Date: 25 AUG 2025

Amenses Task - Collaborative Event & Polling Platform

Objective

Build a **Collaborative Event & Polling Platform** using the **MERN stack** and deploy it using free hosting services.

Backend (Node.js + Express + MongoDB)

Features:

1. **User Authentication**
    - Signup/Login functionality
    - Use **JWT authentication** for securing APIs
2. **Events Management**
    - Create an event with:
        - Title
        - Description
        - Date options (array of possible dates/times)
        - Participants list
    - Update/Delete an event (only allowed for event creator)
    - Invite other registered users to join the event
3. **Polls within Events**
    - Each event must include a poll (e.g., “Choose a suitable date” or “Pick a restaurant”)

    - Participants can vote on poll options
    - Votes should be stored and updated properly
4. **Basic Notifications**
    - When a user is invited to an event, they should see the invitation in their dashboard
    - *(No need for WebSockets or push notifications — simple REST fetch is sufficient)*

Frontend (React / Next.js optional)

Features:

1. **Authentication**
    - Signup/Login pages
2. **Dashboard**
    - List of events created by the user
    - List of events the user has been invited to
3. **Poll Interaction**
    - Option to vote in polls for invited events
    - Ability to view poll results (refresh/re-fetch results to see updates)

Architecture Expectations

- **Folder Structure**: Maintain clean structure
- **Data Modeling**: Show clear relationships between entities (**User** ↔ **Event** ↔ **Poll**)
- **Authentication & Access Control**:
    - Only event creator can edit/delete their events
    - Invited users can only vote
- **Business Logic Separation**: Keep controllers and services clean; don't put everything in one file
- **Environment Variables**: Store secrets (JWT secret, DB connection string) in `.env`
- **Error Handling & Validation**:
    - Send proper error messages (not raw stack traces)
    - Validate inputs (e.g., title required, vote option valid, etc.)

README.md Expectations

Your **README.md** must include:

1. Project setup steps (how to run backend & frontend locally)
2. Tech stack used (libraries/frameworks)
3. Live links (frontend + backend)
4. Architecture Decisions
    - Why you chose your structure
    - How you modeled data
    - How authentication is handled
5. Challenges Faced & Solutions

Deployment

- **Backend**: Deploy on Render, Railway, or any free hosting service
- **Frontend**: Deploy on Vercel, Netlify, or any free hosting service
- **Database**: Use **MongoDB Atlas** (cloud-hosted MongoDB)

Submission Requirements

- **GitHub Repository**
    - Must contain both frontend and backend code (separate folders)
    - Proper commit history (meaningful commit messages, not a single “final commit”)
- **Live Links**
    - Backend API Base URL
    - Frontend Live App Link
- **README.md**
    - Must include all details listed above

Naman Jain | Amenses Innovations Pvt. Ltd.
Naman.jain@amenses.com | 9926400955