Weather Checking Application Documentation

-Gyaneshwar Rao

Table of Contents

1. Introduction

The Weather Checking Application is a Python program that allows users to retrieve and display real-time weather information for various cities. The application is designed to be user-friendly and offers the following features:

- Display weather information for a list of favorite cities.
- Check weather for other cities by specifying the city name.
- Add or remove cities from the list of favorite cities.
- Continuously refresh and display weather information for favorite cities.

This documentation provides a detailed explanation of how to use the application, the code structure, and an overview of its functionality.

2. Prerequisites

Before using the Weather Checking Application, ensure you have the following prerequisites:

- Python 3.x installed on your system.
- A valid API key from the WeatherAPI (https://weatherapi.com/). You need to replace `'6aef3dff462d4f47ac0124818230111'` with your actual API key in the code.

3. Getting Started

To get started with the Weather Checking Application, follow these steps:

- Ensure you have Python 3.x installed on your system.

- Obtain a valid API key from WeatherAPI (https://weatherapi.com/) and replace the placeholder `'6aef3dff462d4f47ac0124818230111'` with your actual API key in the code.
- Save the Python code to a file, for example, `weather_app.py`.
- Open a terminal or command prompt and navigate to the directory where the code file is located.
- Run the application by executing the following command:

```
python weather_app.py
```

The application will start, and you will be presented with a menu to interact with the features.

4. Usage

The Weather Checking Application provides multiple options for users to interact with the weather information. Here are the main functionalities:

Display Weather of Favorite Cities

Choose option 1 to display the weather information for a list of favorite cities. The application will continuously refresh and display weather details for each city. You can choose to stop the refreshing process at any time.

Check Weather for Other Cities

Choose option 2 to check the weather for a city that is not in your list of favorite cities. You will be prompted to enter the city name, and the application will display the current weather information for that city.

Add/Remove Cities to/from Favorite List

Option 3 allows you to add or remove cities from your list of favorite cities. You can choose to add a city to the list by specifying its name or remove a city from the list by entering its name.

Quit the Application

Option 4 allows you to exit the application.

5. Code Explanation

The Weather Checking Application code is organized into several components and functions. Here's an overview of each part of the code:

Importing Required Libraries

The code begins by importing the necessary libraries:

- `requests`: Used for making HTTP requests to the WeatherAPI.
- `time`: Used for adding delays during the refreshing process.

API Key and Favorite Cities

You need to replace the placeholder `'6aef3dff462d4f47ac0124818230111'` with your actual WeatherAPI key. Additionally, the `favourite_cities` list contains the names of your favorite cities.

`get_weather` Function

This function is responsible for making an API request to the WeatherAPI to retrieve weather information for a given city. It handles errors, parses the JSON response, and formats the weather information into a human-readable string.

`display_favourite_cities` Function

This function continuously refreshes and displays weather information for the cities in the `favourite_cities` list. It allows the user to stop the refreshing process when desired.

`check_weather_for_city` Function

This function allows the user to check the weather for a city not in the favorite cities list. It prompts the user to enter the city name and displays the weather information.

Main Execution Block

The main execution block of the code presents a menu to the user, allowing them to choose from various options, including displaying favorite city weather, checking weather for other cities, adding or removing cities from the favorite list, and quitting the application.

6. Conclusion

The Weather Checking Application is a Python program that offers a convenient way to access and display real-time weather information for your favorite cities and other locations. Whether you're a developer looking to understand the code or an end user interested in checking the weather, this documentation provides a clear and detailed explanation of the application's features and functionalities.

Feel free to explore, customize, and use the application to stay informed about the weather conditions in your desired locations.

```
[ ]  !pip install requests
     import requests
     import time
     API_Key='6aef3dff462d4f47ac0124818230111'
     favourite_cities=['hyderabad','vizag','delhi','chennai','mumbai']
     def get_weather(city):
         try:
           url = f'http://api.weatherapi.com/v1/current.json?key={API_KEY}&q={city}'
           response = requests.get(url)
           response.raise_for_status()  # Check for HTTP request errors

           data = response.json()
           if 'error' not in data:
               location = data['location']['name']
               current_time = data['location']['localtime']
               is_day = data['current']['is_day']
               temperature = data['current']['temp_c']
               weather_description = data['current']['condition']['text']
               wind_speed = data['current']['wind_kph']

               time_of_day = 'Day' if is_day == 1 else 'Night'
               windy_condition = 'Breezy' if wind_speed >= 25 else 'Windy' if wind_speed >= 15 else 'Calm'
               weather_info = f'City: {location}, Time: {current_time}, {time_of_day}, ' \
                              f'Temperature: {temperature}°C, Weather: {weather_description}, ' \
                              f'Wind: {windy_condition}, Wind Speed: {wind_speed} km/h'
               return weather_info
           else:
               return f'Error: Unable to fetch weather data for {city}'
         except requests.exceptions.RequestException as e:
             return f'Error: Request error - {str(e)}'
         except ValueError as e:
```

```
            return f'Error: JSON parsing error - {str(e)}'
        except Exception as e:
            return f'Error: {str(e)}'
    def display_favourite_cities():
      while True:
        for city in favourite_cities:
          info=get_weather(city)
          print(info)
        c=input('Do you wish to continue refreshing(y/n)').strip().lower()
        if c!='y':
          break
        time.sleep(5)
    def check_weather_for_city(city):
        weather_info = get_weather(city)
        print(weather_info)
    if __name__ =='__main__':
      print("Welcome to weather checking application")
      while True:
        print("select option:")
        print('1.Display weather of favourite cities')
        print('2.Check weather for other cities')
        print('3.Add/Remove cities to/from favourite list')
        print('4.quit')
        choice=int(input('enter your choice'))
        if choice==1:
          display_favourite_cities()
        elif choice==2:
          city=input("Enter city name:")
          check_weather_for_city(city)
        elif choice==3:
          print("if you'd like to add a city to the list input-a\nto delete a city input-d")
```

```
          ad=input().strip().lower()
          if ad=='a':
            city=input("enter city to add into the list: ")
            favourite_cities.append(city)
            print(favourite_cities)
          elif ad=='d':
            city=input("enter city to delete from the list")
            favourite_cities.remove(city)
            print(favourite_cities)
        elif choice==4:
          break
```