Is your AI-built app safe to launch?

The 10 checks we run first on any app built with Lovable, Bolt or another AI builder on Supabase. You can do all of them yourself in about an hour. No tools to buy.

Time: about 1 hour **You need:** your Supabase dashboard and your code **Only test** an app you own

- [ ] 01 **Turn on Row Level Security (RLS) for every table**

  Your app's public key is visible to every visitor, so RLS is what actually protects your data. Supabase's own docs say to enable RLS on every table in an exposed schema.

  Where: Supabase → Advisors → Security Advisor. Fix every "RLS disabled" warning.

- [ ] 02 **Look for `using (true)` policies**

  A policy whose condition is just `true` on SELECT, UPDATE or DELETE means "anyone". Fine for public blog posts, bad for user data.

  ```
  select tablename, policyname, cmd, qual from pg_policies where schemaname = 'public';
  ```

  Where: Supabase → SQL editor. Check every row where `qual` is `true`.

- [ ] 03 **Test with two accounts**

  Sign up as user A and create something. Log in as user B in another browser and try to see A's data. If you can, your policies need a check like `auth.uid() = user_id`.

- [ ] 04 **Search your code for secret keys**

  Anything in `src/` is public. Search for `service_role`, `sk_live`, `sk-` and `whsec_`. If you find one, move it to an Edge Function secret **and rotate the key**: once it has been public, treat it as leaked.

- [ ] 05 **Check your `VITE_` variables**

  Every `VITE_` value ends up in the browser. Only the Supabase URL and the publishable (anon) key belong there.

- [ ] 06 **Make storage buckets private**

  Is any bucket marked "Public" while holding private files, such as IDs or invoices? Make it private and use signed URLs.

  Where: Supabase → Storage.

- [ ] 07 **Check your Edge Functions identify the user**

  If a function does anything user-specific, it should call `supabase.auth.getUser()` first. `verify_jwt` on its own also accepts the public key.

- [ ] 08 **If you take payments: verify the Stripe webhook signature**

  Your webhook should use `stripe.webhooks.constructEventAsync(...)`. Prices should come from Price IDs on the server, never from the browser.

- [ ] 09 **Protect admin pages on the server**

  Hiding the admin link isn't protection. The data the admin page uses must be protected by RLS or by a function that checks the user's role.

- [ ] 10 **Turn on email confirmation and decent password rules**

  Minimum 8 characters (12 is better), plus leaked-password protection if your plan has it.

  Where: Supabase → Authentication → Providers / Policies.

10 of 10 ticked
A strong start. Re-check after every big change: any change can reopen an old gap.

7 to 9
Fix the gaps before launch, starting with 1–4: they protect your users' data.

6 or fewer
Pause new features for a day and work through the list, or ask for help.

Also: Lovable has a built-in security scan, and it's worth running. Its docs say it can't guarantee complete security, which is true of any tool, so the manual checks above still matter.

Prefer a second pair of eyes?

Our **free mini scan** runs five of the most important checks on your app and sends you a plain-English summary within two business days of your signed permission.

- Read-only: we never change your code, data or settings.
- Nothing is tested until you've signed a short authorization form.
- Works for Lovable, Bolt, Replit, v0, Base44 and other AI builders.

Request your free mini scan → axieo.ca/services/axieo-guard

General guidance, not a full security audit. Only test systems you own or are authorized to test. © 2026 Axieo Technologies Inc.