Assignment: Collaborative Kanban (Real-Time Boards)

Important Instructions

- Strictly follow submission guidelines - any violation will result in disqualification.
- Zero tolerance for plagiarism or cheating - all suspected cases will be rejected immediately.
- Send your resume only if you can complete the assignment at least partially.
- You may be asked to modify your submitted code during a video interview.
- No additional clarifications will be given, interpret and implement requirements independently.
- Focus on completing all base features - a more complete, functional solution significantly improves your selection chances.

Objective

Build a mini real-time collaborative Kanban platform where users can:

1. **Create & View Boards:** Users create boards with columns (e.g., Todo/In Progress/Done) and manage cards.
2. **Real-Time Collaboration:** Multiple users can edit boards concurrently with live updates.
3. **Presence & Notifications:** See who's online on a board and receive in-app notifications on changes.
4. **Post-Action Communication:** Send email notification (optional) for significant events (e.g., card assigned).

Tech Stack

- **Frontend:** React.js / Vue.js
- **Backend:** Node.js / Django
- **Real-Time Communication:** WebSockets
- **Database:** Supabase (PostgreSQL) with Sequelize models
- **Redis:** Upstash (for presence, ephemeral locks, optimistic updates)
- **Email:** SendGrid

Core Features

1. **Board & Column Management:** Create boards, add columns, reorder columns; store in Supabase using Sequelize.
2. **Card CRUD & Assignment:** Create/update/delete cards with title, description, assignee, labels, due date; drag-and-drop between columns.
3. **Real-Time Sync:** All viewers of a board see live updates to columns/cards via WebSockets (create, move, edit).
4. **Presence & Typing Indicators:** Show online users on a board; optional typing/editing indicators per card.
5. **Optimistic UI & Conflict Handling:** Apply optimistic updates on the client; reconcile with server acknowledgements. Prevent conflicting writes with basic versions (updated_at) or lightweight locks in Redis.
6. **Notifications:** In-app notifications for card assignments, mentions (@user), and significant board changes.
7. **Audit Log:** Append-only log of board events (CardCreated/CardMoved/CardUpdated) for troubleshooting and transparency.

8. **Admin Panel (Bonus):** View active boards, connected users, and recent events; throttle/ban spammy clients.

Deliverables

- A link to the hosted site
- GitHub public repo URL containing all the files
- PDF Resume

Evaluation Criteria

- **Deployment:** Bundle both frontend and backend into a single Dockerfile (do not use docker-compose). Deploy the containerized app on Render.com.
- **Real-Time Functionality:** Implement WebSocket handling for live board updates, presence, and notifications.
- **Shared State Management:** Use Redis for presence tracking, ephemeral locks, and fast lookups.
- **Database Interaction:** Connect to Supabase using Sequelize models for CRUD operations and event logging.
- **UI and Flow:** Smooth drag-and-drop, sensible conflict resolution, and clear status indicators.
- **Reliability:** Idempotent updates; recover gracefully from disconnects; audit trails for edits.

Bonus

- **CI/CD:** GitHub Actions for automated build and deploy on push.
- **Comment Threads & Mentions:** Per-card comments, @mentions with notifications.
- **Board Templates:** Pre-configured workflows; export/import boards as JSON.
- **Access Control:** Board roles (owner, editor, viewer) with server-side authorization.

Submission

- **Link:** https://forms.gle/dYPKZmYPBjnWDNEx9
- **Deadline:** 23:59 IST Saturday, 11 September 2025

Resources (Use Only Free Tiers)

- **Redis:** Upstash
- **Database:** Supabase (Postgres)
- **Docker Hosting:** Render.com
- **Free Instance Warmup (Cron Jobs):** cron-job.org
- **Email API:** SendGrid

Note:

Free-tier instances may automatically go idle/sleep. To avoid delays caused by cold starts, set up cron jobs to periodically send requests and keep the instance warm/active.