RESTful API Project

API Specification Document

Course: INFO U762 – 01F

Semester: Spring 2024

Instructor: Dr. Grover Walters

Date: April 22, 2024

David C. Miller

**Weather API**

* My Weather API is using the National Weather Service API located at [API Web Service (weather.gov)](https://www.weather.gov/documentation/services-web-api#/default/station_observation_latest).
* The base API URL is: [https://api.weather.gov/stations/{stationId}/observations/latest](https://api.weather.gov/stations/%7bstationId%7d/observations/latest)
* The station ID that I am using is: KCQT, with is Los Angeles.
* The complete API URL is: <https://api.weather.gov/stations/KCQT/observations/latest?require_qc=false>
* The data return is GEOJSON as a Dictionary data type.
* Python Code Sections:

**API Pull:**

api\_url = "https://api.weather.gov/stations/KCQT/observations/latest?require\_qc=false"

data = fetch\_data\_from\_api(api\_url)

**Converting data from GeoJSON to GeoDataFrame to a List to be Queried:**

gdf = geopandas.GeoDataFrame.from\_features([data])

gdf = gdf.\_\_geo\_interface\_\_

values\_list = list(gdf.values())

**Extracting Current Conditions String Value from List:**

tree\_obj = objectpath.Tree(values\_list)

CurrentCondition = tuple(tree\_obj.execute('$..textDescription'))

CurrentConditionValue = CurrentCondition[0]

print(CurrentConditionValue)

**Extracting Current Temperature and converting to Float Value:**

tree\_obj = objectpath.Tree(values\_list)

temp\_obj = tuple(tree\_obj.execute('$..temperature'))

temp\_value = str(temp\_obj)

tempC = temp\_value[39:]

tempC = tempC[:4]

tempC = float(tempC)

print(tempC)

**Converting the Temperature from Celsius to Fahrenheit and rounding to a single decimal point:**

tempF = round((tempC \* (9/5)) + 32, 1)

print(tempF)

The final values that will be passed on to the custom API will be **tempF** and **CurrentConditionValue**.

**Currency API**

* My currency API is using the ExchangeRate-API Free Currency API – [ExchangeRate-API - Open Access, No Key Required](https://www.exchangerate-api.com/docs/free)
* The API link provides a complete list of foreign currencies all calculated against the US Dollar.
* The API URL is : <https://open.er-api.com/v6/latest/USD>
* The data returns as JSON as a Dictionary data type.
* Value was returned by nested dictionary query.
* Python Code Sections:

**API Pull:**

api\_url = "https://open.er-api.com/v6/latest/USD"

data = fetch\_data\_from\_api(api\_url)

**Extract the Euro conversation rate for one US Dollar. Setting a hypothetical amount of 1000 Euro’s to be converted and rounding result to 2 decimal places:**

rates\_data = {}

EUR = 1000

USD = 0

For k in data.keys():

if 'rates' in k:

rates\_data[k] = data[k]

EUREXCH = rates\_data['rates']['EUR']

USD = round(EUR \* EUREXCH, 2)

print(USD)

The final value that will be passed on to the custom API will be **USD**.

**Brewery API**

* My brewery API is using the Open Brewery DB API – [Open Brewery DB | Documentation](https://www.openbrewerydb.org/documentation/#search-breweries)
* The API link provides a list of over 8250 breweries from around the world.
* The API URL is : <https://api.openbrewerydb.org/v1/breweries/search?query=los%20angeles>
* The data returns as JSON as a List data type.
* Python Code Sections:

**API Pull:**

api\_url = "https://api.openbrewerydb.org/v1/breweries/search?query=los%20angeles"

data = fetch\_data\_from\_api(api\_url)

**Code to select BrewPubs, which have beer and food, and eliminate BrewPubs that do not have website or phone numbers listed:**

df = pd.DataFrame(data)

df.to\_csv("brewery.csv", index = False)

df\_1 = df[df['website\_url'].notna()]

df\_2 = df\_1[df\_1['phone'].notna()]

df\_3 = df\_2[df\_2['brewery\_type'] == 'brewpub']

**Code to randomly select a single entry from the results returned:**

df\_4 = df\_3.sample(n=1)

**Extracted the relevant values to pass on:**

BrewName = df\_4.iat[0,1]

BrewPhone = df\_4.iat[0,12]

BrewURL = df\_4.iat[0,13]

The final values that will be passed on to the custom API will be **BrewName**, **BrewPhone**, and **BrewURL**.