Tidsflyt App Summary

Repository: creaotrhubn26/tidsflyt

What it is

Tidsflyt is a full-stack web application that combines workforce time tracking, case reporting, and a role-based admin/vendor portal.
It also includes a CMS and blog layer to manage public pages, forms, SEO metadata, and content publishing.

Who it's for

Primary persona: staff and team leads in service organizations who log work, manage case reports, and coordinate through admin workflows.
Explicit persona naming by product team: Not found in repo.

What it does

- Tracks time entries with create/list/delete, bulk entry, activity lists, report exports, and chart/report endpoints.
- Persists live timer sessions (elapsed, paused, running state) for resume/sync behavior.
- Supports case-report lifecycle: create/edit/submit, comments, unread counts, and admin approve/reject/feedback flows.
- Enforces role-based access across app routes and APIs (e.g., super_admin, vendor_admin, team roles).
- Provides vendor API management: API enablement, key issuance/revocation, and vendor-scoped access controls.
- Offers CMS/blog operations: builder pages, section templates, versions, media, forms/submissions, SEO, posts, comments, and analytics.

How it works (repo-evidenced architecture)

- Frontend: React + Vite SPA with lazy-loaded pages, wouter routing, React Query state, ThemeProvider, and AuthGuard-protected routes.
- Backend: Express HTTP server; registerRoutes wires feature routes plus SmartTiming routes and vendor API router.
- AuthN/AuthZ: Passport Google OAuth + session store in Postgres, plus JWT paths for admin APIs and role checks in shared role helpers.
- Data layer: PostgreSQL via node-postgres pool and Drizzle ORM with shared schema types used by server and client.
- Flow: UI calls /api endpoints -> route handlers validate/auth -> Drizzle/SQL persistence -> JSON responses consumed by React Query.

How to run (minimal)

- Create .env from .env.example and set DATABASE_URL and SESSION_SECRET (Google OAuth keys optional for full login).
- Install deps: npm install
- Apply schema: npm run db:push
- Start dev server: npm run dev (serves API + client on PORT, default 5000)