



The Superior University, Lahore

Assignment-I (Fall 2023)

Course Title:	Programming for AI				Course Code:	CAI601410	Credit Hours:	4
Instructor:	Prof. Rasikh Ali				Programme Name:	BSDS		
Semester:	4 th	Batch:	F23	Section:	BSDSM-4A	Date:	3 rd March, 2025	
Time Allowed:					Maximum Marks:			
Student's Name:	Asaad Iqbal				Reg. No.	SU92-BSDSM-F23-020		
Lab-Task 7 & 8 1: Weather App								

Task 1

Weather App using API on Flask

Task Overview

WeatherSphere is a web-based weather forecast application built using **Flask** and the **OpenWeather API**. This application allows users to check the current weather conditions and a five-day forecast by entering a city name. It supports temperature display in **Celsius** and **Fahrenheit**.

Features

- Fetch real-time weather data from the OpenWeather API.
- Display current weather conditions, including temperature, humidity, wind speed, and weather description.
- Show a 5-day forecast with daily temperature and weather conditions.
- Toggle between Celsius and Fahrenheit.
- Interactive UI with weather-based animations.

Technology Stack

- **Backend:** Flask (Python)
- **Frontend:** HTML, CSS, JavaScript
- **API:** OpenWeather API
- **Libraries:** Requests (for API calls), JSON

How It Works

1. The user enters a city name in the input field.
2. The application sends a request to the **OpenWeather API**.
3. The API responds with weather data, including temperature, humidity, and weather conditions.
4. The application processes this data and displays it on the UI.
5. The user can toggle between **Celsius** and **Fahrenheit**.

Installation & Setup

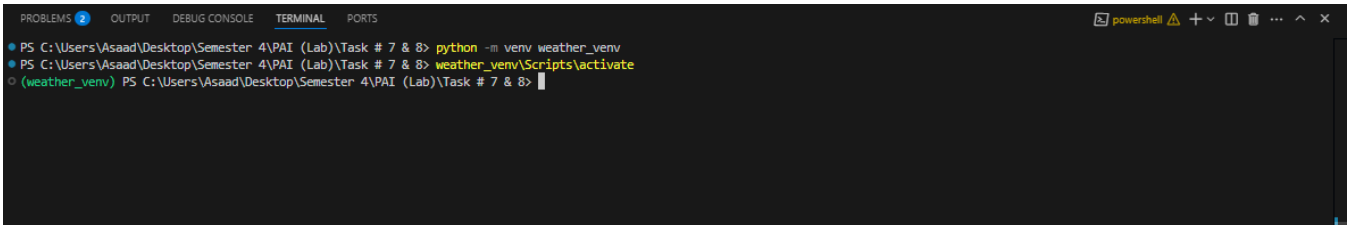
Setting Up the Virtual Environment

To keep dependencies isolated, create a virtual environment before installing the required packages.

1. Open a terminal or command prompt.
2. Navigate to the project directory.
3. Run the following command to create a virtual environment:

```
PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> python -m venv weather_venv
PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> 
```

4. Activating the environment:



```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> python -m venv weather_venv
PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> weather_venv\Scripts\activate
(weather_venv) PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> 
```

Install Dependencies

```
(weather_venv) PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> pip install -r requirements.txt
Collecting Flask==3.0.0 (from -r requirements.txt (line 1))
  Obtaining dependency information for Flask==3.0.0 from https://files.pythonhosted.org/packages/36/42/015c23096649b908c809c69388a805a571a3bea44362fe87e33cf3afa01f/flask-3.0.0-py3-none-any.whl
.metadata
  Downloading flask-3.0.0-py3-none-any.whl.metadata (3.6 kB)
Collecting requests==2.31.0 (from -r requirements.txt (line 2))
  Obtaining dependency information for requests==2.31.0 from https://files.pythonhosted.org/packages/70/8e/0e2d847013cb52cd35b38c009bb167a1a26b2ce6cd6965bf26b47bc0bf44/requests-2.31.0-py3-none-any.whl.metadata
  Using cached requests-2.31.0-py3-none-any.whl.metadata (4.6 kB)
Collecting python-dotenv==1.0.0 (from -r requirements.txt (line 3))
  Obtaining dependency information for python-dotenv==1.0.0 from https://files.pythonhosted.org/packages/44/2f/62ea1c8b593f4e093cc1a7768f0d46112107e790c3e478532329e434f00b/python_dotenv-1.0.0-py3-none-any.whl.metadata
  Downloading python_dotenv-1.0.0-py3-none-any.whl.metadata (21 kB)
Collecting Werkzeug==3.0.0 (from Flask==3.0.0->-r requirements.txt (line 1))
  Obtaining dependency information for Werkzeug==3.0.0 from https://files.pythonhosted.org/packages/52/24/ab44c871b0f07f491e5d2ad12c9bd7358e527510618cb1b803a88e986db1/werkzeug-3.1.3-py3-none-any.whl.metadata
  Using cached werkzeug-3.1.3-py3-none-any.whl.metadata (3.7 kB)
Collecting Jinja2>=3.1.2 (from Flask==3.0.0->-r requirements.txt (line 1))
  Obtaining dependency information for Jinja2>=3.1.2 from https://files.pythonhosted.org/packages/62/a1/3d68cbfd5f4b8f15abc1d571870c5fc3e594bb582bc3b64ea099db13e56/jinja2-3.1.6-py3-none-any.whl.metadata
  Downloading jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting itsdangerous>=2.1.2 (from Flask==3.0.0->-r requirements.txt (line 1))
  Obtaining dependency information for itsdangerous>=2.1.2 from https://files.pythonhosted.org/packages/04/96/92447566d16df59b2a776c0fb82dbc409e07cd95062562af01e408583fc4/itsdangerous-2.2.0-py3-none-any.whl.metadata
  Using cached itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from Flask==3.0.0->-r requirements.txt (line 1))
  Obtaining dependency information for click>=8.1.3 from https://files.pythonhosted.org/packages/7e/d4/7ebdb03970677812aac39c869717059dbb71a4cfc033ca6e5221787892c/click-8.1.8-py3-none-any.whl.metadata
  Downloading click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
Collecting blinker>=1.6.2 (from Flask==3.0.0->-r requirements.txt (line 1))
  Obtaining dependency information for blinker>=1.6.2 from https://files.pythonhosted.org/packages/10/cb/f2ad4230dc2eb1a74edf38f1a38b9b52277f75bef262d8908e60d957e13c/blinker-1.9.0-py3-none-any.whl.metadata
  Using cached blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)

Using cached colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Collecting MarkupSafe>=2.0 (from Jinja2>=3.1.2->Flask==3.0.0->-r requirements.txt (line 1))
  Obtaining dependency information for MarkupSafe>=2.0 from https://files.pythonhosted.org/packages/c1/80/a61f99dc3a936413c3ee4e1eacac96c0da5ed07ad56fd975f1a9da5bc630/MarkupSafe-3.0.2-cp312-cp312-win_amd64.whl.metadata
  Downloading MarkupSafe-3.0.2-cp312-cp312-win_amd64.whl.metadata (4.1 kB)
Downloading flask-3.0.0-py3-none-any.whl (99 kB)
99.7/99.7 kB 520.3 kB/s eta 0:00:00
Using cached requests-2.31.0-py3-none-any.whl (62 kB)
Downloading python_dotenv-1.0.0-py3-none-any.whl (19 kB)
Using cached blinker-1.9.0-py3-none-any.whl (8.5 kB)
Downloading certifi-2025.1.31-py3-none-any.whl (166 kB)
166.4/166.4 kB 158.5 kB/s eta 0:00:00
Downloading charset_normalizer-3.4.1-cp312-cp312-win_amd64.whl (102 kB)
102.8/102.8 kB 593.2 kB/s eta 0:00:00
Downloading click-8.1.8-py3-none-any.whl (98 kB)
98.2/98.2 kB 467.8 kB/s eta 0:00:00
Downloading idna-3.10-py3-none-any.whl (70 kB)
70.4/70.4 kB 481.6 kB/s eta 0:00:00
Using cached itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Downloading Jinja2-3.1.6-py3-none-any.whl (134 kB)
134.9/134.9 kB 569.2 kB/s eta 0:00:00
Downloading urllib3-2.3.0-py3-none-any.whl (128 kB)
128.4/128.4 kB 686.8 kB/s eta 0:00:00
Using cached werkzeug-3.1.3-py3-none-any.whl (224 kB)
Downloading MarkupSafe-3.0.2-cp312-cp312-win_amd64.whl (15 kB)
Using cached colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: urllib3, python-dotenv, MarkupSafe, itsdangerous, idna, colorama, charset-normalizer, certifi, blinker, Werkzeug, requests, Jinja2, click, Flask
Successfully installed Flask-3.0.0 Jinja2-3.1.6 MarkupSafe-3.0.2 Werkzeug-3.1.3 blinker-1.9.0 certifi-2025.1.31 charset-normalizer-3.4.1 click-8.1.8 colorama-0.4.6 idna-3.10 itsdangerous-2.2.0 python-dotenv-1.0.0 requests-2.31.0 urllib3-2.3.0

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(weather_venv) PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> |
```

Set Up API Key

Create a .env file and add:

```
.env x
.env
1 WEATHER_API_KEY=c535b05c56647a1e68b0945ac0015b83
```

Flask Code (Backend Code):

This Flask-based weather application fetches current weather and forecast data using the OpenWeather API. The app defines routes for fetching weather (/weather) and forecast (/forecast) data based on either city name or geographic coordinates (latitude and longitude). It retrieves the API key from environment variables and constructs API requests dynamically. The `get_weather_data` function handles API requests, ensuring error handling and response validation. The `process_current_weather` function extracts and formats relevant weather details like temperature, humidity, wind speed, and sunrise/sunset times, while `process_forecast` processes multi-day forecasts, aggregating temperature ranges and selecting the most frequent weather icon. The application runs on debug mode for development, and the last line prints the API key for debugging purposes. The app renders an `index.html` template for the main page but primarily serves JSON responses for weather queries.

Frontend Code:

The `index.html` file defines a weather application named "WeatherSphere" with a user-friendly interface for retrieving and displaying weather information. The page includes a search bar for entering city names, a button to fetch weather data based on the user's location, and a toggle to switch between Celsius and Fahrenheit. The layout is structured inside a `.container` with a visually appealing background that changes dynamically based on weather conditions. CSS styling enhances the design with smooth transitions, box shadows, and animations for elements like sun rays and raindrops. The JavaScript code handles fetching real-time weather and forecast data from an API, updating the UI dynamically, and managing units using local storage. The page displays details such as temperature, humidity, wind speed, visibility, sunrise, sunset, and a five-day forecast with icons. Error handling ensures user feedback for incorrect inputs, making the application interactive and visually engaging.

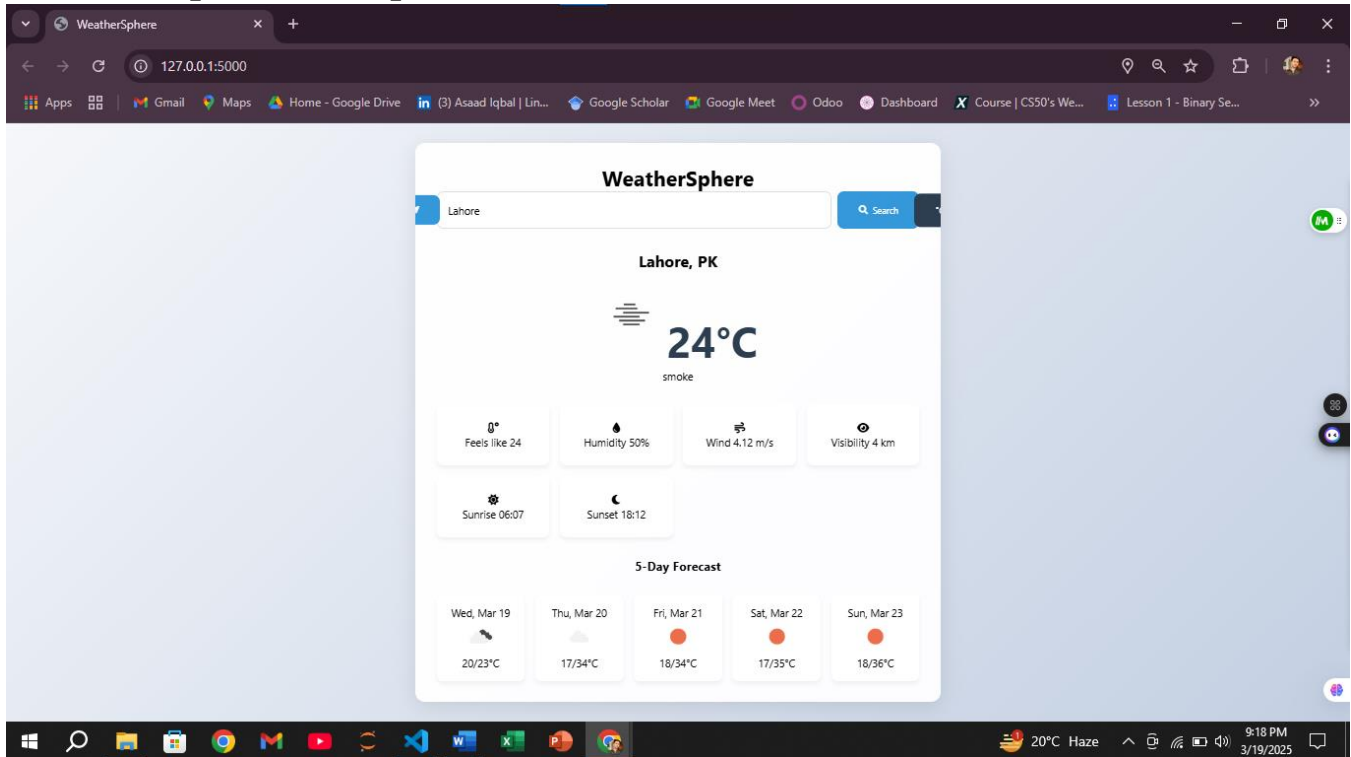
Run the Application

```
(weather_venv) PS C:\Users\Asaad\Desktop\Semester 4\PAI (Lab)\Task # 7 & 8> python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 970-957-419
```

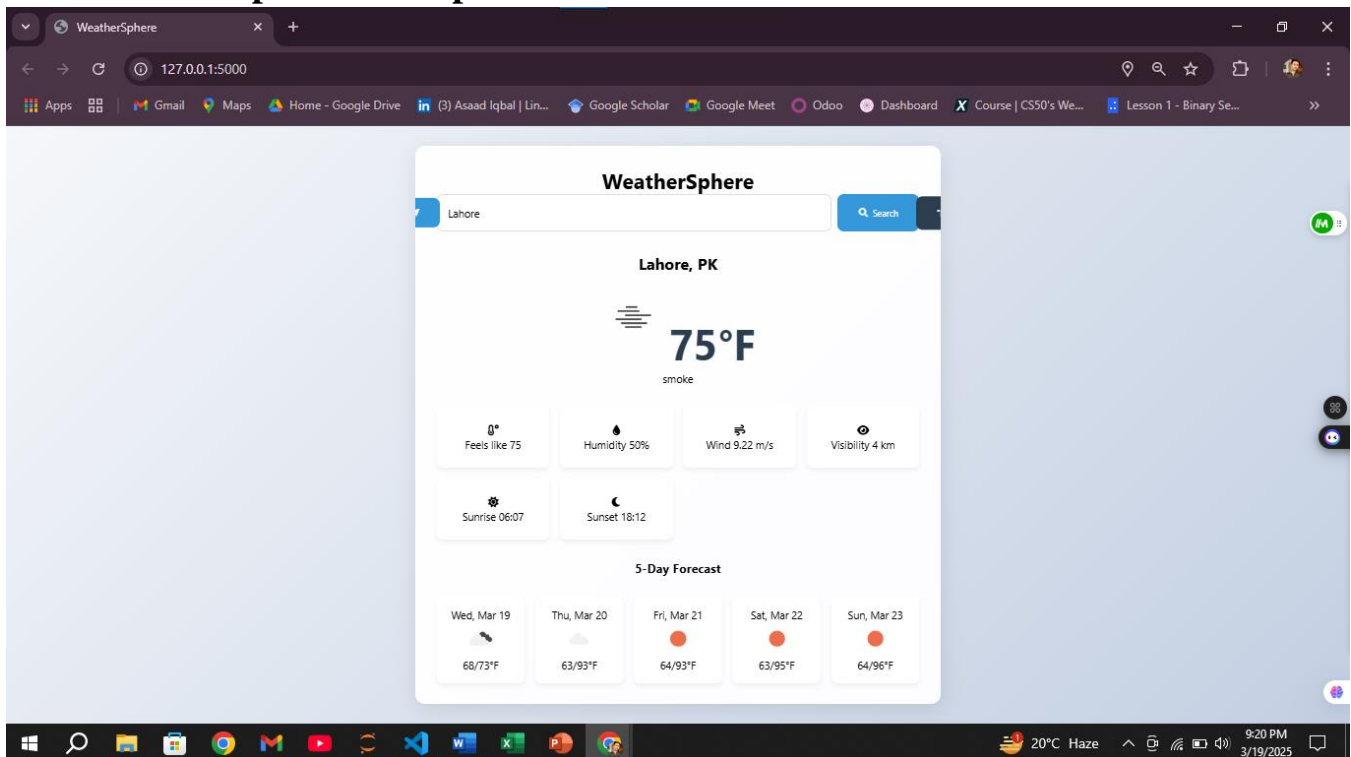
Now we will open this in browser.

Screenshots:

Celsius temperature output:



Fahrenheit temperature output:



Note:

We can change the scale from the button just beside Search.