REFACTOR — SAMPLE AUDIT

Production-readiness review of a vibe-coded SaaS MVP

Client: Anonymized • Stack: React + TanStack + Supabase • LOC: 14,200 • Duration: 5 business days

This is a redacted, representative sample of what every Refactor audit looks like. Real audits are 18–25 pages, repo-tagged, and ranked by cost-to-fix.

Executive summary

The app ships and demos well, but ~38% of the codebase carries production risk. We found 3 critical security issues (RLS bypass, exposed service role, missing rate limiting on auth) and 11 high-impact maintainability issues. None require a rewrite. Estimated refactor time to clear all P0/P1 findings: **2.5 weeks**, $16,800.

Findings by severity

Severity	Count	Examples
P0 — Critical (security)	3	RLS disabled on user_data, service_role in client bundle
P1 — High (correctness)	11	Stale closure in useEffect, race in checkout, n+1 in dashboard
P2 — Medium (maintain.)	27	1,400-line component, any[] in 64 places, no error boundaries
P3 — Low (style/perf)	48	Bundle size, missing memoization, console.log left in

Sample finding: P0 — RLS disabled on user_data

Risk: Any authenticated user can read every other user's billing rows by calling the Data API directly with their own token. We verified this by signing up a second test account and selecting all rows.

Location: supabase/migrations/20240612_init.sql, line 14

Fix (estimated 25 minutes):

```
ALTER TABLE public.user_data ENABLE ROW LEVEL SECURITY;
CREATE POLICY "own rows" ON public.user_data
  FOR SELECT TO authenticated USING (user_id = auth.uid());
```

Sample finding: P1 — Stale closure in useDashboard

Risk: Dashboard refresh interval captures the first render's filter state, so user-applied filters silently revert every 60s. Reported by 2 users in support already.

Fix: Replace bespoke polling with TanStack Query `refetchInterval` tied to the same query key. ~40 LOC removed, no behavior change for the user beyond filters working.

Architecture map

Current: 1 monolithic React app, no route-level code splitting, all data fetched in a single useEffect at App.tsx mount. Bundle 2.4 MB, LCP 5.8s on 4G.

Proposed: TanStack Router loaders + suspense queries per route. No state libraries needed. Estimated bundle after split: 410 KB initial, LCP <1.5s.

Recommended refactor plan

Week	Focus	Deliverable
1	P0 security fixes + CI	All RLS, secrets out of client, GitHub Actions running on PR
2	Route + data layer	TanStack Router migration, queries, Suspense boundaries
2.5	Test suite + hand-off	Vitest + Playwright smoke, runbook, walkthrough

What you get on a real engagement

- 18–25 page PDF + Notion-mirrored version
- Every finding tagged in your repo as a GitHub issue with severity label
- Loom walkthrough of the 5 most important findings
- Live 60-minute Q&A call with the engineer who did the audit
- Fixed price, signed NDA before any code is shared