Contents

1 Project Title

Weather Forecast Application

2 Project Code

I had some issues running my replit account. In order not to waste so much time dwelling on how to fix it, I ended up using jupyter lab so therefore, a screenshot of the code is shown below;

```
1]: pip install tabulate

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: tabulate in c:\programdata\anaconda3\anaconda\lib\site-packages (0.8.10)
Note: you may need to restart the kernel to use updated packages.
```

```
4]: import requests
from tabulate import tabulate
#error handling
try:
    #requesting user to input latitude & longitude of city's weather forecast they want to display
    latitude = float(input("Enter latitude: "))#input("Enter Latitude: ")
    longitude = float(input("Enter longitude: "))#input("Enter Longitude: ")
    #api showing the maximum and minimun temperature weather forecast for the next 7 days
    request_url = f"https://api.open-meteo.com/v1/forecast?latitude={latitude}&longitude={longitude}&daily=temperature_2m_max,temperature_2m_min&timezone=auto"

    response = requests.get(request_url)
    data = response.json()
    daily_data = data["daily"]
    #display weather data in a table
    table_data = list(zip(daily_data["time"], daily_data["temperature_2m_max"], daily_data["temperature_2m_min"]))
    headers = ["Days", "Max Temperature", "Min Temperature"]
    print(tabulate(table_data, headers, tablefmt="grid"))
except Exception as error:
    print("Invalid Location")
    print(error)
```

```
Enter latitude:  40.7128
Enter longitude:  -74.0060
+------------+-----------------+-----------------+
| Days       |   Max Temperature |   Min Temperature |
+============+=================+=================+
| 2025-02-20 |             0.1 |            -8.3 |
+------------+-----------------+-----------------+
| 2025-02-21 |            -1   |            -7.3 |
+------------+-----------------+-----------------+
| 2025-02-22 |             3   |            -7.3 |
+------------+-----------------+-----------------+
| 2025-02-23 |             5.7 |            -0.4 |
+------------+-----------------+-----------------+
| 2025-02-24 |             9.1 |             1.3 |
+------------+-----------------+-----------------+
| 2025-02-25 |             7.1 |             3.7 |
+------------+-----------------+-----------------+
| 2025-02-26 |             8.4 |             3.2 |
+------------+-----------------+-----------------+
```

3 Description

Using latitude and longitude, this Python script retrieves and shows the seven-day weather prediction, including the maximum and minimum temperatures, for a user-specified location. The tabulate package is used to format the output into a legible table when the data is obtained from the Open-Meteo API. As seen in the screenshot above, I made sure to install tabulate to display data in tabular form.

Key Features:

- Utilises an API request to obtain real-time weather data.
- Displays results in a formatted table for improved readability.
- Uses error handling to handle invalid inputs or request failures.
- Allows users to input geographic coordinates (latitude and longitude) to retrieve weather data.

4 Technical Details

Libraries used:
Requests -- Responds to HTTP requests to retrieve meteorological information.
tabulate -- Creates a table with weather data and formats it.
Integrated features:
input() -- Takes latitude and longitude input from the user.
float() -- Transforms input into numbers.
Potential errors (invalid input, API failures, etc.) are handled by the try-except function.

API Usage:
Weather Data API**:** Open-Meteo's weather forecast API.
Request URL Format
Response Structure (JSON Format)
The script extracts the "time", "temperature_2m_max", and "temperature_2m_min" values and displays them in a grid table.

Error Handling:

- If the user enters non-numeric latitude/longitude, a ValueError occurs.
- If the API fails or returns an unexpected response, an exception is caught.
- Errors print "Invalid Location" along with the actual error message.

5 Improved Project Code

With the use of Geocoding API to fetch city coordinates instead of the use of latitude and longitude, all you need to do is input the city.

```
import requests
from tabulate import tabulate

def get_coordinates(city):
    """Fetch latitude and longitude of a city using Open-Meteo Geocoding API."""
    try:
        geo_url = f"https://geocoding-api.open-meteo.com/v1/search?name={city}&count=1&language=en&format=json"
        response = requests.get(geo_url)

        if response.status_code != 200:
            print("Error fetching city coordinates!")
            return None, None

        weather_data = response.json()

        if "results" not in weather_data or len(weather_data["results"]) == 0:
            print(f"City '{city}' not found!")
            return None, None

        city_data = weather_data["results"][0]
        return city_data["latitude"], city_data["longitude"]

    except requests.exceptions.RequestException as e:
        print("Network error while fetching coordinates:", e)
        return None, None

def get_weather_forecast(city):
    """Fetch and display weather forecast for a given city."""
    latitude, longitude = get_coordinates(city)

    if latitude is None or longitude is None:
        return

    try:
        # Fetch weather data using coordinates
        weather_url = f"https://api.open-meteo.com/v1/forecast?latitude={latitude}&longitude={longitude}&daily=temperature_2m_max,temperature_2m_min&timezone=auto"
        response = requests.get(weather_url)

        if response.status_code != 200:
            print(f"API request failed! Status code: {response.status_code}")
            return

        weather_data = response.json()

        if "daily" not in data:
            print("Invalid API response: 'daily' data not found")
```

```
        if "daily" not in data:
            print("Invalid API response: 'daily' data not found")
            return

        daily_data = weather_data["daily"]
        table_data = list(zip(daily_data["time"], daily_data["temperature_2m_max"], daily_data["temperature_2m_min"]))
        headers = ["Days", "Max Temperature", "Min Temperature"]

        print(f"\nWeather forecast for {city.capitalize()}:\n")
        print(tabulate(table_data, headers, tablefmt="grid"))

    except requests.exceptions.RequestException as e:
        print("Network error:", e)
    except KeyError as e:
        print(f"Unexpected API response structure. Missing key: {e}")
    except Exception as error:
        print("An unexpected error occurred:", error)

# Get user input
city_name = input("Enter city name: ").strip()
get_weather_forecast(city_name)
```

```
Enter city name:  lagos

Weather forecast for Lagos:

+------------+------------------+------------------+
| Days       |   Max Temperature |   Min Temperature |
+============+==================+==================+
| 2025-02-25 |             32.5 |             28   |
+------------+------------------+------------------+
| 2025-02-26 |             31.2 |             27.1 |
+------------+------------------+------------------+
| 2025-02-27 |             32.7 |             27.5 |
+------------+------------------+------------------+
| 2025-02-28 |             32.7 |             28   |
+------------+------------------+------------------+
| 2025-03-01 |             32.6 |             28   |
+------------+------------------+------------------+
| 2025-03-02 |             33.1 |             28.1 |
+------------+------------------+------------------+
| 2025-03-03 |             32.6 |             28.1 |
+------------+------------------+------------------+
```

6 Error Handling

- Invalid City Name - Prints "City '<name>' not found!"
- Network Errors - Catches requests.exceptions.RequestException
- API Response Issues - Handles missing keys using KeyError
- Invalid JSON Response - Prints "Unexpected API response structure".