🌞 Weather Dashboard — Project Documentation

📘 Overview

The **Weather Dashboard** is a full-stack web application built using **React (Vite)** for the frontend and **Node.js (Express)** with **MySQL** for the backend.
It allows users to search for the current weather of any city using real-time data from the **OpenWeatherMap API** and view their last 10 searched cities.

This project demonstrates:

- RESTful API integration
- Database operations (store & retrieve searches)
- Modern responsive frontend UI
- Clear modular architecture

🧩 Features

✅ **City Weather Search** — Enter a city to fetch and display real-time temperature, weather conditions, and icons.
✅ **Last 10 Searches** — Stores and displays the last 10 cities searched.
✅ **Database Integration** — Weather data persisted in MySQL.
✅ **Error Handling** — Handles invalid cities or missing API key gracefully.
✅ **Responsive UI** — Clean and professional dashboard layout.

🏗️ System Architecture

Frontend (React + Vite)

- Provides the UI for users to enter city names and view results.
- Makes API requests to the backend.
- Displays weather details and the last 10 searches.

Backend (Node.js + Express)

- Handles requests from frontend (/api/weather, /api/history).
- Integrates with **OpenWeatherMap API**.

- Connects to MySQL to store and retrieve the last 10 searches.

Database (MySQL)

- Stores weather search history.

External API

- Uses OpenWeatherMap API to fetch live weather data.

🗄️ Database Schema

```
CREATE DATABASE weatherdb;


USE weatherdb;


CREATE TABLE searches (
  id INT AUTO_INCREMENT PRIMARY KEY,
  city VARCHAR(100),
  temperature FLOAT,
  description VARCHAR(255),
  icon VARCHAR(100),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

⚙️ Setup Instructions

🖥️ 1. Clone Repository

```
git clone https://github.com/<your-github-username>/weather-dashboard.git
cd weather-dashboard
```

⚙️ 2. Backend Setup

```
cd server
npm install
cp .env.example .env
```

🧾 Add environment variables in .env:

```
PORT=5000

DB_HOST=localhost

DB_USER=root

DB_PASSWORD=yourpassword

DB_NAME=weatherdb

OPENWEATHER_API_KEY=your_openweather_api_key
```

Run the backend:

```
npm run dev
```

💻 3. Frontend Setup

```
cd ../client

npm install

npm run dev
```

Visit the app in your browser:

http://localhost:5173

🔌 API Endpoints

GET /api/weather?city={cityName}

Fetches weather details for the given city and stores it in the database.

Response:

```
{

  "city": "Pune",

  "temperature": 29.5,

  "description": "clear sky",

  "icon": "01d"

}
```

GET /api/history

Returns the last 10 searched cities.

Response:

```
[
 { "city": "Pune", "temperature": 29.5, "description": "clear sky" },
 { "city": "Mumbai", "temperature": 30.1, "description": "few clouds" }
]
```

🧠 Use-Case Flow

1. **User Interaction**
   The user enters a city name in the input field and clicks "Search".
2. **Frontend Request**
   The frontend sends a request to /api/weather?city=<cityName>.
3. **Backend Processing**
   - The backend fetches data from the **OpenWeatherMap API**.
   - Weather details (city, temp, description, icon) are saved in MySQL.
   - Only the latest 10 searches are retained.
4. **Response to Frontend**
   - The backend returns the weather details.
   - The frontend displays temperature, icon, and conditions.
   - It also updates the "Last 10 Searches" list.
5. **User View**
   - User sees updated weather details instantly.
   - Can re-click a previous search to reload that city's weather.

🧪 Example API Flow Diagram

[User Input]

↓

[React Frontend]

↓ (GET /api/weather)

[Node.js Express Backend]

↓ (Fetch from OpenWeatherMap)

[Weather Data Received]

↓

[MySQL → Store Last 10 Searches]

↓

[Send JSON to Frontend]

↓

[Display Result on UI]

🧰 Tools & Libraries

Layer	**Technology**
Frontend	React (Vite), Axios
Backend	Node.js, Express, dotenv, mysql2
Database	MySQL
API Provider	OpenWeatherMap
Dev Tools	Nodemon, ESLint

🔗 GitHub Repository

https://github.com/ryadav2806/Weather_App

🧑‍💻 Author

Developed by: Ravi Yadav
Project: Full Stack
Technologies: React, Node.js, MySQL
Date: 1st November 2025