BUILD SAFELY · BUILDER · 14 MIN READ

Build safely: secrets, data and honest AI use

The habits that keep your users, your accounts and your reputation safe when you build with AI.

Pass the check (4 of 5) online to earn the “Safe Builder” badge:
https://digitalbuilders757.com/university/study/build-safely

Secrets stay secret

A secret is anything that grants access: passwords, API keys, database service keys, payment keys. If a secret is in your front-end code, anyone can open the browser's developer tools and copy it. Researchers have found exactly this in many vibe-coded apps, including AI and payment keys sitting in plain sight.

- Keep secrets in environment variables on the server or host, in a .env file locally, and list .env in .gitignore.
- Call paid APIs (AI, payments, email) from server code or an edge function, never straight from the browser.
- If a secret leaks (pushed to GitHub, pasted in a chat), rotate it: create a new one and revoke the old one. Deleting the commit isn't enough.
- Turn on two-factor authentication for GitHub, your host, your database and your email.

Lock the data by default

Assume anyone can call your API directly. The database or server must decide who sees what: row level security on every table, deny by default, smallest possible policies. Test it by trying to read another user's data while signed in as someone else.

Collect only the data you need, and be extra careful with children's and students' data, health information and anything financial.

Check what AI writes

- Ask the AI to review its own code for security issues, then ask a second tool too.
- Read every change that touches sign-in, payments, permissions or deleting data.
- Don't install packages you can't find on the official registry, and check they're real and maintained. AI can invent package names.
- Back up your database before big changes; keep code in Git.

Honest and fair AI use

Follow your school's or employer's AI rules and say when AI helped. Don't pass off AI writing or art as entirely yours where that matters.

Respect copyright: don't ask AI to reproduce other people's books, songs, logos or characters, and check you have rights to images you use.

AI can reflect bias in its training data. Look at who an AI-powered feature might treat unfairly, and test with a range of people.

Make it accessible: real text on buttons and images, good colour contrast, keyboard navigation. AI tools can check this if you ask.

Key terms

Secret
Anything that grants access: passwords, API keys, service keys.

.env file
A local file holding environment variables and secrets; never committed to Git.

Rotate (a key)
Replace a secret with a new one and revoke the old one.

Row level security (RLS)
Database rules that decide which rows each user may read or change.

Supply-chain risk
Danger from the packages and tools your project depends on, including fake look-alike packages.

Bias (in AI)
Unfair patterns an AI picks up from its training data.

API key
A secret code that identifies your app to a service and is often tied to billing.

.gitignore
A list of files Git must never save, such as .env secrets.

Two-factor authentication (2FA)
A second check at sign-in, like a code from an app, on top of your password.

Least privilege
Give each person, app or AI only the access its job needs.

Accessibility (a11y)
Making apps usable by everyone, including people using screen readers or keyboards.

Try it: Run a 10-minute safety check

1. In your project, search the code for 'key', 'secret' and 'password'. Anything that looks real?
2. Confirm .env is listed in .gitignore.
3. Ask your AI tool: 'Review this project for exposed secrets, missing access checks and unsafe database rules. List findings by severity.'
4. Fix the top finding and turn on two-factor for your GitHub account.

KNOWLEDGE CHECK

Build safely: secrets, data and honest AI use

5 questions. Circle one answer for each; 4 or more correct passes. Answers are at the back.

1. You find your AI API key in your front-end JavaScript. What's the problem?

A. None, because browsers hide JavaScript from people visiting your site
B. It makes the page slower, because the key is checked on every click
C. Anyone can copy it from the browser and misuse it
D. It breaks the colours

2. You accidentally pushed a secret to GitHub. Best fix?

A. Delete the commit and move on
B. Make the repo private, delete the commit, and keep using the same key
C. Rename the file
D. Rotate the secret: make a new one and revoke the old one

3. What does 'deny by default' mean for database access?

A. Nothing is readable until a rule explicitly allows it
B. Nobody can sign up
C. The database is turned off
D. Only admins can see the app

4. An AI suggests installing a package you've never heard of. What should you do first?

A. Install it immediately
B. Check it exists on the official registry and is real and maintained
C. Install three similar ones
D. Ask the AI if it's safe, and install it if the AI says it's very popular

5. Which habit makes your app more accessible?

A. Using images of text, so the fonts look the same on every device
B. Small grey text, which looks cleaner and more modern
C. Real text labels, contrast and keyboard access
D. Auto-playing a welcome message so visitors know where they are

Official sites

OWASP Top 10: https://owasp.org/www-project-top-ten/
GitHub Docs: Secret scanning: https://docs.github.com/en/code-security/secret-scanning/introduction/about-secret-scanning
W3C: Introduction to web accessibility: https://www.w3.org/WAI/fundamentals/accessibility-intro/

Last reviewed 2026-11-25. AI tools change fast; check each tool's own site for current features, plans and prices.

ANSWER KEY

Build safely: secrets, data and honest AI use

1. C: Anyone can copy it from the browser and misuse it
Front-end code is public. Move the call to server code and rotate the key.

2. D: Rotate the secret: make a new one and revoke the old one
Once exposed, assume it's copied. Rotating is the only real fix.

3. A: Nothing is readable until a rule explicitly allows it
Start locked, then open only what each user needs.

4. B: Check it exists on the official registry and is real and maintained
AI can invent package names, and attackers publish look-alikes. Verify first.

5. C: Real text labels, contrast and keyboard access
Accessible apps work for more people, including people using screen readers or keyboards.