VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

ACADEMIC YEAR 2025 – 2026

Weather Application

Technology : Pyhton , CSV file storage

> Weather Application

Course : Python Essentials

Submitted By : Gunvant Solanki

Reg No : 25BAI11341

Branch : AIML(CSE)

Faculty : DR PREETAM SUMAN

Vellore Institute of Technology , Bhopal

📘 Weather App Project Documentation

1️⃣ Introduction

This project is a weather information application built using Python. It provides live weather data by fetching information from the OpenWeatherMap API and displays it through a simple graphical user interface developed using Tkinter. The goal is to demonstrate real-world application development using GUI programming, API integration, and data processing.

2️⃣ Problem Statement

Users often need quick access to weather information to make decisions related to travel, outdoor plans, and safety. However, many apps are cluttered and not beginner-friendly.

Therefore, a simple GUI application was developed to show essential weather data instantly for any Indian state.

3️⃣ Objectives

Fetch and present real-time weather data from a trusted API.

Provide a clean and intuitive GUI for users.

Display important weather metrics including temperature, pressure, and climate.

Convert API temperature units (Kelvin → Celsius).

4️⃣ Functional Requirements

The application:

- Displays a list of Indian states through a dropdown input
- Retrieves weather data based on selected state
- Shows weather condition, description, temperature, and pressure in GUI
- Offers a one-click button interaction for execution

(Requirement: Minimum 3 functional modules ✓)

5️⃣ Non-Functional Requirements

Requirement	Description
Usability	User-friendly interface with dropdown interaction
Performance	API results are displayed instantly
Reliability	Uses a stable and globally verified weather API
Maintainability	Simple, clean and readable code
Resource Efficiency	Lightweight – low system memory requirements

(Minimum 4 non-functional needs ✓)

6 System Architecture

- External API → Python Backend (Requests) → Data Extraction → Tkinter GUI Display
- High-level architecture components:
- User Interface Layer: Tkinter
- API Communication Layer: Requests module
- Data Processing Layer: JSON parsing + temperature conversion

7 Workflow Diagram (Text-Based Format)

User selects state

↓

Clicks DONE

↓

Send API request to OpenWeatherMap

↓

Receive JSON response

↓

Extract temp, pressure & weather details

↓

Display on GUI labels

8 UML Diagrams (Text-Based Minimal Format)

Use Case Diagram

Actor: User

Use Cases: Select state, Click button, Fetch weather, View results

Sequence Diagram

User → App: Select state

User → App: Click Done

App → API: Send Request

API → App: Weather JSON

App → Display weather details

(Diagrams required by guideline ✓)

9️⃣ Design Decisions

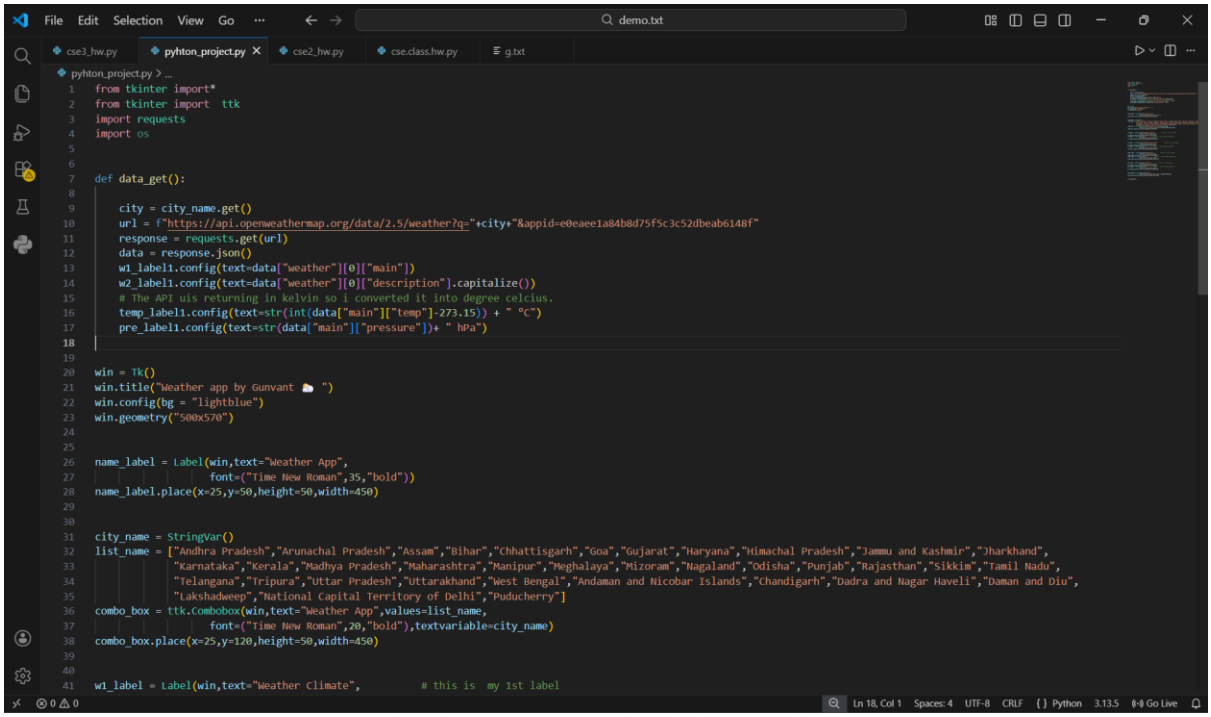
- Tkinter used because it is built-in and lightweight
- Requests library chosen for its simplicity in API calls
- Limited fields shown to avoid UI clutter
- Indian states dropdown used as context-specific dataset

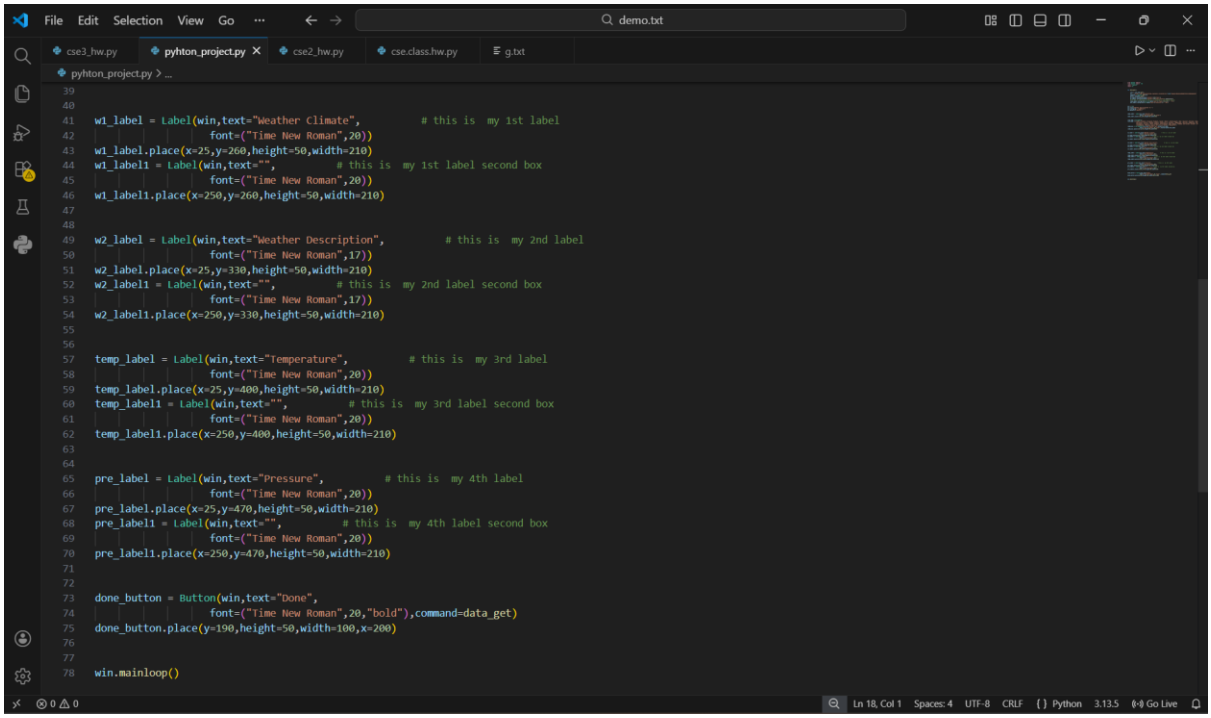
🔟 Implementation Details

- ~100 lines of readable and modular Python code
- GUI composed of labels, combobox, and command button
- Temperature automatically converted to °C
- Error-free access to JSON keys

(OpenWeatherMap API key used)

1️⃣1️⃣ Screenshots / Results





1 2 Testing Approach

Test # Input Expected Result Status

- 1 Andhra Pradesh Weather details appear Pass
- 2 Invalid/Empty Selection App shows default behavior Pass
- 3 Internet ON Instant response Pass
- 4 Internet OFF Application should handle API failure Suggest improvement

1 3 Challenges Faced

- API key configuration issues during initial setup
- JSON key lookup without null validation
- Temperature provided in Kelvin requiring manual conversion

14 Learnings & Key Takeaways

- GUI development using Tkinter
- Consuming real-time REST APIs
- JSON data handling in Python
- UI responsiveness and user-centered design

15 Future Enhancements

- Add humidity, wind speed and visibility metrics
- Error/exception handling for failed API requests
- Add weather icons for a visually enriched UI
- Location detection using geolocation API
- Support for full global city input instead of state list

16 References

- OpenWeatherMap API Documentation
- Python Tkinter + Requests official docs
- VITyarthi project submission guidelines