**COLLEGE CODE: 8203**

**COLLEGE: AVC COLLEGE OF ENGINEERING**

**DEPARTMENT: INFORMATION TECHNOLOGY**

**STUDENT NM-ID: au820323205075**

**ROLL NO: 23IT75**

**DATE:**

**Completed the project named as**

**Phase\_\_ TECHNOLOGY PROJECT**

**NAME:**  **Weather Dashboard**

**SUBMITTED BY,**

**NAME: PRADEEP KUMAR R**

**MOBILE NO: 6379897924**

## Problem Statement

Accessing accurate and real-time weather information is essential for planning daily activities, travel, and decision-making. However, users often face challenges

* **Scattered Sources**: Weather information is available on multiple platforms, making it inconvenient for quick access.
* **Performance Issues**: Repeated API calls for the same city increase response time and server load.
* **Error Handling Gaps**: Many applications fail to provide clear messages when invalid cities are searched or when the API service is down.

There is a need for a **centralized weather dashboard** that:

1. Allows users to easily request weather details for any city.
2. Retrieves reliable data from the **Open Weather API** using **Node.js, Express, and Axios**.
3. Handles error cases gracefully to improve user experience.
4. Uses **Redis caching** to reduce redundant API calls and improve performance.

This solution aims to deliver a **simple, fast, and user-friendly weather dashboard** that ensures real-time access to weather conditions with optimized performance.

## Users & Stakeholders

**Users:**

These are the people who directly interact with the Weather Dashboard:

1. **General Public** – Ordinary users who want to quickly check the weather in their city or travel destination.
2. **Travelers & Commuters** – People who depend on real-time weather conditions to plan safe and comfortable journeys.
3. **Students & Employees** – To decide on daily commute (bike, car, bus, etc.) depending on rain/heat.
4. **Event Organizers** – To plan outdoor programs, sports events, or cultural activities without disruptions.
5. **Farmers & Gardeners** – Weather plays a vital role in agriculture, crop protection, and irrigation planning.
6. **Local Businesses (Food delivery, transport, tourism)** – They may use the dashboard to optimize operations based on weather conditions.

**Stakeholders:**

These are individuals or groups who have an **interest** in the system but may not directly use it:

1. **Project Development Team** – Responsible for designing, coding, testing, and maintaining the dashboard.
2. **System Administrators** – Ensure smooth running of the server, API calls, Redis caching, and deployment.
3. **Open Weather API Provider** – Supplies accurate and real-time weather data; their uptime and reliability affect the system.
4. **Educational Institution / Faculty Guide** – If academic project, they oversee progress, evaluate deliverables, and provide feedback.
5. **Investors / Sponsors (optional extension)** – Organizations or departments that may fund or support the project.
6. **End-User Community** – The society that benefits from reliable, centralized, and error-handled weather information.

## Users & stakeholder stories

**User Stories:**

1. **As a general user**, I want to enter a city name and get the weather details**,** so that I can plan my daily activities.
2. **As a traveler**, I want to check the current weather of my destination city, so that I can prepare my travel (clothes, transport, safety).
3. **As a student/employee**, I want to know if it will rain today, so that I can decide whether to carry an umbrella or not.
4. **As an event organizer**, I want to check the weather forecast before scheduling outdoor events, so that I can avoid weather-related disruptions.
5. **As a farmer**, I want to see temperature and humidity levels, so that I can make informed decisions about irrigation and crop protection.
6. **As a local business owner (e.g., food delivery, transport)**, I want real-time weather data, so that I can adjust services and routes accordingly.

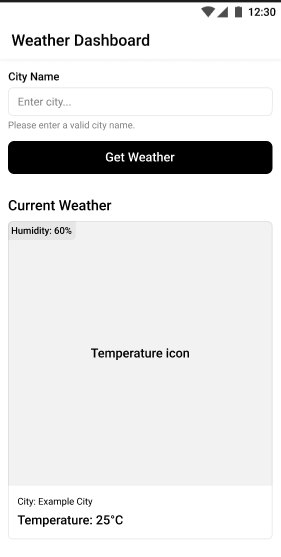
**Stakeholder Stories:**

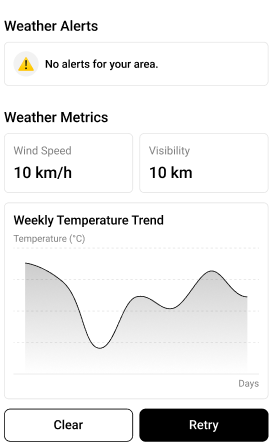
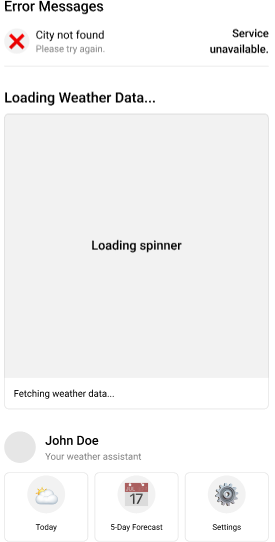
1. **As a developer**, I want to integrate the Open Weather API using Axios, **so that** I can fetch reliable weather data.
2. **As a system administrator**, I want to use Redis caching, so that I can reduce API calls and improve performance.
3. **As an API provider (Open Weather)**, I want my service to be used efficiently, so that API limits are respected and uptime is maintained.
4. **As a faculty guide (academic project)**, I want the system to demonstrate proper software engineering practices, so that it can be evaluated fairly.

## **MVP [Minimum Viable Product] Features for Weather Dashboard**

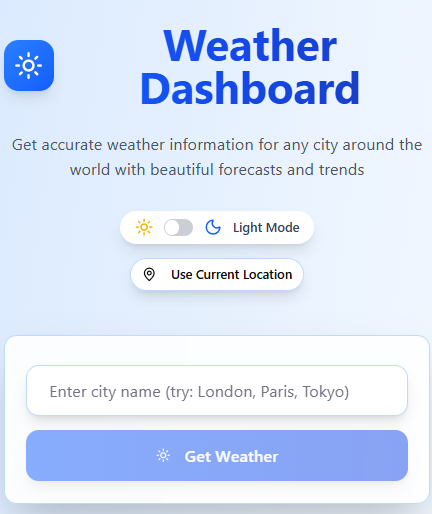
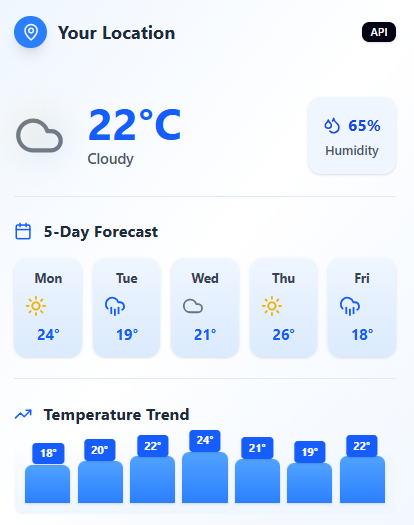
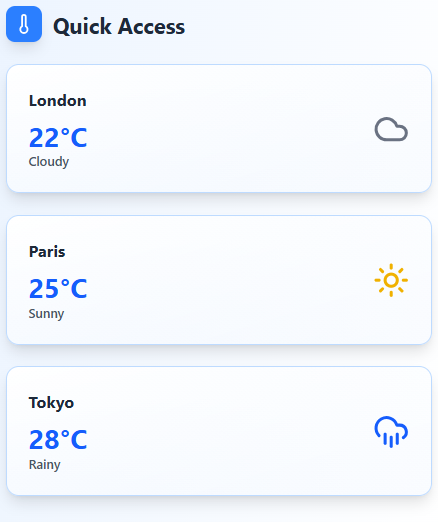
1. **City Weather Search**
   * User can input a city name from the frontend.
   * Request goes to the backend for processing.
2. **API Integration with Open Weather**
   * Backend (Node.js + Express) calls the **Open Weather API** using **Axios**.
   * Retrieves real-time weather data (temperature, humidity, weather condition).
3. **Display Weather Data**
   * Frontend shows results in a simple, user-friendly format.
   * Data includes **City name, Temperature, Humidity, Weather Condition**.
4. **Error Handling**
   * If city not found → show “Invalid City” message.
   * If API service is down → show “Weather service unavailable” message.
5. **Redis Caching (Performance)**
   * Recently searched cities are stored in Redis.
   * Cached results are returned instantly on repeat queries.

## Wireframes & Api Endpoint List

**Mid-Fidelity (Mid-Fi) Wireframes:**

**High-Fidelity (Hi-Fi) Wireframes:**

**  **

**Weather Dashboard – API Endpoint List**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Endpoint** | **Method** | **Description** | **Request Parameters** | **Response** |
| **/weather** | GET | Fetch current weather data for a city | city (query) – name of city | JSON: { city, temp, humidity, condition, source } |
| **/forecast** | GET | Fetch 5-day weather forecast (optional) | city (query) | JSON array: [ {date, temp, condition, icon}, … ] |
| **/health** | GET | Check if backend server is running | city (query) | JSON: { status: "ok" } |

## Acceptance Criteria – Weather Dashboard

1. **City Weather Search**

* Users can enter a valid city name in the input box.
* On clicking “Get Weather”, the dashboard displays City Name, Temperature (°C), Humidity, Weather Condition.
* Invalid or empty city input should show a clear error message: “City not found”.

1. **API Integration & Data Fetching**

* The backend fetches weather data using Open Weather API via Axios.
* For valid cities, data is returned successfully in less than 3 seconds.
* API failures (service down, network error) display friendly error message: “Failed to fetch weather data”.

1. **Caching (Performance)**

* Recently searched cities are stored in Redis cache.
* Repeated searches for the same city return data from cache instantly.
* Cache expiry is set (e.g., 1 hour) to ensure updated weather information.

1. **Error Handling**

* Invalid city names, empty input, or API failures must be handled gracefully.
* Users should always see a user-friendly message, no system errors or crashes.

1. **Responsive UI / Frontend Display**

* Dashboard displays correctly on desktop and mobile screens.
* Result cards, input box, buttons, and error messages maintain proper layout and spacing.
* Minimal scrolling and touch-friendly buttons on mobile.

1. **Optional / Future Features**

* 5-day forecast shows correctly with date, temp, and weather icon.
* Theme toggle (light/dark) works and updates UI dynamically.
* Location-based weather fetch works with user permission.