Weather Tracking Application - Project Architecture

1. High-Level Overview

This project consists of five main components:

1. **React Front-End**:
    - Users can register and log in.
    - Displays weather data in real-time.
    - Shows weather-based recommendations.
2. **Python Back-End**:
    - Fetches weather data from an external API.
    - Formats data for Prometheus scraping.
    - Serves weather data to the React front-end.
3. **Prometheus**:
    - Scrapes weather data from the Python back-end.
    - Stores historical weather data as time-series metrics.
4. **Alertmanager**:
    - Triggers alerts for extreme weather conditions.
    - Sends notifications when conditions return to normal.
5. **Grafana**:
    - Dashboards for visualizing weather trends.
    - Provides forecasts based on historical data.

Implementation Plan

Step 1: Set Up the React Front-End

- Initialize a React project with authentication (using Firebase/Auth0 or a Node.js-based backend).
- Create components for weather display and recommendations.
- Fetch and display data from the Python back-end.

Step 2: Build the Python Back-End

- Set up a Flask/FastAPI server.
- Fetch weather data from OpenWeatherMap or another API.
- Expose Prometheus metrics in `/metrics` endpoint.
- Serve weather data to the React front-end via REST API.

Step 3: Configure Prometheus

- Install and configure Prometheus to scrape data from the Python server.
- Define Prometheus metrics for temperature, humidity, wind speed, etc.

Step 4: Set Up Alertmanager

- Configure alerts for extreme weather (high temperature, storms, etc.).
- Send notifications via email, Slack, or other services.

Step 5: Install and Configure Grafana

- Connect Grafana to Prometheus.
- Create dashboards for weather visualization.
- Implement forecasting and trend analysis.