HOSTINGER

SECURITY CHECKLIST

Security checklist for your vibe-coded app

Hostinger Learning lab

CONTENTS

What's inside

The vibe coding app security checklist

Just because your AI-built app works doesn't mean it's secure. Go through these 5 checks before you launch to close the most common ways vibe-coded apps get broken into.

How to use this checklist: Work through each section below and check off every item as you confirm it. If something isn't checked, that's your next fix before you go live.

API keys are never exposed

- [ ] **No keys in frontend code**
  Search your project for `sk-`, `api_key`, `Bearer`, and `service_role`. None of these should appear in frontend code.
- [ ] **No keys in Git history**
  Check whether a `.env` file was ever committed. Deleting it later doesn't remove it from your history.
- [ ] **External calls go through a backend**
  Your frontend should talk to your backend, and your backend should talk to the external service. Not the other way around.
- [ ] **Exposed keys are invalidated**
  If a key was ever exposed, generate a new one in the provider's dashboard. Removing it from your code isn't enough.

Using Hostinger AI Builder? Give your API key in the prompt and the Builder will generate the backend integration for you safely.

Authentication and authorization are in place

- [ ] **Two-account test passes**
  Log in as user one, copy a resource URL, then log in as user two and try to open it. It should fail.
- [ ] **Using an established auth provider**
  Supabase Auth, Clerk, Auth0, or Google sign-in, rather than a custom-built login system.

Using Hostinger AI Builder? Authentication runs through its integrated backend, and you can prompt it to set up protected routes for logged-in users only.

AI endpoints are rate limited

- [ ] **Requests per user or IP are limited**
  Your backend tracks how many calls a user or IP has made and blocks them once they go over the limit.
- [ ] **Spending limits are set**
  Set a monthly spending limit and billing alerts with your AI provider as a backup safety net.

In the Hostinger AI Builder, the key management for its built-in AI features is already handled. This mainly matters if you've connected an external AI service yourself.

Database permissions are locked down

- [ ] **Second-account test passes**
  Create a second user account and try to access the first account's data. It should be blocked.
- [ ] **Row or record-level security is on**
  If you're on Supabase, Firebase, or your own database, every query should be scoped to the logged-in user, not left on default open access.

If you are using Hostinger AI Builder, database security is managed for you, but the app logic is still on you. Prompt it directly – for example "make sure users can only see and edit their own records in the contacts collection" – and test it in the Data tab before switching to Live.

User input is never trusted

- [ ] **Frontend validation is in place**
  Forms catch missing fields, invalid formats, and short passwords, for instant feedback. Remember this layer can be bypassed.
- [ ] **Backend re-validates everything**
  Required fields, data types, and allowed values are checked again on the server before anything is saved.
- [ ] **User-generated content is sanitized**
  Anything displayed back to users is run through a sanitizer, like DOMPurify, before it's rendered as HTML.
- [ ] **File uploads are controlled**
  Only safe file types are allowed, size limits are set, and uploaded files are never assumed to be safe by default.