Full Stack Engineering Challenge: J250

Real-Time Dashboard

Background

Real-time data is important for modern applications, especially for tasks like monitoring time-sensitive information. In this challenge, you will build a real-time dashboard for greenhouse sensors that monitor temperature, humidity, and air quality. The goal is to create an efficient and interactive application that handles incoming data smoothly and displays it clearly.

Objective

Your task is to build a React-based application that shows live sensor data. It should process and display this data in real time while allowing users to interact with it. The dashboard should support sorting, filtering, and pagination.

Requirements

Functional Requirements:

1. Display the sensor data in a user-friendly way. (Be prepared to justify any choices)
2. Allow users to sort data by timestamp, showing the most recent data first.
3. Add filters for temperature, air quality, and humidity.

Technical Requirements:

1. Build the application using React and TypeScript. Do not use external libraries.
2. Use efficient state management to handle the data and user interactions.
3. Optimize performance to ensure the dashboard works well during continuous updates.
4. Handle errors gracefully and provide fallback options for invalid data or unexpected issues.
5. Deploy the application on a platform like GitHub Pages, Vercel, or Netlify.

Evaluation Criteria

1. The application should meet all functional and technical requirements.
2. It should run efficiently, even with frequent updates from many sensors.
3. The code should be clean, modular, and follow best practices for TypeScript.
4. The deployed version should be accessible and work as expected.
5. Include clear and well-organized documentation that explains the project, its structure, assumptions, and includes performance analysis.

Bonus Features

1. Implement responsive design for mobile view.

Submission Instructions

1. **Live Application:**
   - Provide a link to the live version of your application, hosted on a platform like GitHub Pages, Vercel, Netlify etc.
2. **Source Code Repository:**
   - Include a link to the public repository (e.g., GitHub) where the submission files are hosted.
3. **README File**:

- A detailed `README.md` that includes:
  - A summary of the project and its main features.
  - An explanation of the project structure and key components.
  - A performance analysis, including specific performance metric (e.g, memory usage, update speed, etc.).

Starter Code

Use the starer code provided in the email to simulate real-time data updates. This code will help you focus on the dashboard logic.