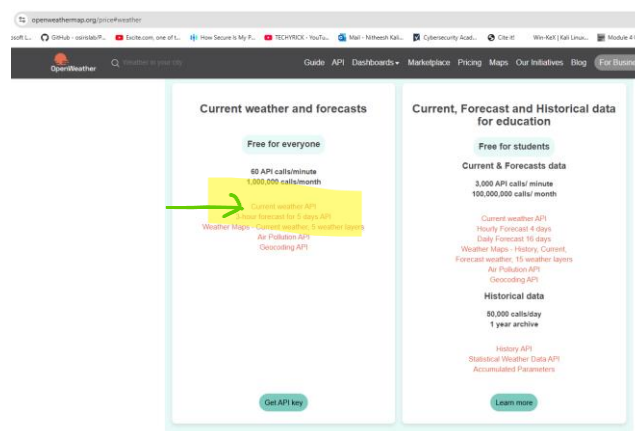
Cancel Save



+ Understand OpenWeather API

- The API documentation is located at <https://openweathermap.org/api>
- There, you'll find various APIs they offer like
 - Current Weather Data,
 - 5 Day / 3 Hour Forecast,
 - Air Pollution etc.
- Find the Current Weather Data API:
 - This section will guide you on how to create URLs for different cities, along with necessary parameters

<https://openweathermap.org/current>



- The standard endpoint for fetching current weather by city name is:

openweathermap.org/current

Call current weather data

How to make an API call

API call

```
https://api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}&appid={API key}
```

Parameters

Parameter	Required	Description
lat	required	Latitude. If you need the geocoder to automatic convert city names and zip-codes to geo coordinates and the other way around, please use our Geocoding API
lon	required	Longitude. If you need the geocoder to automatic convert city names and zip-codes to geo coordinates and the other way around, please use our Geocoding API
appid	required	Your unique API key (you can always find it on your account page under the "API key" tab)
mode	optional	Response format. Possible values are <code>xml</code> and <code>html</code> . If you don't use the <code>mode</code> parameter format is JSON by default. Learn

- The query parameters are added to this endpoint to specify details like: City name (Swansea), Your API key, Measurement units etc
- Based on the documentation, the request URL for Swansea, with metric units and an API key:
`http://api.openweathermap.org/data/2.5/weather?q=Swansea&appid=YOUR_API_KEY&units=metric`
Replace YOUR_API_KEY with your actual OpenWeather API key.

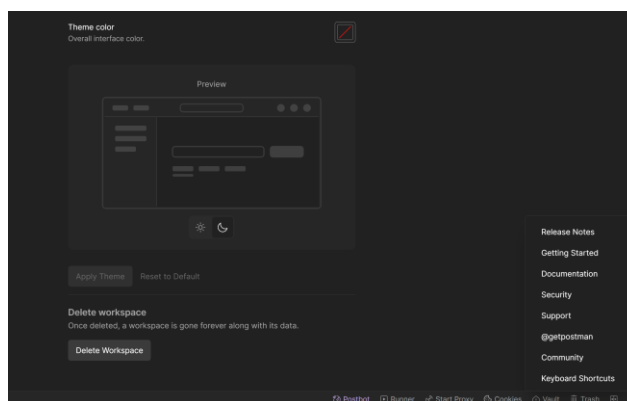
2. Understanding the Request

+ Postman

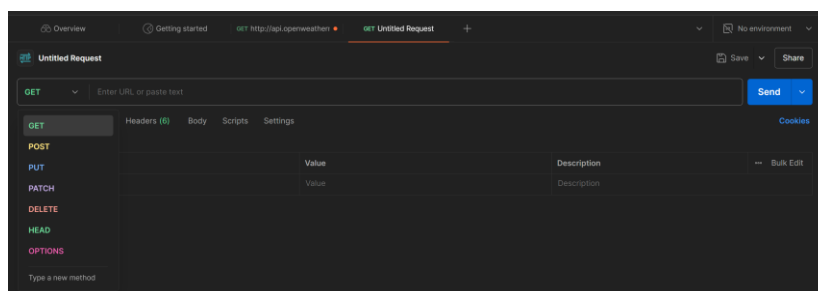
A tool used for testing APIs (Application Programming Interfaces).

It provides a user-friendly interface to send HTTP requests to APIs, receive and view responses, and even automate and document API calls.

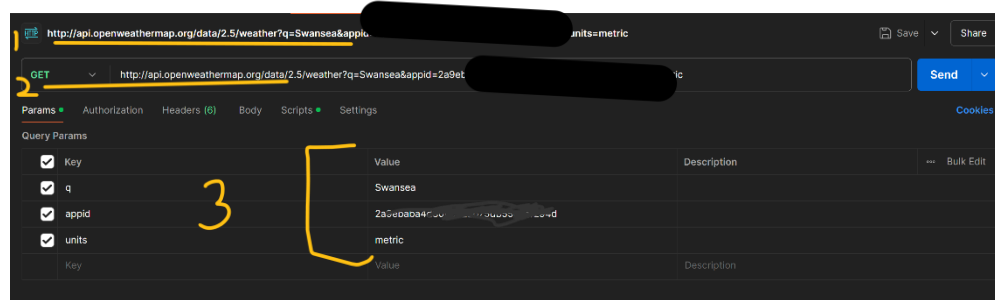
- Install Postman (Ignore if already installed – DevASC has Postman already)
- Bottom Right, click on the (?) Icon and read the Documentation to further understand Postman.



- Click the New Request (+) icon

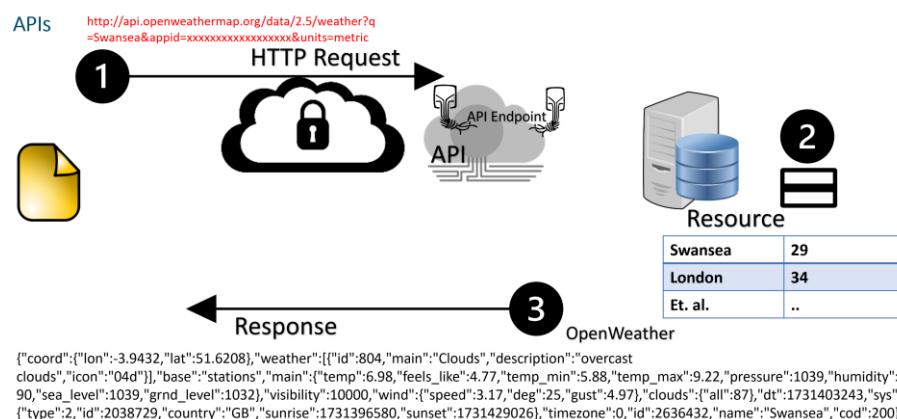


- Enter your URL and select appropriate Request. Here, in our case it is a get request.



1. HTTP
2. GET
3. Parameters

- The response sent by the API is



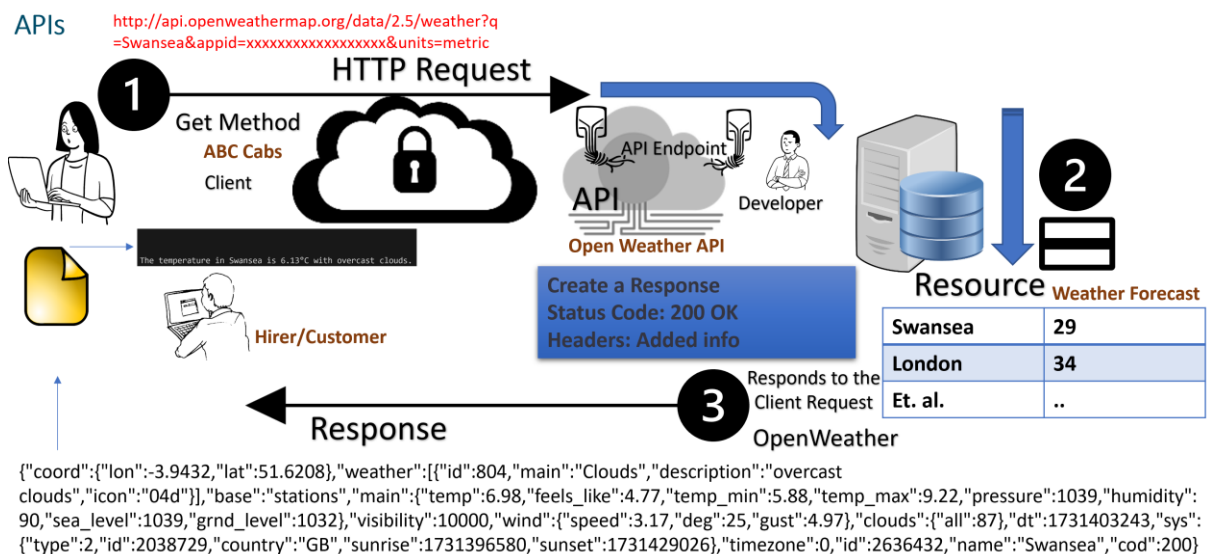
- The OpenWeather API responds in **JSON** format, a structured text format ideal for data exchange between systems. JSON organizes data in key-value pairs.

3. Write a Python Script to Parse it and make it presentable as per your customer's requirements.

- + Refer to the Python Script, (from Moodle – Week 5 Tab): Executing that file will give you the following output.

```
The temperature in Swansea is 6.13°C with overcast clouds.
```

- + In this code,
 - When you receive the API response, Python interprets the JSON data as a dictionary, allowing access to the values through keys
 - `data["main"]["temp"]`).
 - In your Python code, you're accessing specific pieces of information within this JSON dictionary and combining them into a formatted string.
 - Python takes the JSON data, extracts relevant information, and formats it using f-strings or other string formatting methods to create a user-friendly, readable sentence.
 - Extract data
 - `temp = data["main"]["temp"]`
 - `description = data["weather"][0]["description"]`
 - Format as a human-readable string
 - `print(f"The temperature in Swansea is {temp}°C with {description}.")`
- + And the Final Outcome is



4. Refactor the Python Script to get the Input from the user, fetch and display Weather Data for multiple cities

- + Task
 - Asks the user to input multiple city names.
 - Fetches and displays the weather data for each city entered.
 - Format the output nicely.

- Handle cases where a city isn't found by printing a friendly error message.