Mohamed Safty

Senior Product Engineer (AI-native, full-stack)
Menofia, Egypt | Remote | me.a.safty@gmail.com
LinkedIn | Upwork | GitHub

SUMMARY

Product engineer with 4+ years shipping TypeScript products on web, mobile and backend. I do my best work next to a founder. I find the product gap, decide how it should work, build it end to end and get it through App Store review.

On Universe Dawn I'm the lead engineer working with the founder. I moved the desktop to TanStack Start, replaced generated API clients with ts-rest contracts and one shared state package. I built the first version of the mobile app in 2024, then rebuilt it on Expo and led its overhaul. I also built the AI-agent delivery platform the team ships with, which took my merged output up 10x. I built PitchHub (live on the App Store and Google Play) and the Meezan ERP frontend on my own. Top Rated on Upwork, 100% Job Success, 5.0 rating.

SKILLS

Product: MVP scoping, UX and interaction design, onboarding and activation, monetization and store billing, App Store and Google Play review policy, analytics and funnel plans
Frontend: React, TanStack Start/Router/Query, Next.js, TypeScript, Zustand, Tailwind CSS, shadcn/ui
Mobile: React Native, Expo, Expo Router, EAS Build and Update, push notifications, PWA
Backend: Node.js, ts-rest, Express/tsoa, WebSockets, RabbitMQ, MongoDB, MySQL, Supabase (RLS, edge functions), Convex
Platform: pnpm monorepos, shared packages, GitLab CI, Docker, Helm review environments, Jest, Vitest, Playwright
AI engineering: Claude Code, Codex, agent rules, autonomous fix workers, review and evidence loops

EXPERIENCE

Lead Engineer, Mobile & Platform (contract), Team Parallax

Universe Dawn | Sep 2022 to Present | Remote

Universe Dawn is a sci-fi strategy MMO with a React desktop client, an Expo app for iOS, Android and mobile web, and Node.js microservices. I'm the main engineer, working directly with the founder, and I own the client apps and the API platform.

Platform and architecture

- Moved the desktop client from Next.js to TanStack Start and Vite. After login the game runs as a client-side app, so a client router with typed loaders fits it better than a server-first framework. Auth moved into route loaders with signed session cookies, and config now loads at runtime so one image deploys to every environment.
- Replaced the generated API client libraries with ts-rest contracts at the gateway, with typed request and response schemas and authorization per route. The generated clients kept drifting from the gateway and every change meant regenerating them.
- Built `shared-state`, one package for queries, mutations, WebSocket subscriptions, sessions and the persisted cache, used by desktop and mobile. Each app used to keep its own copy, so a fix often landed on one side only. Sharing that logic is what made the mobile overhaul possible.
- Migrated the mobile app to Expo in 2026 and kept it current through SDK 57 and 58. Set up GitLab CI for EAS: each MR gets a preview build with QR codes, and releases ship as OTA updates unless the native fingerprint changed. Also served the app as an installable PWA at m.universe-dawn.com.
- Cut mobile config revalidation from a 764 KB download (7 to 9 s on staging) to a 4-byte hash check (about 0.7 s), and held the native splash until the first real screen. On desktop, sidebar navigation became instant through viewport preloading and caching tuned to the data the WebSocket already keeps fresh.

Product decisions

- Built the first version of the mobile app in 2024: sign-in and registration, i18n, navigation and the core game screens.
- After the Expo move, led an overhaul of the mobile app, 93 MRs merged through one integration branch. It brought fights, fleets, shipyard, chat, quests, the academy and premium up to desktop parity and redesigned the screens for touch.
- Redesigned mobile navigation. Five popup menus on one flat stack became a bottom tab bar with a stack per tab. Back no longer jumps across sections, the tab bar shows where you are, and every URL kept working, so deep links didn't break.
- Designed the tactical scan radar for mobile and desktop. Contacts sit on a square radar inside an x/y/z coordinate frame, with pinch to zoom and a tap that opens a spot card with "Fly here". Before this, players read coordinates off a table.
- Designed push notification opt-in. The app asks after onboarding or the first fleet send instead of at launch. "Not now" brings the prompt back after 3 days, at most 3 times, and players choose categories (fights, queues, diplomacy, general). The backend sends through Expo Push in 7 languages and disables dead devices based on push receipts.

- Shipped chat reporting, blocking and a moderator queue to meet Apple guideline 1.2 on user-generated content, and limited message deletion to senders and moderators.
- Wrote the PostHog analytics plan: matching events on mobile and desktop, one shared event core so the clients can't drift, core funnels, and consent for session replay.

AI-agent delivery platform

- Built and run the platform that lets coding agents ship safely in a large TypeScript monorepo. It has agent rules, seeded review scenarios with a CI check that changes to gameplay services include one, isolated dev stacks, a shared build queue, EAS preview builds for each MR, fixed git hooks and pnpm 11, and test pruning. Autonomous workers (a Discord Codex worker and a PostHog error intake) publish fixes through the same review. My merged output went up 10x (Oct 2026 against my Feb to Sep monthly average) with screenshot or recording evidence on every MR.

Earlier work (2022 to 2025): React desktop features, Express/tsoa services on RabbitMQ, MongoDB and MySQL, alliance and fleet rules, a weapon class rename that normalized legacy ship data, and React, Next.js and Arwes upgrades.

PitchHub, founder and angel investor matching app

Solo developer for a client founder | May 2025 to Present | App Store | Google Play

- Scoped the v1 MVP with the founder: onboarding by role, video pitches as the core format instead of decks, swipe discovery, matching and push notifications.
- Set the matching rules. An investor's like creates a pending match and only the founder can confirm it, enforced in Postgres with a restrictive RLS policy. The contact email appears only after confirmation.
- Let visitors browse published pitches before signing up, so they see what the app offers before the account wall.
- Chose Expo and Supabase so one developer could ship to both stores. The backend has 6 tables, 21 RLS policies, database triggers for match notifications and 2 edge functions. Sign-in covers email, Google and Apple, and EAS channels publish an OTA preview on every push to master.
- Kept media costs down. Video goes to ImageKit with 5-minute signed upload tokens from an edge function instead of a Supabase Storage bucket, and I dropped image transforms when Supabase made them a Pro feature.
- Leading v2. It adds Free, Standard and Premium tiers with a 30-day trial, store-native billing through RevenueCat (free under $2.5k monthly revenue) instead of Stripe web checkout, and a first paywall on pitch views. Founder and investor roles merge into one profile with two-way matching, and the backend moves from Supabase to Convex, migrating existing users instead of resetting them.

Meezan, multi-tenant ERP for gold and jewelry businesses

Lead frontend engineer (contract) | Aug 2025 to Present

- Own the frontend architecture, UX and visual design of an Arabic and English accounting and ERP dashboard for Saudi gold and jewelry shops. The client's team owns the backend API.
- One subdomain per business. 99 routes across 36 feature modules cover sales, purchase and return invoices, vouchers, the chart of accounts, journals, gold karat balances, inventory counts, VAT reports and ZATCA e-invoicing per branch.
- Designed one invoice foundation for sales, purchases and returns, so a new document type reuses invoice lines, attachments, payment cases and branch state.
- Designed for staff who enter invoices all day, with virtualized tables, validated forms, attachment upload from a phone by QR code, and full right-to-left layout.
- Built on Next.js 16, React 19 with React Compiler and TanStack Query, Table and Virtual. Stimulsoft PDF and Excel reports load lazily, and the file proxy is hardened against SSRF.

Freelance Full-Stack TypeScript Developer, Upwork

Aug 2022 to Present

- Top Rated, 100% Job Success and a 5.0 rating across 11 jobs and 1.3K hours.
- Monorepo product development with TypeScript, React, Node.js and MongoDB (2022 to 2023), a chart of accounts frontend in React (2023), a store management web app (2023), and React components built from XD designs with Storybook (2022).