/goal
Onderzoek of en hoe Tado Studio verantwoord kan worden herbouwd als zelfstandige implementatie, zonder Stacki-code over te nemen. Lever een onderbouwd advies en eventueel een losstaand klikbaar prototype. Stop voor Codex-review en akkoord van Fahd. Start geen productontwikkeling.

Context
Tado is een lokale Astro-builder met visuele bewerking, HTML/CSS/JS-editors, geïntegreerde Codex-chat, skills, koppelingen, CMS/collections, Core Framework en Git. Het product begon als Stacki-fork. Fahd wil een eigen premiumproduct en onderzoekt of een zelfstandige basis de investering waard is.
Productrepo: /Users/fahd/Developer/tado-studio
Projectcontext: /Users/fahd/Mijn Drive/1- codex - werkmap/projecten/tado-studio
Lees toepasselijke AGENTS.md-bestanden, profiel, relevante dossiers, implementatietaak en v2-ontwerpbesluiten. Controleer actuele code en licenties. Neem eerdere kostenramingen niet als vaststaand aan.

Harde grenzen

- De productrepo is uitsluitend leesbaar.
- Wijzig geen geïnstalleerde app, gebruikersprofielen of websiteprojecten. Geen push, merge, deployment, GitHub Actions of betaalde AI-tests.
- Bewaar onderzoek en prototype in een nieuwe map onder het projectpad/artefacten/. Volg voor documentatie en taakregistratie de vaultafspraken. Behoud bestaand werk.
- Een prototype is volledig losstaand, met fictieve gegevens en gesimuleerde interacties. Geen verbinding met Tado, echte projecten of AI-diensten. Benoem simulaties. Kopieer geen Stacki-implementatie naar het prototype.

Onderzoek

1. Feitelijke uitgangssituatie
   Inventariseer huidige functies, architectuur en afhankelijkheden. Welke delen zijn ongewijzigd overgenomen, aangepast of nieuw? Controleer herkomst; een gewijzigd bestand is niet automatisch zelfstandig werk. Scheid Stacki, eigen Tado-bijdragen en externe libraries/assets. Gebruik bestandsaantallen niet als maat voor productwaarde of juridische onafhankelijkheid.
2. Licentie en zelfstandigheid
   Onderzoek op basis van primaire bronnen wat MIT toestaat en welke vermeldingen vereist blijven. Onderscheid eigen branding, marketing, distributielicenties en historische herkomst. Onderzoek ook afzonderlijke commerciële toestemming als alternatief, zonder iemand te benaderen.
   Beschrijf een aantoonbaar zelfstandige ontwikkelaanpak op basis van functionele specificaties en eigen tests. Hernoemen, refactoring of AI-parafraseren van bestaande code geldt niet als bewijs. Benoem onzekerheden en punten voor juridische toetsing. Beloof geen onherleidbaarheid of nul verplichtingen voor externe libraries.
3. Vergelijk drie routes
   A. Bestaande fork professioneel doorontwikkelen.
   B. Onderdelen geleidelijk zelfstandig vervangen.
   C. Een nieuwe codebase met eigen architectuur en UI/UX.
   Vergelijk klantwaarde, technische winst, risico, onderhoud, migratie, vertraging van

marktvalidatie en kosten. Geef een eerlijke aanbeveling; een herbouw is geen vooraf bepaalde uitkomst.

4. Technisch plan en raming
   Werk bij een kansrijke herbouw modules, afhankelijkheden en acceptatiecriteria uit. Besteed aandacht aan Astro-bronbehoud, DOM/selectiekoppeling, CSS-bronnen, gelijktijdige AI-/gebruikerswijzigingen, autosave, Undo/Redo, conflictherstel en macOS/Windows.
   Scheid demonstratie, bruikbare bèta en commercieel betrouwbare release. Raam per onderdeel inspanning met bandbreedtes en aannames. Onderscheid ingehuurde ontwikkelkosten, directe AI/toolkosten en Fahds test-/beslistijd. Benoem wat onbekend is. Stel een kleine technische proef voor om de raming te toetsen; voer die niet uit.

Oplevering
Lever een beslisnotitie met bronnen, vergelijking, aanbeveling, kosten, risico's, fasering en go/no-go-criteria. Een prototype is optioneel en alleen zinvol als het een ontwerpbeslissing helpt beoordelen; het bewijst geen technische haalbaarheid. Vat samen welke keuze Fahd moet maken. Stop na oplevering.

Werk onafhankelijk. Lees het onderzoek van de andere agent niet. Bewaar jouw onderzoek in een afzonderlijke artefactmap met jouw agentnaam.