Leave Management Dashboard

Admin & Candidate (Node, React, MySQL, JWT)

Problem Description

Build a lean leave app with two roles: Admin and Candidate. Admin invites candidates by email; candidates self-register only through invite links. Candidates submit leave requests (CL, SL, EL, ML) with documents where required. Admin reviews and approves/rejects with a required comment and manages annual entitlements per candidate. Approved requests deduct balances instantly. Once the allowed quota for a leave type is exhausted, the Candidate cannot apply for that leave category anymore. Keep screens and endpoints minimal, while ensuring the core operations run smoothly.

Task Scope

Authentication & Roles

- JWT login for Admin and Candidate; passwords hashed with bcrypt.
- RBAC checks on protected routes.
- **Invitation-only registration**: Admin sends invite → Candidate completes sign-up via invite token.

Candidate

- View my balances and my request history.
- Create a leave request: category (CL, SL, EL, ML), startDate, endDate, reason.
- Upload document when rules require.
- See request status updates.
- **If entitlement balance is exhausted for a category, Candidate cannot submit a new request for that category.**

Admin

- Send invitations (email) for new candidates; revoke/resend invites.
- Set/adjust annual entitlements per candidate and category.
- List and filter all requests by status/date/candidate.
- Approve or reject with a required comment.

Balances & Rules

- Default categories: CL (6), SL (6), EL (12), ML (180) — editable by Admin per candidate/year.
- Count working days only (exclude Saturday and Sunday).
- No half-days.
- Prevent negative balances:
    - Block leave application if entitlement balance < requested days.

    - Candidate sees a clear error message (e.g., *“No balance left for Sick Leave”*).
- Documents required: SL if effective working days > 2; ML always.

Files

- Store to /uploads with unique filenames; max 5 MB; types: pdf, jpg, jpeg, png.
- Files downloadable from request detail.

Audit Log

- Record key actions with who, when, what changed (invite sent/accepted, create request, approve, reject, entitlement change), with optional note.

Deliverables (Mandatory)

- GitHub repository link.
- README with setup, environment variables, backend/frontend run steps, and assumptions.
- Export of the ChatGPT/Claude thread if used.
- Short video (≤ 4 minutes) demonstrating: Admin invites → Candidate registers via invite → Candidate applies → Admin approves/rejects → balances update; SL/ML document rule; quota exhaustion preventing a new request.
- Deployed link if feasible.
- Database migration/DDL scripts and a simple bootstrap step to create the **first Admin** user (document in README).

Timeline

Expected effort: 3–4 hours. Submission deadline: 3 days from assignment.

Evaluation Points

- **Correctness**: weekend exclusion; document enforcement (SL > 2, ML always); approval flow; **balance deduction and quota exhaustion enforcement**; no negative balances.
- **Data & API design**: sensible MySQL schema, useful indexes, transactions for balance updates; clear routes and consistent error shape.
- **Security/RBAC**: JWT, role checks, invite token validation, upload type/size validation.
- **Reliability & clarity**: meaningful validations and errors; atomic updates; simple, predictable flows.
- **UX simplicity**: minimal, easy pages for Admin and Candidate.
- **Operations**: migrations/DDL provided; README is crisp; app runs cleanly end-to-end.