R RepoRisk Receipt

AI-BUILT SAAS LAUNCH CHECKLIST

Before users, payments or private data, check the risk paths.

A practical pre-launch checklist for founders and small teams shipping Next.js, Stripe, Supabase, Clerk and Vercel apps with AI coding tools.

SELF-CHECK TARGET

72 / 100

7	35	24-48h
AREAS	CHECKS	RECEIPT

01 / PAYMENT RISK

High

Stripe webhook checks

Your paid state should come from verified server events, not from a success page or client redirect.

- [ ] Verify Stripe-Signature with the endpoint secret.
- [ ] Use the raw request body before parsing JSON.
- [ ] Make webhook handlers idempotent by event id or subscription id.
- [ ] Update subscription state only after trusted webhook events.
- [ ] Log failed verification without mutating payment state.

02 / DATA ACCESS

High

Supabase RLS checks

A working demo can still expose customer data if tables are public or policies are missing.

- [ ] Enable RLS on customer, workspace, billing and admin-facing tables.
- [ ] Add owner or membership-based select, insert, update and delete policies.
- [ ] Keep service role keys server-only and out of client bundles.
- [ ] Test that user A cannot read or mutate user B data.
- [ ] Avoid public table access for private SaaS data.

03 / AUTHORIZATION

High

API route auth checks

Hiding UI is not authorization. Every server mutation needs its own session and role check.

- [ ] Validate the session inside every API route and server action.
- [ ] Check account, workspace and role before writing data.
- [ ] Do not trust userId, customerId or role values from the client.
- [ ] Return consistent 401 and 403 responses for blocked paths.
- [ ] Add tests for unauthenticated and wrong-workspace requests.

04 / SECRET EXPOSURE

Medium

Environment variable exposure checks

AI-generated code often moves secrets across client and server boundaries without noticing.

- [ ] Scan for hardcoded API keys, tokens and provider secrets.
- [ ] Review every NEXT_PUBLIC_ variable for intentional client exposure.
- [ ] Keep Stripe, Supabase service role and AI provider keys server-only.
- [ ] Remove committed .env files and rotate exposed credentials.
- [ ] Check built client assets for leaked keys before launch.

05 / ADMIN ACCESS

High

Admin route checks

Admin screens are high-impact paths. They need server-side role checks, not just hidden navigation.

- [ ] Protect admin pages with server-side auth and role validation.
- [ ] Protect admin API routes separately from the UI.
- [ ] Avoid trusting route groups or middleware as the only guard.
- [ ] Record who changed critical customer or billing data.
- [ ] Verify direct URL access to admin routes is blocked.

06 / DATA INTEGRITY

Medium

Database unique constraint checks

Billing and identity bugs become expensive when duplicate customer, subscription or workspace rows appear.

- [ ] Add unique constraints for provider customer ids and subscription ids.
- [ ] Add unique constraints for workspace slugs, invite tokens and external ids.
- [ ] Use transactions for multi-step billing and onboarding writes.
- [ ] Handle duplicate webhook events without duplicate records.
- [ ] Backfill missing constraints before production data grows.

07 / REGRESSION RISK

Medium

Critical tests checks

You do not need a huge test suite, but payment, auth and data writes need coverage before real users arrive.

- [ ] Test payment webhook success, failure and replay behavior.
- [ ] Test auth-required API routes with no session and wrong role.
- [ ] Test subscription gating and plan changes.
- [ ] Test database writes for ownership and duplicate handling.
- [ ] Add at least one end-to-end happy path for signup to paid access.

Free checklist outcome

If any high-risk area is unchecked, fix it before connecting live payments, inviting users or writing production customer data.

Need file paths and repair prompts?

Submit your GitHub repo or zip. RepoRisk returns a PDF / Markdown launch risk receipt within 24-48 hours.