### **Project Report: Air Quality & Weather Monitoring Dashboard**

#### **Overview**

The **Air Quality & Weather Monitoring Dashboard** is a web-based tool that provides real-time weather and air quality information for cities in South Korea. It includes features like current weather, a five-day forecast, air quality data, weather alerts, and a map showing rain patterns. Users can also compare weather conditions between cities and upload CSV files to visualize data.

This project was designed to be easy to use and visually engaging, using modern tools and technologies.

#### **Main Features**

1. **Current Weather Information**:
   * Displays live weather details such as temperature, humidity, and wind speed using the OpenWeather API.
2. **Five-Day Weather Forecast**:
   * Shows weather predictions for the next five days in a clean, interactive layout.
3. **Air Quality Monitoring**:
   * Displays air quality data with widgets and charts for better understanding.
4. **Weather Alerts**:
   * Provides weather warnings and alerts specific to South Korea. (OpenWeatherAPI)
5. **Rain Precipitation Map**:
   * An interactive map (using Leaflet) that shows rain patterns.
6. **Weather Comparison**:
   * Allows users to pick two cities and compare weather details like temperature, humidity, and wind speed.
7. **Sunrise and Sunset Information**:
   * Shows the sunrise and sunset times for the user’s location.
8. **CSV Upload and Visualization**:
   * Lets users upload CSV files and visualize data in easy-to-read charts.(d3.js)

#### **What I tried and what worked**

1. **Switching APIs**:
   * I first used the WeatherStack API for weather data, which worked well but had a limit on free calls each month. Because of this, I switched to the OpenWeather API, which offers more calls for free.
2. **Mapping Tools**:
   * I tested a custom GIS tool for the precipitation map but found it slow and incompatible with some browsers. Switching to Leaflet solved these problems.
3. **File Upload Visualization**:
   * Initially, I tried handling CSV file uploads and creating charts with plain JavaScript, but it was complicated. I switched to D3.js, which made it much easier.

#### **What I tried and didn’t Work**

1. **Using OpenAPI**:
   * I tried using OpenAPI standards to make the API integration more streamlined. However, the process of creating and maintaining dynamic schemas was too complex for this project, so I dropped the idea.
2. **Custom Chart Libraries**:
   * At first, I used a lesser-known charting library to create graphs, but it was buggy and lacked support for responsive layouts. Switching to Chart.js fixed these issues.

#### **Challenges and Lessons Learned**

* **API Limits**: Learning to balance functionality and cost when choosing APIs was an important lesson.
* **Browser Compatibility**: Testing on different browsers helped ensure the dashboard worked smoothly for all users.
* **Better Libraries**: Picking the right tools (like Leaflet and D3.js) saved time and improved results.

This project is a user-friendly tool for tracking weather and air quality. It faced some challenges but resulted in a reliable and engaging dashboard for users.