# Name: Hashim Nadeem

**RollNo: SU92-BDSFM-F25-001**

# Flask Weather App Documentation

## 1. Introduction

The Flask Weather App is a simple web application built using the Flask framework. It allows users to check the current weather of any city by fetching real-time data from the OpenWeatherMap API. The app demonstrates integration of APIs, environment variables, and basic HTML/CSS templating with Flask.

## 2. Features

• Fetches live weather data for any city using OpenWeatherMap API.

• Displays temperature, weather description, humidity, and wind speed.

• Allows users to choose between Celsius and Fahrenheit units.

• Simple and clean user interface built with HTML and CSS.

## 3. Technologies Used

• Python 3  
• Flask  
• Requests  
• OpenWeatherMap API  
• HTML, CSS  
• Python-dotenv

## 4. Project Structure

weather-app/  
│  
├── app.py → Main Flask application file  
├── requirements.txt → List of required dependencies  
├── .env → Environment variables (stores API key)  
├── templates/ → HTML templates  
│ ├── base.html  
│ └── index.html  
└── static/  
 └── styles.css → CSS styling

## 5. Installation and Setup

Follow these steps to run the app locally:

1. Clone or extract the project folder.  
2. Open a terminal and navigate to the project directory.  
3. Create and activate a virtual environment:  
 • Windows:  
 python -m venv .venv  
 .\.venv\Scripts\activate  
 • macOS/Linux:  
 python3 -m venv .venv  
 source .venv/bin/activate  
4. Install dependencies:  
 pip install -r requirements.txt  
5. Add your OpenWeatherMap API key to the .env file:  
 OPENWEATHER\_API\_KEY=your\_api\_key\_here  
6. Run the Flask app:  
 python app.py  
7. Open your browser and visit:  
 http://127.0.0.1:5000/

## 6. How It Works

• The user enters a city name on the homepage and submits the form.  
• Flask receives the city name and sends a GET request to the OpenWeatherMap API using the 'requests' library.  
• The API returns current weather data in JSON format.  
• Flask processes this data and renders it in the 'index.html' template.  
• The user sees the city’s weather, temperature, humidity, and wind speed displayed on the page.

## 7. Screenshots (Optional)

You can add screenshots of your running web app here for better presentation.

## 8. Future Improvements

• Add 5-day weather forecast.  
• Include city autocomplete suggestions.  
• Improve UI with Bootstrap or Tailwind CSS.  
• Add user location detection using geolocation API.

## 9. Conclusion

This Flask Weather App is a beginner-friendly project to learn web development using Flask, API handling, and frontend integration. It can be extended with more advanced features like weather forecasting, dynamic backgrounds, and charts.