Retrieve weather data using OpenWeatherMap API

```
import requests
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
api_key = 'f4cff5113d01d09ca915b3b5fa0f7c0a'
city = 'Nashik'
url = f'http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric'

response = requests.get(url)
weather_data = response.json()
weather_data
```

```
{'coord': {'lon': 73.8, 'lat': 19.9833},
 'weather': [{'id': 803,
   'main': 'Clouds',
   'description': 'broken clouds',
   'icon': '04n'}],
 'base': 'stations',
 'main': {'temp': 22.38,
  'feels_like': 22.89,
  'temp_min': 22.38,
  'temp_max': 22.38,
  'pressure': 1012,
  'humidity': 85,
  'sea_level': 1012,
  'grnd_level': 939},
 'visibility': 10000,
 'wind': {'speed': 1.33, 'deg': 67, 'gust': 1.36},
 'clouds': {'all': 62},
 'dt': 1729628857,
 'sys': {'country': 'IN', 'sunrise': 1729645337, 'sunset': 1729686938},
 'timezone': 19800,
 'id': 1261731,
 'name': 'Nashik',
 'cod': 200}
```

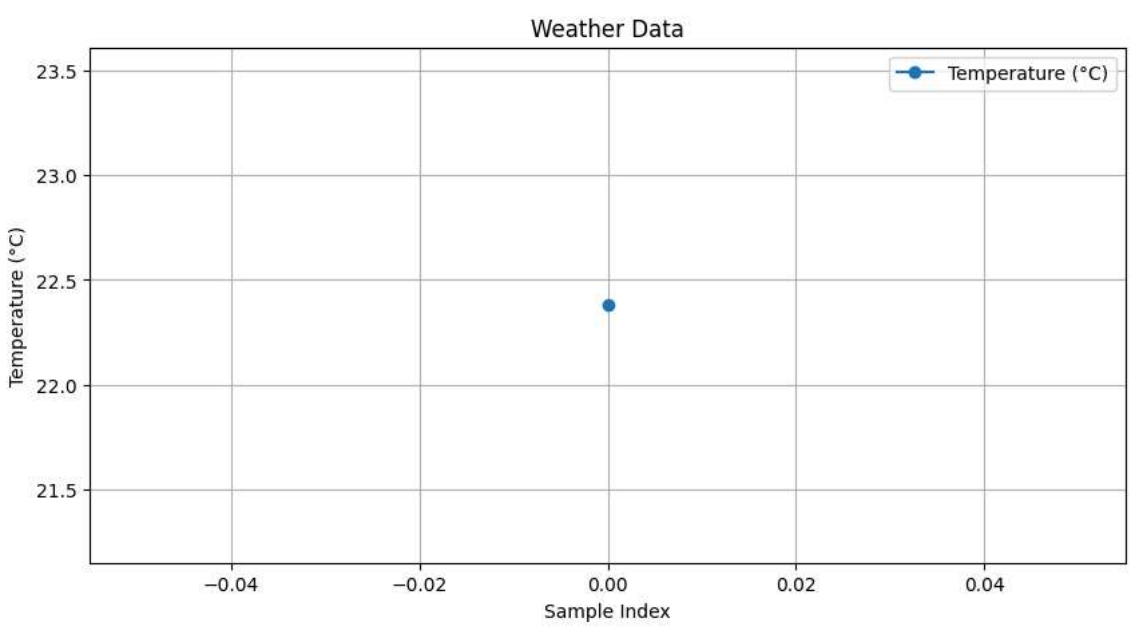
```
def extract_weather_attributes(data):
    try:
        temperature = data['main']['temp']
        humidity = data['main']['humidity']
        wind_speed = data['wind']['speed']
        precipitation = data.get('rain', {}).get('1h', 0)
        return {
            "temperature": temperature,
            "humidity": humidity,
            "wind_speed": wind_speed,
            "precipitation": precipitation
        }
    except KeyError:
        print("Error extracting weather data")
        return None

weather_attributes = extract_weather_attributes(weather_data)



import pandas as pd

def preprocess_data(data):
    df = pd.DataFrame(data)
    df.fillna(0, inplace=True)
    return df
weather_df = preprocess_data([weather_attributes])


weather_df
```

	temperature	humidity	wind_speed	precipitation
0	22.38	85	1.33	0

```
average_temp = weather_df['temperature'].mean()
max_temp = weather_df['temperature'].max()
min_temp = weather_df['temperature'].min()

print(f"Average Temperature: {average_temp}°C")
print(f"Max Temperature: {max_temp}°C")
print(f"Min Temperature: {min_temp}°C")
```

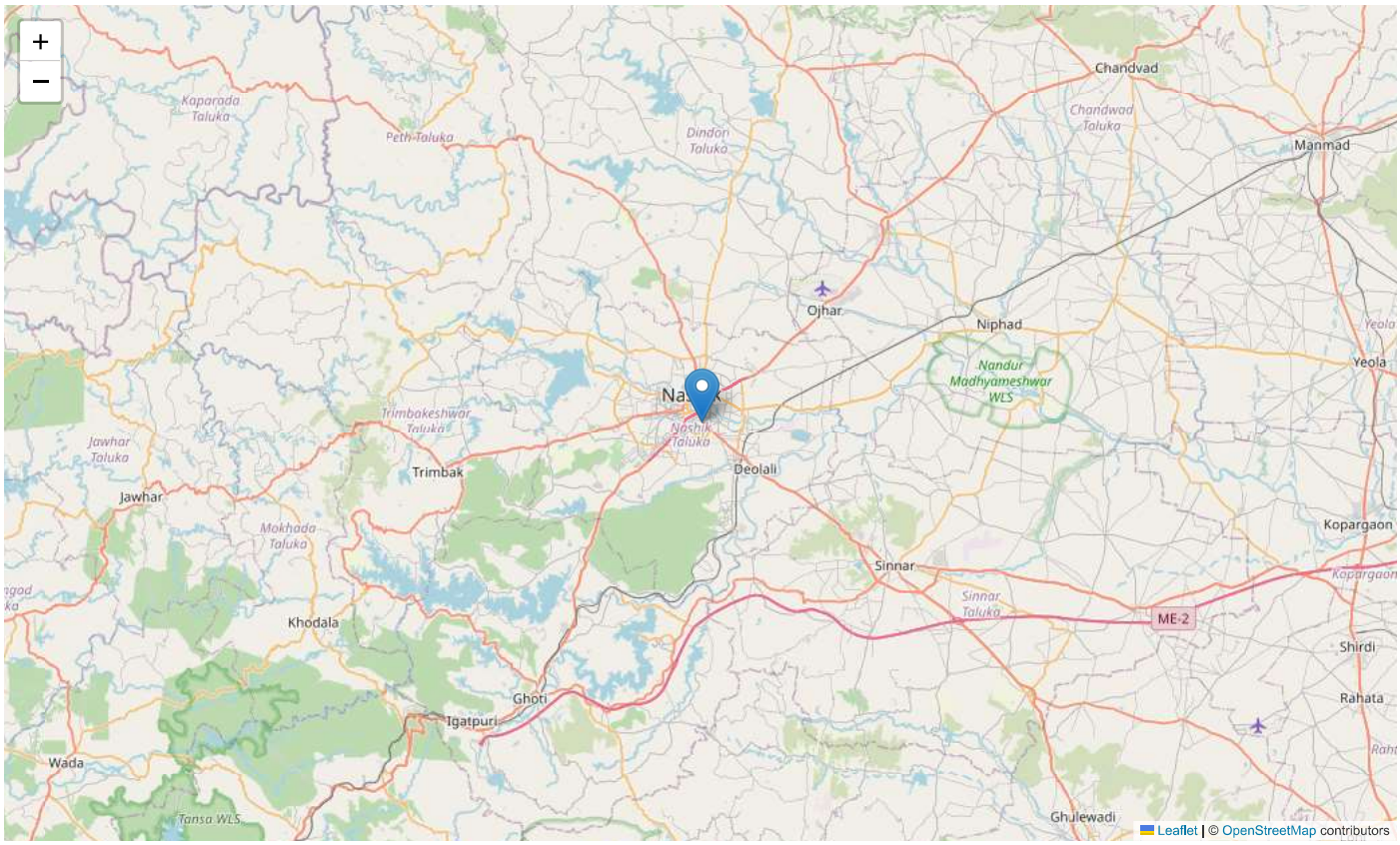
```
Average Temperature: 22.38°C
Max Temperature: 22.38°C
Min Temperature: 22.38°C
```

```
import matplotlib.pyplot as plt

def visualize_weather_data(df):
    plt.figure(figsize=(10, 5))
    plt.plot(df['temperature'], label='Temperature (°C)', marker='o')
    plt.title('Weather Data')
    plt.xlabel('Sample Index')
    plt.ylabel('Temperature (°C)')
    plt.legend()
    plt.grid()
    plt.show()

visualize_weather_data(weather_df)
```




```
import folium

def create_map(lat, lon):
    map_weather = folium.Map(location=[lat, lon], zoom_start=10)
    folium.Marker([lat, lon], popup="Weather Info").add_to(map_weather)
    return map_weather
lat = weather_data['coord']['lat']
lon = weather_data['coord']['lon']
map_weather = create_map(lat, lon)
map_weather.save("weather_map.html")
```

```
map_weather
```




```
import seaborn as sns

def plot_correlation(df):
    plt.figure(figsize=(8, 6))
    sns.heatmap(df.corr(), annot=True, fmt=".2f", cmap='coolwarm')
    plt.title('Correlation Heatmap')
    plt.show()
plot_correlation(weather_df)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/matrix.py:202: RuntimeWarning: All-NaN slice encountered
  vmin = np.nanmin(calc_data)
/usr/local/lib/python3.10/dist-packages/seaborn/matrix.py:207: RuntimeWarning: All-NaN slice encountered
  vmax = np.nanmax(calc_data)
```