Task 2: API Testing

For the sake of accuracy (Personal created code vs professionally built software) I shall be employing a code I have created to send an API request with the provided information and interpret the Response code/ JSON data.

Information:

* Personal code platform – Python via pycharm, available on repository
* <https://github.com/LegendGT86/Interview/blob/LegendGT86-patch-1/interview%20retry.py>
* API Key: afca0dc69110a51fa8cc681e4ea6f2ed
* Locations : Cape town RSA (-33.9249, 18.4241) and London UK (51.5072,-0.1276)
* URL: <https://api.openweathermap.org/>
* Platform: Win 10 HE, Google Chrome Version 95.0.4638.54 (Official Build) (64-bit)
* Secondary software tester: Postman Version 9.1.2

ID: Weat-API

Summary: Ensure API works are required and current weather information from both locations is correctly retrieved

Steps:

1. Open personal code in Pycharm
2. Confirm Lat/Long of CapeTown (<https://www.latlong.net/place/cape-town-south-africa-854.html>) and enter variable values
3. Confirm Lat/Long of London (https://latitude.to/map/gb/united-kingdom/cities/london) and enter variable values
4. Run code which will send requests with required endpoints and receive according responses in JSON format stored as 2 variables.
5. Code will ascertain status code of response and parse data, printing out information pertaining to current weather in both locations
6. Similar data will be inserted into the input fields in Postman app (URL/Co ordinates/Query Parameters)
7. Data is then compared to confirm accurate retrieval
8. API Validation (event speed and data transmission) will also be noted
9. The test will be repeated for Weather data as per a designated time period (hour/day/minute) using the above steps
10. As a confirmation method, the weather for the two locations as per 2 additional weather sites (<https://www.ibm.com>) ( <https://www.accuweather.com>) as well as the primary website (<https://www.openweather.org>) to confirm findings

Expected results:

* The weather results would be consistent with a very small margin of deviation based on weather testing equipment/techniques. The API would return a 200 response and provide all the necessary information in JSON format. Retrieval times would be consistent

Actual result:

* Data was as expected, consistent in time and correct across the board

Status:

* Pass

Optimization Notes:

* Collections can be made of all the inspected sites to reduce loading and retrieval time
* Personal code can be altered to create a dictionary containing central location keys and co-ordinate values for quicker retrieval and implementation
* A Loop function can be implemented in the code with a timer function to ascertain average load times per cycle
* A secondary valid API key can be obtained to further verify the data of the current API Key