Global Real-Time Weather Data Collection and Analysis using Web Scraping

In [ ]:

Introduction

Weather patterns play a crucial role in shaping human activities, from agriculture and transportation to health and disaster management. With advancements in technology, it is now possible to monitor and analyze real-time weather data from multiple locations across the globe. This project implements a Python-based approach to fetch, process, and store real-time weather information from different cities using the WeatherAPI service. The collected data is structured and saved for further analysis and decision-making.

In [ ]:

Aims of the Project

- To automate the process of fetching real-time weather data from multiple global cities.
- To extract key weather parameters such as temperature, humidity, wind speed, and conditions.
- To store the collected data in a structured format (CSV) for easy access, analysis, and visualization.
- To provide a scalable solution that can be extended to include more cities or integrate with dashboards for live monitoring.

In [ ]:

In [ ]:
```
# Weather Data Fetcher - Jupyter Notebook Implementation
# Step-by-step weather data collection and CSV export
```

Import Libraries and Setup

In [1]:
```
import requests
import csv
import pandas as pd
from datetime import datetime
```

In [ ]:

In [2]:
```
# Your WeatherAPI key
API_KEY = "5795f5cc64c04ea3af3211036250509"
BASE_URL = "http://api.weatherapi.com/v1/current.json"

print("✓ Libraries imported successfully")
print(f"✓ API Key configured: {API_KEY[:10]}...")
```

```
✓ Libraries imported successfully
✓ API Key configured: 5795f5cc64...
```

In [ ]:

Define Cities List

In [3]:
```
# List of cities to fetch weather data for
cities = [
    "London", "New York", "Tokyo", "Sydney", "Paris",
    "Mumbai", "Lagos", "Cairo", "Moscow", "Toronto"
]

print(f"✓ {len(cities)} cities configured:")
print(f"  {', '.join(cities)}")
```

```
✓ 10 cities configured:
  London, New York, Tokyo, Sydney, Paris, Mumbai, Lagos, Cairo, Moscow, Toronto
```

In [ ]:

Create Weather Fetching Function

In [4]:
```python
def get_weather_data(city):
    """Fetch weather data for a single city - returns dict or None"""
    try:
        # Make API request
        response = requests.get(BASE_URL, params={'key': API_KEY, 'q': city}, timeout=10)
        response.raise_for_status()
        data = response.json()

        # Extract key information in one compact dictionary
        return {
            'city': data['location']['name'],
            'country': data['location']['country'],
            'temperature_c': data['current']['temp_c'],
            'temperature_f': data['current']['temp_f'],
            'condition': data['current']['condition']['text'],
            'humidity': data['current']['humidity'],
            'wind_kph': data['current']['wind_kph'],
            'pressure_mb': data['current']['pressure_mb'],
            'feels_like_c': data['current']['feelslike_c'],
            'fetch_time': datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        }
    except Exception as e:
        print(f"✗ Error fetching {city}: {str(e)[:50]}...")
        return None

print("✓ Weather fetching function created")
```

```
✓ Weather fetching function created
```

In [ ]:

Fetch Weather Data for All Cities

In [5]:
```python
print("Fetching weather data...")
weather_data = []

# Fetch data for each city
for i, city in enumerate(cities, 1):
    print(f"[{i}/{len(cities)}] {city}...", end=" ")
    data = get_weather_data(city)
    if data:
        weather_data.append(data)
        print("✓")
    else:
        print("✗")

print(f"\n✓ Successfully collected data for {len(weather_data)} cities")
```

```
Fetching weather data...
[1/10] London... ✓
[2/10] New York... ✓
[3/10] Tokyo... ✓
[4/10] Sydney... ✓
[5/10] Paris... ✓
[6/10] Mumbai... ✓
[7/10] Lagos... ✓
[8/10] Cairo... ✓
[9/10] Moscow... ✓
[10/10] Toronto... ✓

✓ Successfully collected data for 10 cities
```

In [ ]:

Display Sample Data

In [6]:
```python
# Show first few results as preview
if weather_data:
    print("\n Sample Weather Data:")
    print("-" * 60)
    for data in weather_data[:3]:  # Show first 3 cities
        print(f"{data['city']}, {data['country']}: {data['temperature_c']}°C, {data['condition']}")

    if len(weather_data) > 3:
        print(f"... and {len(weather_data)-3} more cities")
```

```
 Sample Weather Data:
------------------------------------------------------------
London, United Kingdom: 16.3°C, Clear
New York, United States of America: 26.1°C, Sunny
Tokyo, Japan: 26.2°C, Sunny
... and 7 more cities
```

In [ ]:

Create and Save CSV File

In [7]:
```python
# Generate filename with timestamp
filename = f"weather_data_{datetime.now().strftime('%Y%m%d_%H%M%S')}.csv"

# Write to CSV using built-in csv module.
with open(filename, 'w', newline='', encoding='utf-8') as file:
    if weather_data:  # Only if we have data
        writer = csv.DictWriter(file, fieldnames=weather_data[0].keys())
        writer.writeheader()
        writer.writerows(weather_data)

print(f"✓ Data saved to: {filename}")
print(f"✓ Total cities: {len(weather_data)}")
print(f"✓ Data fields: {len(weather_data[0].keys()) if weather_data else 0}")
```

```
✓ Data saved to: weather_data_20250905_223837.csv
✓ Total cities: 10
✓ Data fields: 10
```

In [ ]:

Display CSV Content

In [8]:
```python
# Read and display the CSV content using pandas for nice formatting
try:
    df = pd.read_csv(filename)
    print(f"\n CSV File Contents ({len(df)} rows):")
    print("=" * 80)
    print(df.to_string(index=False))
except Exception as e:
    print(f"Could not display CSV: {e}")
```

```
 CSV File Contents (10 rows):
================================================================================
    city                  country  temperature_c  temperature_f     condition  humidity  wind_kph  pressure_mb
feels_like_c          fetch_time
  London           United Kingdom           16.3           61.3         Clear        77       6.8       1022.0
16.3 2025-09-05 22:31:46
New York United States of America           26.1           79.0         Sunny        60      17.6       1013.0
26.9 2025-09-05 22:31:46
   Tokyo                    Japan           26.2           79.2         Sunny        79       4.0       1009.0
29.0 2025-09-05 22:31:46
  Sydney                Australia           13.0           55.4 Partly cloudy        67      12.6       1031.0
11.9 2025-09-05 22:31:47
   Paris                   France           16.1           61.0         Clear        77       3.6       1023.0
16.1 2025-09-05 22:31:47
  Mumbai                    India           27.0           80.6          Mist        89      18.7       1005.0
30.8 2025-09-05 22:31:48
   Lagos                  Nigeria           25.3           77.5 Partly Cloudy        89      18.4       1015.0
27.2 2025-09-05 22:31:48
   Cairo                    Egypt           27.3           81.1         Clear        62      16.2       1012.0
29.8 2025-09-05 22:31:49
  Moscow                   Russia           13.0           55.4         Clear        88       6.8       1023.0
12.8 2025-09-05 22:31:49
 Toronto                   Canada           20.0           68.0      Overcast        64      26.3       1005.0
20.0 2025-09-05 22:31:50
```

In [ ]:

In [10]:
```python
df.head(10)
```

Out[10]:

	city	country	temperature_c	temperature_f	condition	humidity	wind_kph	pressure_mb	feels_like_c	fetch_time
0	London	United Kingdom	16.3	61.3	Clear	77	6.8	1022.0	16.3	2025-09-05 22:31:46
1	New York	United States of America	26.1	79.0	Sunny	60	17.6	1013.0	26.9	2025-09-05 22:31:46
2	Tokyo	Japan	26.2	79.2	Sunny	79	4.0	1009.0	29.0	2025-09-05 22:31:46
3	Sydney	Australia	13.0	55.4	Partly cloudy	67	12.6	1031.0	11.9	2025-09-05 22:31:47
4	Paris	France	16.1	61.0	Clear	77	3.6	1023.0	16.1	2025-09-05 22:31:47
5	Mumbai	India	27.0	80.6	Mist	89	18.7	1005.0	30.8	2025-09-05 22:31:48
6	Lagos	Nigeria	25.3	77.5	Partly Cloudy	89	18.4	1015.0	27.2	2025-09-05 22:31:48
7	Cairo	Egypt	27.3	81.1	Clear	62	16.2	1012.0	29.8	2025-09-05 22:31:49
8	Moscow	Russia	13.0	55.4	Clear	88	6.8	1023.0	12.8	2025-09-05 22:31:49
9	Toronto	Canada	20.0	68.0	Overcast	64	26.3	1005.0	20.0	2025-09-05 22:31:50

In [ ]:

Project Summary

This project successfully demonstrates how to collect, process, and store global real-time weather data in an automated way. By covering multiple cities across continents, it enables comparative climate analysis and offers a foundation for future applications in environmental research, weather forecasting models, and smart decision-making systems.

Importance:

- Provides real-time insights into global weather conditions.
- Facilitates climate monitoring, disaster preparedness, and urban planning.
- Acts as a building block for predictive analytics in meteorology.
- Can be integrated into dashboards for continuous monitoring or combined with machine learning models for forecasting.

In [ ]:

Author : Uzoh C. Hillary

Email : uzohhillary@gmail.com

Github: https://github.com/Uzo-Hill

LinkedIn: http://www.linkedin.com/in/hillaryuzoh

In [ ]: