

Vibe coding security checklist

Ten checks to run before you launch an app built with Lovable, Bolt, Cursor or similar on Supabase. Every item comes from [our test of six scanners](#) on an app with 10 planted holes.

● Based on tests run Oct 16, 2026 Free, no signup needed

Why a checklist: the best single scanner caught 4.5 of our 10 holes, and all six together caught 7.5. A clean scan means the tool found nothing with the checks it ran, not that your app is safe.

Run two scans

About 15 minutes. They look at different layers, so run both.

- ☐ **1. Run Supabase's built-in Security Advisor and fix every error.**
Supabase dashboard → Advisors → Security Advisor. Each finding links to a fix page with SQL you can copy. Rescan after fixing.
In our test it was the only tool that found database holes (3 of 10): a table with RLS off, a view exposing user emails, and a function anyone could call. Our pick, 76/100.
- ☐ **2. Scan your code, secrets and packages.**
Connect your repo to a free code scanner such as Aikido's free plan, or run Semgrep Community Edition on your own machine if you'd rather not share your code.
Aikido's free plan caught the most of any tool: 4.5 of 10 (leaked keys, the XSS and the vulnerable package). Runner-up, 66/100. Semgrep CE caught both leaked keys.

Check by hand what the scanners missed

No scanner we tested caught these reliably. Each takes a few minutes.

☐ 3. Policies check the owner, not just “signed in”.

Policies should compare `auth.uid()` to the row’s owner column. A policy that says using (true), or only checks the authenticated role, lets every user see everyone’s rows.

Test Create a second test user, sign in as them, and try to read or edit the first user’s rows. It should fail.

All six tools missed this one.

☐ 4. Every Edge Function and API route checks who’s calling.

Keep “Verify JWT” on. Inside the function, use the caller’s token rather than the service-role key, so RLS still applies.

Test Call each function with no login (open its URL, or curl with no Authorization header). It should refuse.

All six tools missed this one. One even reported that API routes required auth when ours didn’t.

☐ 5. Storage buckets are private, and never listable or uploadable by anonymous users.

Make buckets private unless their files are truly public. Remove select or insert policies on `storage.objects` that are open to anon.

Test While signed out, try to list the bucket and upload a file. Both should fail.

Only one tool half-flagged this, with the wrong fix.

☐ 6. No user text is rendered as raw HTML.

Search your code for `dangerouslySetInnerHTML`, `innerHTML` and `v-html`. Anything a user typed should be rendered as text.

Test Save a note containing `<img src=x onerror=alert(1)>`. No alert should pop up.

Only one tool of six caught this.

☐ 7. No known-vulnerable packages.

Run `npm audit` and update what it flags. Go by `npm audit` rather than a scanner’s suggested version.

Only one tool of six flagged our outdated package, and its suggested version still had advisories.

Confirm the basics the scanners did catch

Quick to confirm, painful to miss.

☐ 8. RLS is on for every table in the public schema.

With RLS off, anyone with your project URL and public key can read, edit and delete every row, per Supabase's docs.

We confirmed it on our test app: an anonymous request read and added notes.

☐ 9. No secret or service_role key in front-end code or a committed .env.

The anon or publishable key is meant to be public. The service_role or secret key bypasses RLS and must never reach the browser. If one ever shipped, rotate it: deleting the line isn't enough, it stays in git history.

Three tools found our planted service-role key; GitHub's push protection blocked the committed .env.

☐ 10. No view or function exposes auth.users.

Views and SECURITY DEFINER functions in the public schema can leak user emails or let anyone act on other users' data.

Security Advisor flagged both in our test.

After the checklist

- Read the full verdict, with every tool's score and the test log: [security check for your vibe-coded app](#).
- Going live on your own domain? [Cloudflare Pages vs Vercel](#).
- Need login? [Clerk vs Supabase Auth vs Firebase Auth](#).

This checklist is general guidance from one small test app, not a security audit. Real apps have more tables, functions and code paths. Free to share and print; please link back.