

**COLLEGE CODE : 8203**

**COLLEGE NAME : A.V.C COLLEGE OF ENGINEERING**

**DEPARTMENT : CSE**

**STUDENT NM-ID :** **5029EE305301B2FA1CF4AA1F1A50DB0D**

**ROLL NO : 820323104061**

**DATE : 08.09.2025**

**Phase1-- PROBLEM UNDERSTANDING AND REQUIREMENTS**

**TECHNOLOGY PROJECT NAME : WEATHER DASHBOARD**

**SUBMITTED BY,**

**s.meenatchi sundari**

**99439 28868**

# Phase 1 — Problem Understanding & Requirements

## Problem Statement

Users often need quick and accurate weather updates for their city or travel destinations. Manually checking weather websites or apps can be time-consuming, and results may vary. The goal is to create a Weather Dashboard that fetches real-time weather data for any city using the OpenWeather API and presents it in a clean, user-friendly dashboard.  
  
This dashboard will be designed to provide a seamless experience to users by integrating a modern user interface with a robust backend powered by Node.js and Express. The system will also ensure scalability and efficiency by including caching mechanisms for repeated queries.

## Users & Stakeholders

| **Role** | **Description** |
| --- | --- |
| End Users | Individuals seeking quick access to current weather data for any location. |
| Developers | Engineers responsible for building, maintaining, and optimizing the dashboard. |
| API Provider | OpenWeather, which supplies real-time weather data via its public API. |

Primary Users:

- Travelers who want weather updates for planning trips.  
- Daily commuters checking city weather before leaving.  
- Students and professionals who plan daily activities based on weather conditions.  
- General users who want instant access to temperature, humidity, and conditions.

Stakeholders:

- Development team (responsible for building and maintaining the app).  
- OpenWeather API provider (third-party dependency offering weather data).  
- Project mentor/manager (reviewing deliverables and performance).  
- End users (who will provide feedback for improvements and usability).

## User Stories

1. As a user, I want to search for a city and instantly view its current weather so that I can plan my day.  
2. As a user, I want to see temperature, humidity, wind speed, and weather description clearly presented in a dashboard view.  
3. As a user, I want the app to handle invalid city names gracefully and show an error message instead of crashing.  
4. As a user, I want faster results for repeated queries (via caching) to save time and avoid delays.  
5. As a user, I want a simple, mobile-friendly UI so that I can access weather data on the go.  
6. As a stakeholder, I want a reliable backend service that can scale with multiple users querying weather data.

## MVP Features

- Search bar for entering city name.  
- Fetch weather data from OpenWeather API in real time.  
- Display temperature, humidity, wind speed, and weather condition in an organized layout.  
- Error handling for invalid cities or API downtime with user-friendly messages.  
- Basic caching with Redis for faster repeated queries.  
- Lightweight and responsive frontend to ensure usability across devices.

## Wireframes / API Endpoint List

Wireframes (basic idea):  
- A clean search bar at the top where users can enter the city name.  
- A weather card showing: City, Temperature, Description, Humidity, Wind Speed, and Weather Icon.  
- An error message display when the city is invalid or the API fails.  
- A simple and intuitive layout ensuring accessibility.  
  
API Endpoints:  
- GET /weather/:city → Returns weather data for a given city in JSON format.  
- Health Check Endpoint → Ensures backend API is running and accessible.

## Acceptance Criteria

✅ Users can enter a city and retrieve real-time weather data without delays.  
✅ Data includes temperature, humidity, description, wind speed, and weather icons for better understanding.  
✅ Errors are displayed for invalid city names in a readable format.  
✅ Responses are returned within 1 second if cached, otherwise fetched from API efficiently.  
✅ Application is mobile and desktop responsive, ensuring accessibility for all users.  
✅ Codebase is modular and follows best practices for maintainability and scalability.