Fullstack Developer Home Task – Interactive Dashboard with Firebase

Your task is to build a fullstack web application using **React**, **Firebase Firestore**, **Firebase Cloud Functions**, and **Firebase Authentication**. The app will display and manage user traffic data, shown both in a table and as a chart. This task is designed to evaluate your skills in frontend/backend integration, secure data handling, code structure, and UX decisions.

Core Requirements:

Frontend (React)

- Build a responsive UI that displays a table and a chart showing user traffic data.
- Implement sorting (e.g., by date or number of visits) and filters (e.g., date range).
- Provide a form to **add**, **edit**, and **delete** entries.

Data Structure

Store the following data in Firestore (collection: `trafficStats`):


```
[
  { "date": "2025-03-01", "visits": 120 },
  { "date": "2025-03-02", "visits": 140 },
  { "date": "2025-03-03", "visits": 98 },
  { "date": "2025-03-04", "visits": 132 },
  { "date": "2025-03-05", "visits": 101 },
  { "date": "2025-03-06", "visits": 87 },
  { "date": "2025-03-07", "visits": 94 },
  { "date": "2025-03-08", "visits": 178 },
  { "date": "2025-03-09", "visits": 164 },
  { "date": "2025-03-10", "visits": 112 },
  { "date": "2025-03-11", "visits": 106 },
  { "date": "2025-03-12", "visits": 133 },
  { "date": "2025-03-13", "visits": 90 },
  { "date": "2025-03-14", "visits": 124 },
  { "date": "2025-03-15", "visits": 110 },
  { "date": "2025-03-16", "visits": 175 },
  { "date": "2025-03-17", "visits": 188 },
  { "date": "2025-03-18", "visits": 147 },
  { "date": "2025-03-19", "visits": 133 },
  { "date": "2025-03-20", "visits": 119 },
  { "date": "2025-03-21", "visits": 102 },
  { "date": "2025-03-22", "visits": 111 },
  { "date": "2025-03-23", "visits": 154 },
```

```
  { "date": "2025-03-24", "visits": 162 },
  { "date": "2025-03-25", "visits": 120 },
  { "date": "2025-03-26", "visits": 108 },
  { "date": "2025-03-27", "visits": 113 },
  { "date": "2025-03-28", "visits": 95 },
  { "date": "2025-03-29", "visits": 142 },
  { "date": "2025-03-30", "visits": 170 },
  { "date": "2025-03-31", "visits": 128 },
  { "date": "2025-04-01", "visits": 105 },
  { "date": "2025-04-02", "visits": 87 },
  { "date": "2025-04-03", "visits": 156 },
  { "date": "2025-04-04", "visits": 131 },
  { "date": "2025-04-05", "visits": 122 },
  { "date": "2025-04-06", "visits": 149 },
  { "date": "2025-04-07", "visits": 95 },
  { "date": "2025-04-08", "visits": 143 },
  { "date": "2025-04-09", "visits": 137 },
  { "date": "2025-04-10", "visits": 128 },
  { "date": "2025-04-11", "visits": 109 },
  { "date": "2025-04-12", "visits": 117 },
  { "date": "2025-04-13", "visits": 138 },
  { "date": "2025-04-14", "visits": 160 },
  { "date": "2025-04-15", "visits": 151 },
  { "date": "2025-04-16", "visits": 100 },
  { "date": "2025-04-17", "visits": 134 },
  { "date": "2025-04-18", "visits": 141 },
  { "date": "2025-04-19", "visits": 108 },
  { "date": "2025-04-20", "visits": 157 },
  { "date": "2025-04-21", "visits": 120 },
  { "date": "2025-04-22", "visits": 99 },
  { "date": "2025-04-23", "visits": 126 },
  { "date": "2025-04-24", "visits": 153 },
  { "date": "2025-04-25", "visits": 115 },
  { "date": "2025-04-26", "visits": 130 },
  { "date": "2025-04-27", "visits": 98 },
  { "date": "2025-04-28", "visits": 118 },
  { "date": "2025-04-29", "visits": 167 },
  { "date": "2025-04-30", "visits": 148 }
]
```

Backend (Cloud Functions)

- Create HTTP-triggered Firebase Cloud Functions to handle all operations:
  - `GET` to fetch all traffic entries
  - `POST` to add a new entry
  - `PUT` to update an existing entry
  - `DELETE` to remove an entry
- Do **not** allow direct Firestore access from the frontend.

Authentication

- Use Firebase Authentication (Email/Password or Google)
- Only logged-in users should access the dashboard
- (Optional bonus) Simulate role-based access: viewers can only read, editors can create/update/delete (based on email or UID check in backend)

Charting

- Use any open-source chart library to display visit data.
- Support **view toggling** between daily, weekly, and monthly (aggregate data accordingly).

Project Structure & Quality

- Organize the code as if it were going to production
- Use environment variables where needed
- Write clean, modular, and readable code

Bonus (Optional but Impressive):

- Deploy the frontend (e.g., to Firebase Hosting or Vercel) and backend functions (to Firebase)
- Add unit tests or integration tests for either frontend or backend
- Include pagination or infinite scroll in the table

Time Limit:

Try to limit yourself to **6–8 hours**. This doesn't need to be production-ready, but your code, logic, and project structure should reflect solid development practices.

Submission:

Please share a **GitHub repository** (or ZIP) with:

- All code (frontend and backend)
- A `README.md` file with instructions to run the project locally
- Working link **fully deployed**