To build a "Real-Time Weather App" Python project using OpenWeatherMap API, requests, and Streamlit following the mini guide, here is a structured approach and example outline:

**Project Objective:**

Display current weather and 5-day forecast for any city, including dynamic weather icons and temperature unit toggle.

**Tools:**

* **OpenWeatherMap API**: to fetch current weather and forecast.
* **requests**: for HTTP API calls.
* **Streamlit**: for interactive UI and charts.

**Mini Guide Steps:**

1. **Create input for city names:**
   * Use st.text\_input() for city name entry.
2. **Fetch data from OpenWeatherMap:**
   * Use OpenWeatherMap current weather API (weather) and forecast API (forecast).
   * Need to sign up for API key.
3. **Show temp, humidity, sunrise/sunset:**
   * Parse JSON response, display data in Streamlit with user-friendly format.
   * Convert sunrise/sunset from UNIX timestamp.
4. **Add 5-day forecast chart:**
   * Use forecast endpoint (3-hour intervals),
   * Aggregate or plot forecast temps over next 5 days,
   * Use Streamlit line chart or Plotly.
5. **Add dynamic icons (rainy, cloudy):**
   * OpenWeatherMap provides icon codes,
   * Dynamically show weather icons in UI.
6. **Allow unit toggle (Celsius/Fahrenheit):**
   * Toggle button to switch units,
   * Convert temperature as needed,
   * Pass unit param to API (units=metric or units=imperial).

**Example Streamlit Code Skeleton:**

python

**import** streamlit **as** st

**import** requests

**from** datetime **import** datetime

**import** pandas **as** pd

API\_KEY = "your\_openweathermap\_api\_key"

**def** kelvin\_to\_celsius(kelvin):

**return** kelvin - 273.15

**def** kelvin\_to\_fahrenheit(kelvin):

**return** 9/5 \* (kelvin - 273.15) + 32

**def** get\_weather(city, units='metric'):

url\_current = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={API\_KEY}&units={units}"

url\_forecast = f"http://api.openweathermap.org/data/2.5/forecast?q={city}&appid={API\_KEY}&units={units}"

current = requests.get(url\_current).json()

forecast = requests.get(url\_forecast).json()

**return** current, forecast

**def** format\_unix\_time(unix\_time):

**return** datetime.fromtimestamp(unix\_time).strftime('%H:%M:%S')

st.title("Real-Time Weather App")

city = st.text\_input("Enter city name:", "London")

unit\_choice = st.radio("Select temperature unit:", ("Celsius", "Fahrenheit"))

units = 'metric' **if** unit\_choice == "Celsius" **else** 'imperial'

**if** city:

current, forecast = get\_weather(city, units)

**if** current.get("cod") != 200:

st.error("City not found or API error.")

**else**:

st.subheader(f"Current Weather in {city}")

st.write(f"Temperature: {current['main']['temp']} °{unit\_choice[0]}")

st.write(f"Humidity: {current['main']['humidity']}%")

st.write(f"Sunrise: {format\_unix\_time(current['sys']['sunrise'])}")

st.write(f"Sunset: {format\_unix\_time(current['sys']['sunset'])}")

icon\_code = current['weather'][0]['icon']

icon\_url = f"http://openweathermap.org/img/wn/{icon\_code}@2x.png"

st.image(icon\_url)

st.subheader("5-Day Forecast")

*# Process forecast data*

df\_forecast = pd.DataFrame(forecast['list'])

df\_forecast['dt\_txt'] = pd.to\_datetime(df\_forecast['dt\_txt'])

df\_forecast = df\_forecast.set\_index('dt\_txt')

temps = df\_forecast['main'].apply(**lambda** x: x['temp'])

st.line\_chart(temps)

**Deliverables:**

* Streamlit real-time weather app with city input,
* Displays current weather with humidity, sunrise/sunset times, dynamic icons,
* 5-day temperature forecast line chart,
* Temperature unit toggle between Celsius/Fahrenheit,
* Sample queries to test cities and screenshots of outputs.