**WEATHER APP – PROJECT REPORT**

**Project Title:**

**Weather Forecast Web App using OpenWeatherMap API and Streamlit**

**Objective:**

To create a simple web-based weather application using **Streamlit** that fetches real-time weather data using the **OpenWeatherMap API** and displays it in a clean user interface.

**Technologies Used:**

* **Python**
* **Streamlit** – for web app UI
* **requests** – to access weather API
* **OpenWeatherMap API** – to get live weather data

**Key Features:**

* User can input any city name.
* Real-time display of:
  + Temperature (°C)
  + Weather condition (e.g., clear, cloudy)
  + Humidity
  + Wind speed
* Simple and responsive UI with Streamlit

**Implementation Summary:**

* The user enters a city name.
* The app sends a GET request using requests to OpenWeatherMap API.
* API returns JSON data with weather details.
* Streamlit displays the data in a user-friendly format.

**Example Code Snippet:**

import requests

import streamlit as st

api\_key = "YOUR\_API\_KEY"

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

if city:

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

response = requests.get(url)

data = response.json()

if data.get("main"):

st.write(f"Temperature: {data['main']['temp']} °C")

st.write(f"Weather: {data['weather'][0]['description']}")

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

st.write(f"Wind Speed: {data['wind']['speed']} m/s")

else:

st.error("City not found.")

**Challenges Faced:**

* Handling invalid inputs or API errors
* Display formatting and layout in Streamlit
* Managing API key securely

**Conclusion:**

This project demonstrates how to build a lightweight web application using Streamlit that fetches and displays real-time weather data using a public API, offering practical experience in API handling and web app development in Python.