**1. Import Library**

import streamlit as st

import requests

import pandas as pd

from sklearn.linear\_model import LinearRegression

import matplotlib.pyplot as plt

from matplotlib import rcParams

* **streamlit**: Library utama untuk membangun aplikasi berbasis web
* **requests**: Untuk melakukan permintaan data ke API OpenWeatherMap
* **pandas**: Digunakan untuk memproses dan mengelola data tabular (data historis)
* **LinearRegression**: Dari library sklearn, digunakan untuk membangun model prediksi suhu
* **matplotlib**: Untuk membuat visualisasi data berupa grafik
* **rcParams**: Digunakan untuk mengatur tampilan seperti font

sebelum library di import ke dalam python harus dipastikan sudah di install karena jika belum maka akan menampilkan error

**2. Fungsi untuk Mengambil Data Cuaca dari OpenWeatherMap**

def get\_weather(city, api\_key):

url = f"https://api.openweathermap.org/data/2.5/weather?q={city},ID&appid={api\_key}&units=metric"

response = requests.get(url)

return response.json() if response.status\_code == 200 else None

* Fungsi ini menerima **city** (nama kota) dan **api\_key** (kunci API)
* Membuat permintaan ke URL API OpenWeatherMap dengan format JSON
* JSON (JavaScript Object Notation) adalah format data yang digunakan untuk menyimpan dan bertukar informasi. JSON sangat umum digunakan dalam API karena formatnya sederhana dan mudah dibaca oleh manusia maupun mesin
* Jika status kode respons adalah 200 (berhasil), data dikembalikan dalam bentuk JSON; jika tidak, fungsi mengembalikan None

**3. Konfigurasi API**

api\_key = "0b177fb6f5bd23c0f6d36429a8c2ba99"

* Menyimpan **API key** yang digunakan untuk mengakses layanan OpenWeatherMap
* Jadi daftar akun dulu di OpenWeatherMap jika sudah ada API Key akan terlihat kemudian disalin dimasukkan kedalam kodingan tersebut

**4. Data Historis dan Model Prediksi**

**Data Simulasi**

data = pd.DataFrame({

'kelembapan': [60, 65, 70, 75, 80, 85, 90],

'tekanan': [1000, 1005, 1010, 1015, 1020, 1025, 1030],

'suhu\_max': [30, 31, 32, 33, 34, 35, 36]

})

* Data ini adalah simulasi yang berisi informasi historis kelembapan (%), tekanan udara (hPa), dan suhu maksimum (°C)
* Misal kelembapannya 60% maka tekanannya 1000 hPa dan suhu maksimalnya 30°C
* Digunakan untuk melatih model prediksi suhu maksimum

**Model Linear Regression**

X = data[['kelembapan', 'tekanan']]

y = data['suhu\_max']

model = LinearRegression().fit(X, y)

* Model Linear Regression memetakan hubungan sederhana antara kelembapan, tekanan, dan suhu maksimum, sehingga bisa memprediksi suhu maksimum berdasarkan data input

penjelasan ringkasnya:

1. **Input dan Output:**
   * **Input (x):** Kelembapan (*x1*) dan tekanan (*x2*)
   * **Output (y):** Suhu maksimum (*y*)
2. **Persamaan Linear Regression:** Model memprediksi suhu maksimum (*ypred*) dengan persamaan:

*ypred = a ⋅ x1 + b ⋅ x2 + c*

* + a,b,c : Koefisien (berat) dan bias yang dipelajari dari data

1. **Pelatihan Model:**
   * Model mempelajari hubungan antara input (kelembapan, tekanan) dan output (suhu maksimum) dari data historis
   * Tujuannya: Menemukan nilai a,b,ca, b, c agar prediksi mendekati data sebenarnya
2. **Prediksi:**
   * Jika kelembapan = 70, tekanan = 1015, dan *a* = 0.04, *b* = 0.03, *c* = 10:

*ypred = (0.04 ⋅ 70)+(0.03 ⋅ 1015)+10 = 43.25*

* + Prediksi suhu maksimum = **43.25°C**.

**5. Konfigurasi Tampilan Grafik**

rcParams['font.family'] = 'DejaVu Sans'

rcParams['font.size'] = 12

* Mengatur font grafik agar lebih mudah dibaca

**6. Judul Aplikasi**

st.title("Aplikasi Cuaca Indonesia dengan Prediksi")

st.write("Dapatkan informasi cuaca terkini dan prediksi suhu maksimum di berbagai kota di Indonesia.")

* **st.title**: Menampilkan judul utama aplikasi
* **st.write**: Menampilkan deskripsi singkat tentang aplikasi

**7. Input Lokasi**

city = st.text\_input("Masukkan nama kota di Indonesia:", "Jakarta")

* **st.text\_input**: Input dari pengguna untuk memasukkan nama kota (default: Jakarta)
* **Default** berarti nilai awal yang muncul pada input pengguna
* Di sini, pengguna aplikasi akan melihat "Jakarta" sebagai kota default, sehingga jika mereka tidak memasukkan kota lain, aplikasi akan menampilkan data cuaca untuk Jakarta

**8. Proses Data Cuaca**

**Memanggil API**

weather\_data = get\_weather(city, api\_key)

* Memanggil fungsi **get\_weather** untuk mengambil data cuaca kota yang dimasukkan pengguna
* Contohnya, jika seseorang membuka aplikasi, mereka akan memasukkan nama kota seperti "Surabaya"
* Kode **get\_weather(city, api\_key)** akan mengambil data cuaca berdasarkan input pengguna yaitu kota “Surabaya”

**Menampilkan Data Cuaca**

kelembapan = weather\_data['main']['humidity']

tekanan = weather\_data['main']['pressure']

suhu\_sekarang = weather\_data['main']['temp']

suhu\_min = weather\_data['main']['temp\_min']

suhu\_max = weather\_data['main']['temp\_max']

kondisi = weather\_data['weather'][0]['description'].capitalize()

* Data cuaca diambil dari respons JSON dan ditampilkan menggunakan **st.write**:
  + Suhu
  + Kelembapan
  + Tekanan udara
  + Kondisi cuaca (misalnya, hujan atau cerah)

Penjelasan sedikit :

weather\_data['main']['humidity'] berarti:

* Masuk ke bagian main dalam data cuaca, lalu ambil nilai humidity (kelembapan)

weather\_data['weather'][0]['description'] berarti :

* mengambil deskripsi cuaca dari data API, misalnya "clear sky"

**Prediksi Suhu Maksimum**

suhu\_prediksi = model.predict([[kelembapan, tekanan]])[0]

st.write(f"📈 Prediksi Suhu Maksimum (berdasarkan kelembapan & tekanan): {suhu\_prediksi:.2f}°C")

* Menggunakan model Linear Regression untuk memprediksi suhu maksimum berdasarkan kelembapan dan tekanan udara
* **Cara Kerja Linear Regression:**
* Model menghitung suhu maksimum menggunakan rumus persamaan linier:  
  **Suhu Maks = a × Kelembapan + b × Tekanan + c**  
  Di mana:
  + **a** dan **b** adalah koefisien yang dipelajari model
  + **c** adalah bias (nilai awal)
* Contoh: Jika kelembapan = 70% dan tekanan = 1010 hPa, model akan memprediksi suhu maksimum berdasarkan pola dari data historis
* Hasil prediksi ditampilkan di aplikasi, memberi pengguna informasi tambahan tentang suhu maksimum berdasarkan kondisi cuaca saat ini

**9. Visualisasi Data**

temps = {

"Suhu Maksimum API": suhu\_max,

"Suhu Minimum": suhu\_min,

"Suhu Sekarang": suhu\_sekarang,

"Prediksi Suhu Maksimum": suhu\_prediksi

}

plt.figure(figsize=(6, 4))

plt.bar(temps.keys(), temps.values(), color=['red', 'blue', 'orange', 'green'])

plt.title(f"Grafik Suhu di {city}", fontsize=15, fontweight='bold')

plt.ylabel("Suhu (°C)", fontsize=12)

plt.xlabel("Kategori", fontsize=12)

plt.xticks(rotation=0, fontsize=7)

plt.yticks(fontsize=7)

st.pyplot(plt)

* **temps**: Dictionary berisi data suhu dari API dan prediksi
* **plt.bar**: Membuat grafik batang suhu untuk visualisasi
* Grafik ditampilkan menggunakan **st.pyplot**
* Kode ini membuat diagram batang untuk membandingkan **suhu maksimum**, **minimum**, **sekarang**, dan **prediksi suhu maksimum** secara visual berdasarkan data yang ada. Diagram ini mempermudah pengguna memahami perbedaan nilai suhu secara cepat

**10. Pesan Error dan Info**

st.error("Gagal mendapatkan data cuaca. Pastikan nama kota benar dan coba lagi.")

st.info("Masukkan nama kota untuk melihat cuaca.")

* Jika data tidak ditemukan, aplikasi menampilkan pesan error
* Jika pengguna belum memasukkan kota, aplikasi menampilkan informasi mengenai kota Jakarta karena sudah default

**11. Tambahan Perintah**

cd E:\final\_project\_cuaca

python -m streamlit run app.py

**Penjelasan:**

1. **cd E:\final\_project\_cuaca:**
   * **Tujuan:**
     + Perintah ini digunakan untuk berpindah ke direktori tempat file aplikasi Streamlit (app.py) berada
   * **Alasan:**
     + Agar terminal bekerja di folder yang benar, sehingga perintah selanjutnya bisa menemukan file app.py
2. **python -m streamlit run app.py:**
   * **Tujuan:**
     + Menjalankan aplikasi Streamlit dengan file Python bernama app.py
   * **Rincian:**
     + **python -m:** Menjalankan modul Python (streamlit dalam hal ini)
     + **streamlit run app.py:** Menginstruksikan Streamlit untuk menjalankan aplikasi berdasarkan kode yang ada di app.py

**Mengapa Perlu Ada?**

* **Lokasi File:**
  + Terminal harus berada di lokasi file app.py agar Streamlit dapat menemukan dan menjalankannya
* **Streamlit:**
  + Streamlit bukan IDE seperti Jupyter Notebook yang otomatis menjalankan kode; perlu perintah eksplisit untuk menjalankan server aplikasi
* **Menjalankan Aplikasi:**
  + Perintah ini memulai server lokal Streamlit, sehingga aplikasi dapat dibuka di browser untuk dilihat dan digunakan

tambahan perintah tersebut memastikan bahawa kita berada di direktori yang benar tempat file aplikasi berada dan Aplikasi Streamlit berjalan melalui terminal, sehingga dapat membuka dan melihatnya di browser local.

**Kesimpulan:**

* Kode ini membuat aplikasi cuaca berbasis Streamlit dengan prediksi suhu maksimum.
* Data cuaca real-time diambil dari OpenWeatherMap, sementara prediksi dihitung menggunakan model Linear Regression.
* Aplikasi ini menampilkan data cuaca dan visualisasi grafik yang mudah dipahami.