Department of Natural Applied Science

Texas A&M University Victoria

CollabTrack A Team & Personal Project Organizer

Team: Uche, Michael, Jonathan

Instructor: Professor Wenjuan Huang

Course: COSC 4320 Fall 2025

Abstract

CollabTrack is a lightweight goal and project organizer that supports both single-user and small team workflows. Users create projects, break them into tasks, assign due dates, and keep work moving with a simple comments/activity log inside each project. For shared work, invited collaborators see the same board, can pick up tasks, and track progress together. The emphasis is on reducing procrastination and context-switching: the dashboard foregrounds “today” items, late tasks, and upcoming deadlines, while project-level activity keeps communication tied to the work itself. To ensure we finish a practical, high-quality demo this term, we’re intentionally keeping the stack simple—**Node.js + Express + EJS + SQLite**—with optional reminders and a future “AI-assisted task” idea reserved as a stretch goal. The result will be a clean, usable prototype that demonstrates clear value for students, small clubs, and anyone who wants a straightforward way to plan, execute, and complete projects on time.

Project Description

Main goals

Build a simple, reliable **goal/project organizational tool** that helps individuals and teams plan work, track progress, and meet deadlines without the bloat of enterprise project suites. The near-term outcome is a working prototype suitable for a GUI demo, showing end-to-end flows from sign-up to project/task completion.

Main functionality & characteristics

User accounts: Registration, login, logout.

Projects: Personal or shared (invite-only) spaces.

Tasks: Title, description, assignee, due date, priority, and status (to do / doing / done).

Communication: Per-project comments / activity log to keep decisions with the work.

Deadlines & reminders: In-app reminders; optional email for demo if time permits.

Privacy: Personal projects are private; shared projects visible only to invitees.

This list reflects your core feature set and tight scope for a successful term demo.

Intended users & key usability goals

Primary users:

General public seeking a simple organizer; students/teachers; small teams and clubs; people who struggle with timeliness and want better habits. **Usability goals**: (1) capture a new task in <10s, (2) see "what matters today" at a glance, (3) keep chat/comments tied to tasks to avoid context loss, and (4) low-friction invites for collaborators.

Potential for further development

Integrations: Calendar export/sync (Google/Outlook), optional email notifications.

Social (opt-in): Invite-only collaboration and shareable project templates.

Analytics: Completion rates, pace/velocity trends, "at-risk" tasks.

These reflect your listed growth areas and remain optional beyond the core demo.

Challenges & intended technology

Challenges:

(1) security & privacy (safe password storage, access control).

(2) scalability path (start simple, design for growth).

(3) scope control to finish the demo.

(4) communication UX (start with comments/activity log rather than full real-time chat).

Proposed Stack (pragmatic for this term):

- **Backend:** Node.js + Express (routing/API)
- **Frontend:** EJS templates (server-rendered views)
- **Database:** SQLite (local dev/early demo), with a clear path to Postgres later
- **Auth:** JWT + bcrypt
- **Scheduling/Email (optional):** node-cron, Nodemailer
- **Testing/Quality:** Jest/Supertest for routes; Prettier/ESLint for consistency; basic CI on GitHub

Why this stack: It prioritizes delivery speed and stability. Server-rendered views keep complexity low, while a clean repository/DAO layer keeps the door open for a future React/PostgreSQL upgrade without rewriting the app.

How this project supports professional growth

- Practice real collaboration (issues, branches, PR reviews).
- Experience a full stack (routing, views, persistence, auth).
- Apply an incremental process (small vertical slices, weekly milestones, retrospectives).
- Build deploy/readiness habits (basic logging, error handling, and simple health checks).

Team members (roles)

- **Uche — Backend/API (Lead):** Owns API design, DB schema, authentication, and deployment basics.
- **Michael — Frontend (EJS/UX):** Auth pages, dashboard layout, project/task forms, and accessibility checks.
- **Jonathan — Backend & QA:** Implements routes, adds tests (Jest/Supertest), and supports CI reliability.

External advisors

None formally. Instructor guidance will be requested for scope, milestones, and evaluation criteria.

Market Potential or Open-Source Significance

Market need. Students, clubs, and small teams often outgrow basic notes but don't want heavyweight project suites. Many rely on calendar apps (broad but not task-centric) or chat apps (great for conversation, weak for structured follow-through). CollabTrack occupies the middle ground: task-first planning, due dates, and a simple activity log—fast to adopt, easy to teach, and effective under time pressure.

Differentiation.

1. **Simplicity:** No complex setup or configuration; start a project and invite collaborators instantly.

2. **Task-First Workflow:** Work units (tasks) are the center; comments and activity live where the work happens.

3. **Follow-Through:** Subtle nudges and optional reminders push items to done without overwhelming users.

Barriers & how we address them. The space is crowded. We focus on a narrow persona (students/small teams), fast onboarding, and a clean UI. The MVP avoids feature creep (e.g., full Kanban plugins, complex roles) and proves value with just enough functionality to coordinate, track, and finish assignments.

Operational considerations. Usage clusters near deadlines. We'll paginate lists, optimize common queries, and queue reminders to smooth load. Documentation (quick-start guide, inline tips) and a minimal help page keep support overhead low for a student-run project.

Scalability path. Start with SQLite; design a clean data access layer to swap in PostgreSQL later. Keep the server stateless, log requests and errors, and add basic metrics. This makes future horizontal scaling and real-time features straightforward.

Open-source value (optional). CollabTrack can double as a teaching repo: clean Express patterns, EJS best practices, and a minimal but real app for students to extend. Issue templates, CONTRIBUTING.md, and starter tasks can help onboard new contributors.

Novelty. CollabTrack's novelty is disciplined minimalism: invite-only collaboration, task-first design, and comments/activity embedded directly in the workflow—plus an optional AI assist to speed up task entry—delivering just enough power to finish real work, fast.

Corrections & clean-up applied

- Unified title and naming (CollabTrack) across all sections.
- Fixed grammar, capitalization, and role labels for consistency.
- Tightened scope to ensure a deliverable, demo-ready MVP.
- Clarified stack and migration path (SQLite → PostgreSQL) and why it's chosen.
- Streamlined usability goals into concrete, measurable targets.