Low Level Design (LLD)

WEATHER APP

Ngamlenmang Touthang

Document Version Control

Date Issued	Version	Description	Author
24/01/23	1	Initial commit with temperature and weather condition name data	Ngamlenmang Touthang
25/01/23	2	React toastify added and readme update	Ngamlenmang Touthang

Contents

Abstract

The weather web application is designed to provide users with up-to-date and accurate weather information for a given city around the world. The application utilizes OpenWeathermap API’s to collect data which is then processed and formatted for easy viewing on the user interface. The user interface is designed to be simply and user-friendly, displaying current weather condition with temperature and date. The application is designed to be responsive, adapting to different screen sizes and devices to provide optimal user experience. The application is deployed on a web server, making it easily accessible to users through a web browser.

1. Introduction

1.1. Why this Low-Level Design Document?

The purpose of this Low-Level Design (HLD) Document is to present a detailed description of the Weather app. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraint under which it must operate and how the system will work. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for its approval.

The main objective of the project is to get current weather data of a particular city. Based on the weather data provided by OpenWeatherMap api

1.2. Scope

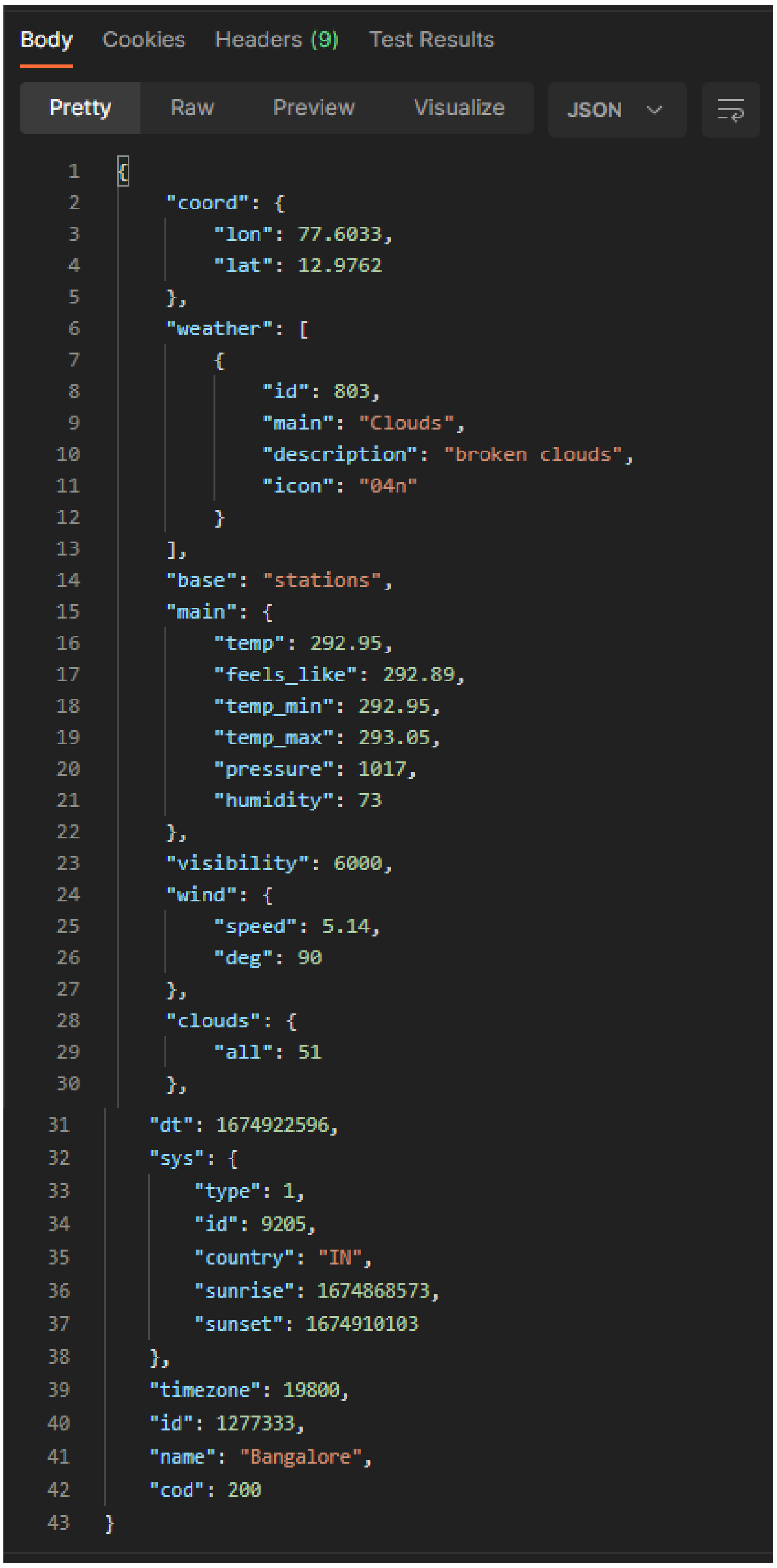
The Low level design documentation presents the structure of the system, such as data will be collected form the API, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system

1.3. Constraints

We will only be selecting only current weather data of a city

2. Technical Specifications

2.1. Dataset



```
{
    "coord": {
        "lon": 77.6033,
        "lat": 12.9762
    },
    "weather": [
        {
            "id": 803,
            "main": "Clouds",
            "description": "broken clouds",
            "icon": "04n"
        }
    ],
    "base": "stations",
    "main": {
        "temp": 292.95,
        "feels_like": 292.89,
        "temp_min": 292.95,
        "temp_max": 293.05,
        "pressure": 1017,
        "humidity": 73
    },
    "visibility": 6000,
    "wind": {
        "speed": 5.14,
        "deg": 90
    },
    "clouds": {
        "all": 51
    },
    "dt": 1674922596,
    "sys": {
        "type": 1,
        "id": 9205,
        "country": "IN",
        "sunrise": 1674868573,
        "sunset": 1674910103
    },
    "timezone": 19800,
    "id": 1277333,
    "name": "Bangalore",
    "cod": 200
}
```

2.2. Weather dataset overview

The weather data set from openweathemap API consist of objects of data namely, object of coordi, array of weathers, key value of base, object of main, key value of visibility, object of wind, object of winds, key value of dt, object of sys, key of timezone, id, name and code.

2.3. Input schema

In the input schema we will be putting any desired city name

2.4. Getting city data

Among the different data object received for a particular city the following data are taken for this system.

- Weather mains which consist of – Clouds, Thunderstorm, Drizzle, Rain, Snow, Clear, Clouds, Mist, Smoke, Haze, Dust, Fog, Sand, Ash, Squall, Tornado
- Weather main temperature which is received as kelvin is converted into celsius

2.5. Logging

The system should log/notify every event so that the user will what the app is running into.

- The system app will notify if the user enter something that is not a name of city.
- The system app will notify the user if the API key is invalid or has issue from the external API provider.

2.6. Database

No database is used, data is provided by an external API (OpenweatherMap API)

2.7. Deployment

The Weather app is deploy via Netlify via github.

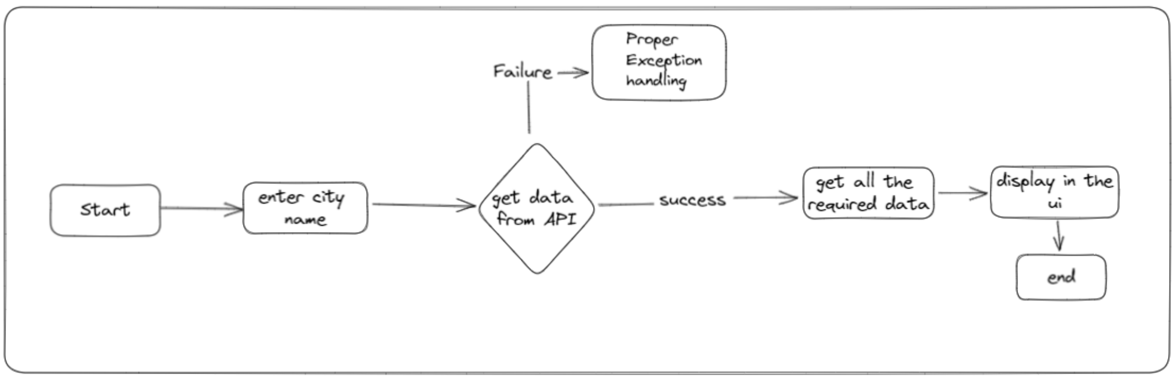
3. Technology stack

Front End	**HTML, CSS, TailwindCSS, React**
Backend	**Not used**
Database	**Not used**
API	**OpenWeatherMap API**
Deployment	**Netlify**
Version control	**Git**

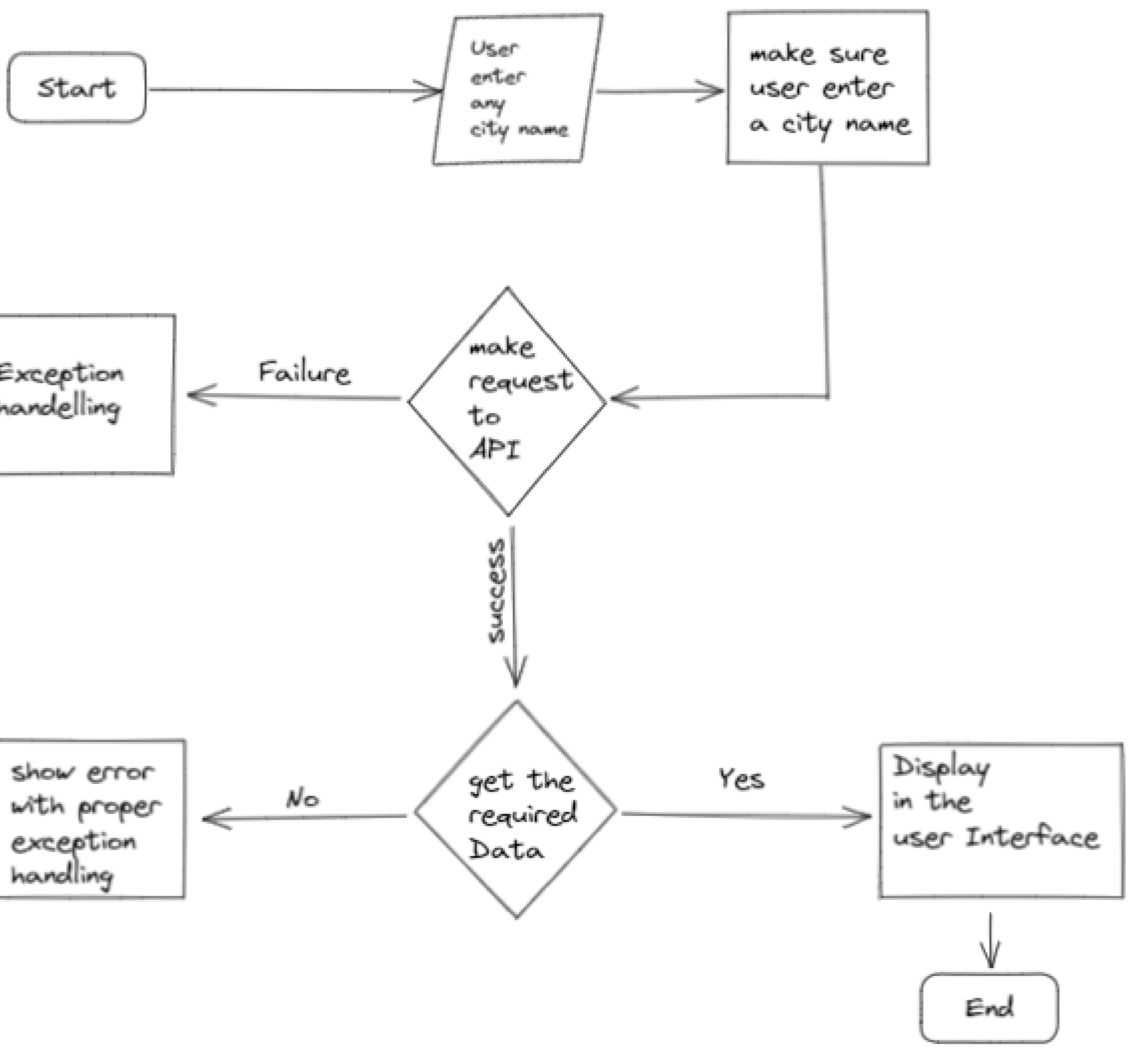
4. Proposed Solution

Based on the city name enter and the weather data received from the external API (openweathermap API) we display some of the weather data to the user interface, provided with a gif image corresponding to the weather condition. Example if the weather is cloudy and cloud gif is display along with the temperature.

5. Design Workflow



6. User I/O workflow



7. Key performance indicators (KPI)

- Daily active users the number of unique users who access the app on a daily basis.
- Retention rate: The percentage of users who continue to use the app over time.
- Time spent in the app: The amount of time users spend using the app per session.
- Accuracy of weather forecast: The percentage of weather forecasts that are accurate.
- Number of location searches: Number of location searches made by users on the app