Samaritans Sr. Sec. School

City Weather Finder: A Python Program for Real-Time Weather Updates

Submitted by: Aaroh Solanki & Naitik Boriya

Submitted to: Miss Hema Ma'am

Session: 2024-25

Certificate

This is to certify that the project titled "Real-Time Weather Display Program" has been successfully completed by Aaroh Solanki and Naitik Boriya, students of Class 12th Diamond, under the guidance of Miss Hema Ma'am.

This project is submitted as part of the academic requirements for the school year 2024-25. The work presented in this project is an original effort of the students, demonstrating their understanding of Python programming and its application in retrieving real-time weather data.

We acknowledge their dedication and hard work in completing this project and wish them success in their future endeavors.

Approved by:
Miss Hema Ma'am

Principal's Signature:

Submitted by:

Aaroh Solanki (Student)

Naitik Boriya (Student)

Table Of Content

Acknowledgement

I would like to express my sincere gratitude to everyone who supported me in completing this project. I am especially grateful to our Principal, for creating a positive learning environment and encouraging us to pursue knowledge beyond the classroom.

Special thanks to my teacher, Miss Hema Ma'am, for their guidance and encouragement, which helped me gain valuable insights into Python programming and project development.

I would also like to acknowledge my family and friends for their support and encouragement throughout the project. Their belief in my capabilities provided me with the motivation to complete this project with dedication.

Finally, I extend my gratitude to OpenWeatherMap for providing a reliable and user-friendly API, which allowed me to work with real-time data, thus making this project a rewarding learning experience.

Thank you all for your invaluable support.

Introduction

In recent years, access to real-time data has become increasingly valuable in various applications, especially in weather forecasting, which affects both individuals and organizations worldwide. This project aims to develop a Python-based program that fetches and displays real-time weather data for any city chosen by the user. By utilizing an online weather API, this project demonstrates how Python can be used to interact with web services and process live data.

This project will provide valuable insights into coding with APIs, handling JSON data, and formatting outputs in Python. Additionally, it introduces students to the basics of HTTP requests and gives them an understanding of how weather data is managed and shared by meteorological agencies.

Objective

The objective of this project is to create an interactive and user-friendly command-line tool using Python that enables users to retrieve and display real-time weather data. Specifically, the project aims to:

Connect to an online API to fetch live weather information.
Present the fetched data, including temperature, humidity, weather description, and wind speed, in a readable format.
Allow users to enter any city name and receive the latest weather information for that location.
This project demonstrates the following technical skills:

API Integration: Connecting with an external API service to obtain data.
Data Processing: Extracting, organizing, and displaying specific information from the returned data.
Python Basics: Working with libraries, handling user input, and error handling.

Required Libraries and Tools

The project requires Python 3.x and the requests library, which is used to make HTTP requests to fetch data from the API. To set up the project environment, you can follow these steps:

- Python Installation: Ensure that Python is installed. It can be downloaded from https://www.python.org/.

- Install Requests Library: Run the command below in your command-line terminal to install the requests library, which will be used to interact with the API.

```
~ pip install requests
```

API Setup and Configuration

To retrieve weather data, we use the OpenWeatherMap API, which provides extensive meteorological information. Follow these steps to set up the API:

1.) Sign Up on OpenWeatherMap: Create a free account at OpenWeatherMap.

2.) API Key Generation: After signing up, generate an API key, which will be used to authenticate requests.

3.) API URL Structure: The API URL format is as follows:

```
http://api.openweathermap.org/data/2.5/weather?q=
{city}&appid={API_KEY}&units=metric
```

- Replace {city} with the user-input city name.

- Replace {API_KEY} with the unique key obtained from OpenWeatherMap.

- units=metric specifies that the temperature is in Celsius.

Project Code Explanation

Below is the complete code for the weather display program, followed by an explanation of each part.

```python
import requests

def get_weather(city):
    api_key = "YOUR_API_KEY"  # replace with your actual API key
    url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"

    try:
        response = requests.get(url)
        data = response.json()
        if data["cod"] != "404":
            main = data["main"]
            weather_desc = data["weather"][0]["description"]
            temp = main["temp"]
            humidity = main["humidity"]
            wind_speed = data["wind"]["speed"]

            return (f"City: {city}\n"
                    f"Temperature: {temp}°C\n"
                    f"Weather: {weather_desc}\n"
                    f"Humidity: {humidity}%\n"
                    f"Wind Speed: {wind_speed} m/s")
        else:
            return "City not found."
    except Exception as e:
        return "Error retrieving weather data."

if __name__ == "__main__":
    city = input("Enter city name: ")
    weather_info = get_weather(city)
    print(weather_info)

```

Explanation

1.) Function get_weather(city): This function takes a city name as input and uses the OpenWeatherMap API to retrieve weather data.

2.) URL Setup: The URL is constructed with the city and API key.

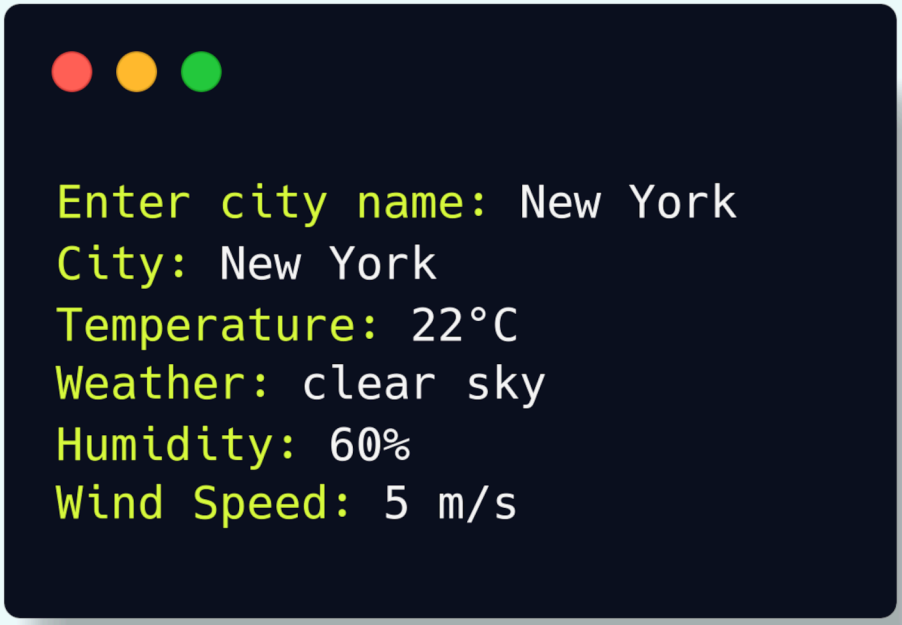
3.) Error Handling: If the city is not found, an error message is displayed.

4.) Data Processing: The JSON data returned by the API contains several fields. The function extracts the temperature, humidity, weather description, and wind speed and formats them.

5.) User Interaction: The user is prompted to enter a city name, which is passed to the get_weather() function.

Output Example

When executed, the program should output the weather information in a readable format. Below is an example output:



This format makes the data easy to read and understand, providing a quick snapshot of the weather in the selected city.

Limitations

This project, while functional, has some limitations:

1.) API Key Requirement: Users need to have their own API key, which may limit accessibility.

2.) Internet Dependency: The program requires an active internet connection to retrieve weather data.

3.) Error Handling: The error handling is basic and could be enhanced to provide more detailed error messages.

4.) Limited Customization: Currently, the program only provides data in metric units (Celsius). Additional features could allow users to choose units (e.g., Fahrenheit).Q

Conclusion

This project provides a hands-on introduction to API integration, HTTP requests, and data processing in Python. By creating a real-time weather application, students gain valuable experience working with JSON data and formatting output for user interaction. The project also demonstrates the practical use of Python in accessing and processing live data, which has numerous applications beyond weather forecasting.

Future improvements could include additional features such as more extensive error handling, unit selection (Celsius/ Fahrenheit), and data visualizations.

References

1.) Python Documentation

2.) OpenWeatherMap API Documentation

3.) Requests Library: https://docs.python-requests.org/

4.) Sumita Arora (Class 12 Book)