AI STUDIO · CHAPTER 6

~$ open code-your-own-tools

Code your own tools

You don’t need to read code to build a tool with an AI. You need a short brief, one small step at a time, your own eyes on every result, and a journal. Here’s the method, then one of my tools as it really is.

ENTRY A6 CATEGORY AI STUDIO CHECKED 7 NOVEMBER 2026 READING 6 MIN
SECTIONS 7



ls cover/ A6.webp

EVIDENCE ◆ Verified ◆ Vivian’s field notes

IN 30 SECONDS

01 Start with a short brief: what the tool does, what it never does, and how you’ll know it works.

02 Ask for one small step, run it yourself, look at it, then say what’s wrong in plain words.

03 Keep a journal, so a new conversation picks up where the last one stopped.

04 Keys, passwords and card numbers never go into the chat or into the code.

05 Anything that moves money gets tested with fake money first, and it needs a limit, a total and a stop.

YOU WILL LEARN

- ✓ How to write a brief an AI can build from.
- ✓ The loop of asking, running, looking and correcting.
- ✓ What a project journal holds and why it matters.
- ✓ How Vivian's Autodrain works, and where it falls short of this wiki's own rules.
- ✓ What a safe autodrain looks like, in a simulator with fake money.

> *I'm not a developer. I build my tools anyway, one small step at a time, and I look at every result before I ask for the next one.*
>
> — VIVIAN

~/01-start-with-a-short-brief

Start with a short brief

An AI builds what you describe, so describe it before anything else. Half a page is enough. The brief is also what you'll compare every result against, so write it in your own words. ◆

A brief in five lines

a-brief-in-five-lines.md · 5 lines copy

```
1 What the tool does, in one sentence.
2 What it never does. For anything near money, say it never sends
  without a limit.
3 Where it runs: a web page, a browser extension or a small program
  on your computer.
4 What must never fail, such as the stop button.
5 How you'll know it's done: what you'll see on screen when
  it works.
```

Starting from a brief

Example conversation. Invented for this exercise.

you	Here's my brief. A page that shows three wishlist goals with a progress bar each. It never asks for money and never stores names. Done when I open the file and see three goals with bars. Before you write anything, ask me what's unclear.
them	Three questions. Do the goals come from a list you type or from your wishlist page? Should a bar count gifts or amounts? Dark theme, light theme, or both?
you	A list I type. Gifts, not amounts. Dark only.
them	Clear. Step one is the page with three made-up goals and no styling yet. Open it and tell me what you see.

Asking the AI to list what's unclear before it writes anything saves you a whole round of fixing the wrong thing. ◆ A monthly chat plan is enough for small tools like this one. See the Budget planner for what each plan costs.

~/02-one-small-step-at-a-time

One small step at a time

Big requests come back big and broken, and you can't tell which part broke. Small steps give you something you can check in a minute. ◆

The loop

ask → run → look → say → write ↺

Ask for one step

One screen, one button or one rule. Say which step of the brief it is. ◆

Run it yourself

Open the page or start the program on your own computer. A step you haven't run isn't done, whatever the chat says. ◆

Look at it

Compare what's on screen with the brief. Take a screenshot when something looks wrong. ◆

Say what's wrong

Describe what you see, not what you guess. "The bar stays empty after I tick a gift" helps more than "the code is broken". ◆

Write one line down

Add the date and what changed to your journal before the next step. ◆

Saying what's wrong

Example conversation. Invented for this exercise.

you	The bar stays empty when I tick a gift. Screenshot attached. Nothing else changed.
them	The tick isn't saved before the bar redraws. I'll change only that part. Reload and tick the second goal.
you	The second goal works now. The first one still stays empty.
them	The first goal has a different name in the list than on the page. I fixed the name and nothing else. Try once more.

Ask for the smallest change that fixes the problem, and say what must stay as it is. When a fix breaks something that worked before, go back to the last version that worked and try again from there. ◆

~/03-keep-a-journal

Keep a journal

Long conversations drift. The model loses track of early decisions, and a new conversation starts from nothing. A journal fixes both: paste it at the top of a new chat and the work picks up where it stopped, even weeks later. ◆

```
# Three files that keep a project alive
```

three-files-that-keep-a.md · 3 lines copy

```
1 A brief: what the tool does, what it never does and when
  it's done.
2 A plan: the steps in order, each ticked only after you've seen
  it work.
3 A journal: one entry per session, newest at the top, with what you
  decided, why, and what you turned down.
```

The "turned down" part matters more than it looks. Without it, the next conversation suggests the same rejected idea again. ◆

~/04-keep-secrets-out

Keep secrets out

Everything on the list in Pick your AI applies twice over when you code, because code gets copied, shared and uploaded. A key or a password pasted into the chat can end up written into the code. ◆

Keeping a key out

Example conversation. Invented for this exercise.

you	The tool needs my API key. Should I paste it here so you can put it in the code?
them	No. Keep it out of this chat and out of the code. I'll make the tool ask for the key when it starts, and you'll type it on your own computer.

Test with made-up names and fake money. A tool that moves money gets built and tested against a copy or a test mode first, and it pays nothing until its stop works. ◆ Safe design has the full routine, and Check a tool before you install it is the same list from the other side.

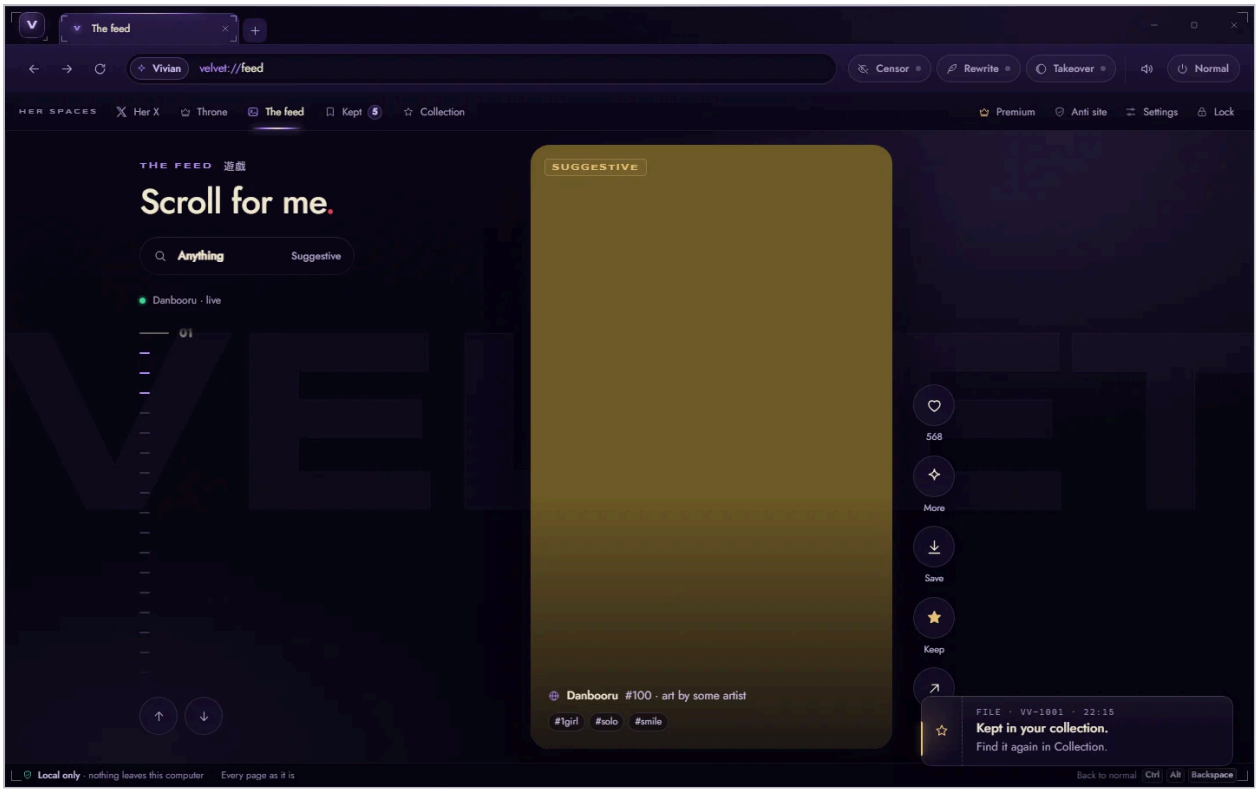
~/05-what-i-built-this-way

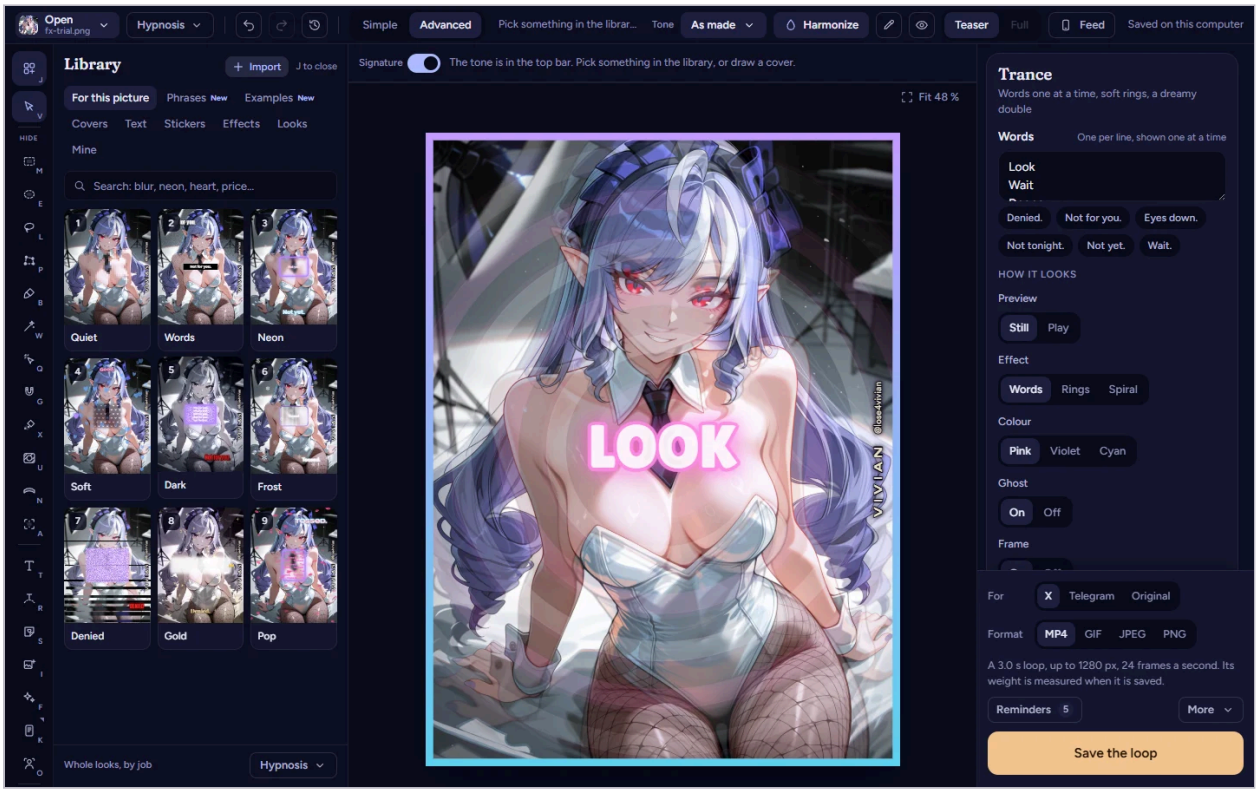
What I built this way

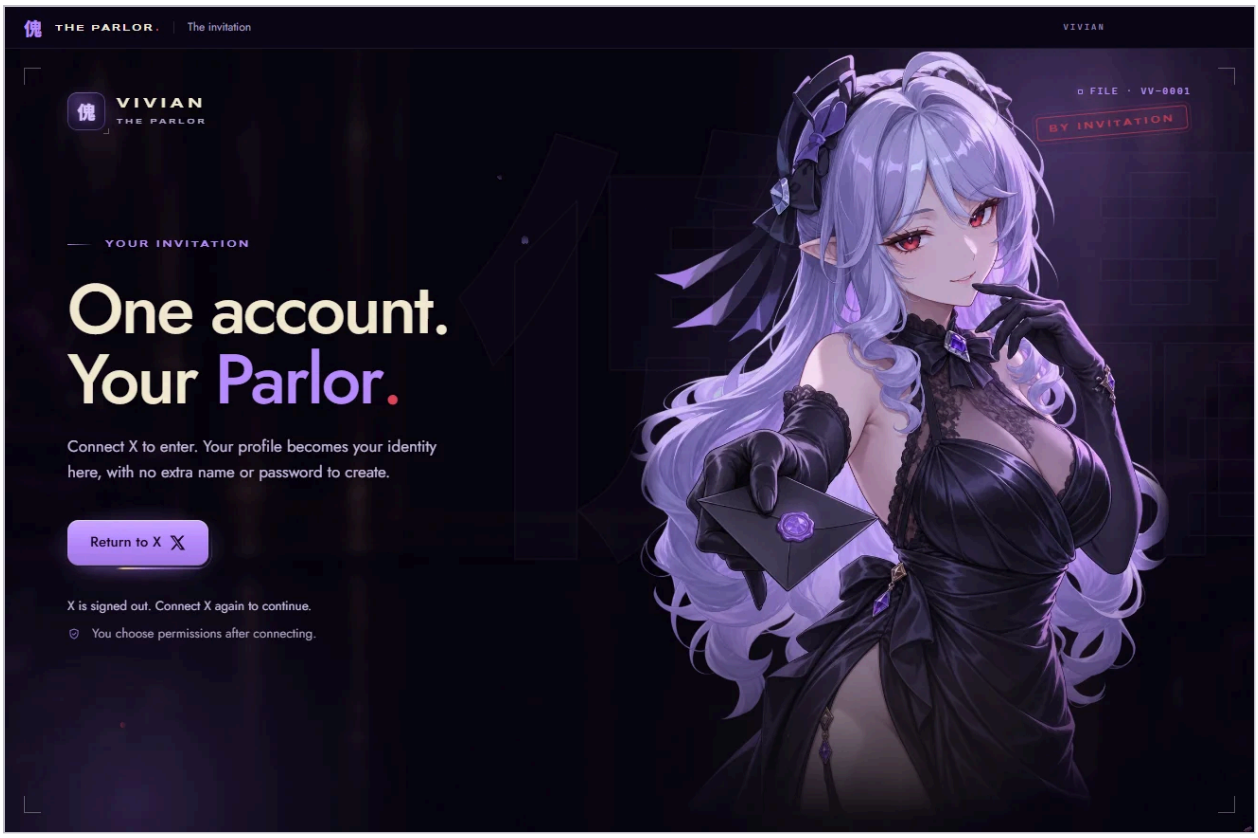
Everything below came out of that loop: a short brief, one small step at a time, a look at the result, then the next step. I didn't write the code by hand, and you don't need to read it to judge a tool. You judge it by what it does on screen. ◆

Three of my own apps

~$ ls three-of-my-own-apps/ 3 items







Velvet, the editor of my image tool and my Hub, The Parlor. All three were built with an AI, step by step, and none of them shows its code here. ◆

~/06-example-vivian-s-autodrain

Example: Vivian's Autodrain

Vivian's Autodrain is a Chrome extension I built with this method. It buys a gift from my Throne wishlist again and again, with the card of the sub who installs it. ◆

Its popup has a Drain button, a level bar and three modes. Dumb sends a USD 5 gift, Addicted a USD 10 gift and Surrender a USD 20 gift, and no price is shown on screen. The stronger the mode, the more its switch moves and glows. ◆

While it runs, my wishlist page plays effects: a spiral, images of me, short lines of text one at a time, faint clips with their sound and an ambient track. Every

USD 10 paid adds a level, and levels unlock image sets: Normal from the start, Censor at level 2, Armpits at level 6 and Feets at level 10. ◆

It loops through the checkout on its own: empty the cart, add the gift, check out, pick the card, press Pay Now, start again. It never types a card. If no card is ready after 10 seconds, it pauses and waits for the sub to press Pay Now by hand. A payment with an unknown result, or a bank check, stops everything and says why. ◆

There's no server and no account. Everything stays in the sub's browser. The effects follow the reduced-motion setting and never flash more than three times a second. Surrender takes two clicks and locks the modes. Only removing the extension stops it. ◆

It started from a one-page brief and a plan of eight steps. The first step was reading the real checkout page without paying anything, and every payment test ran on a copy of that page. ◆

Measured against this wiki's own Safe design rules, it falls short in three places. There's no running total on screen, only levels. There's no limit the sub sets before starting. And Surrender has no stop: only removing the extension ends it. That's how it is today, and I'm telling you so you can judge it the way you'd judge anyone else's tool. ◆

A platform's rules apply to any tool you point at it. Throne's help pages say the wishlist can't be used to exchange goods or services. ◆ They also say a profile can't offer content or services in exchange for gifts. ◆ Read the rules of the platform before you build for it. ◆

Try a safe autodrain

Here's the same idea built the other way round. It runs on fake money, asks for your limit first, shows the total all the time, and stops at the limit by itself. ◆

SIMULATOR, FAKE MONEY

A safe autodrain, with fake money

The gift it sends

Coffee	Lunch	Dinner
USD 5.00	USD 10.00	USD 20.00

Your limit, in US dollars

50

Set it before you start. While it runs you can lower it, never raise it.

SPENT

USD 0.00

LIMIT

USD 50.00

FAKE WALLET

USD 200.00

Not running.

Try the stop in every state: before you start, while it runs, while it's paused and after it ends. Then lower the limit while it runs and watch it stop. ◆

Esc stops it too. If you leave the page, it pauses. Nothing here is bought, saved or sent.

Count what happened. With a USD 50 limit and the USD 20 gift, it sends two gifts and stops at USD 40, because a third would pass the limit. A real

tool would also take the total from the payment provider, which a page can't do. ◆

TRY IT

It works, says the chat

An AI writes a whole autodrain in one answer and says it's ready. You haven't run it yet.

A Install it and try it once with your own card

Not this one. One real payment is a test you can't take back. And you haven't seen the stop work yet.

B Run it on fake money or a copy of the page, and check the limit, the total and the stop one by one

The right answer. Right. "It's ready" in a chat proves nothing. What you've seen working does.

C Ask the AI to double-check its own code and install it if it says yes

Not this one. A second look from the same chat isn't a test. Run it and look.

REMEMBER IT

- Write a short brief first, in your own words.
- One step, run it, look, say what's wrong.
- Keep a journal with what you decided and what you turned down.
- No keys or card numbers in the chat or the code.
- Anything with money gets fake money first, then a limit, a total and a stop.

SOURCES AND EVIDENCE

Check it

Every claim with its evidence level and its sources.

EVIDENCE LEVELS

◆ Verified	Checked against the original source.
◆ Reported	Stated by a serious secondary source read in full.
◆ Community	Widely said in the scene; no document proves it.
◆ Vivian's field notes	First-hand experience, not a study.
◆ Unverified	Plausible, not proven yet.

01 Half a page is enough.

◆

02 The brief is also what you'll compare every result against, so write it in your own words.

◆

03 Asking the AI to list what's unclear before it writes anything saves you a whole round of fixing the wrong thing.

◆

04 Big requests come back big and broken, and you can't tell which part broke.

◆

05 Small steps give you something you can check in a minute.

◆

06 Say which step of the brief it is.

◆

07 A step you haven't run isn't done, whatever the chat says.

◆

08 Take a screenshot when something looks wrong.

◆

09 "The bar stays empty after I tick a gift" helps more than "the code is broken".

◆

10 Add the date and what changed to your journal before the next step.

◆

11 Ask for the smallest change that fixes the problem, and say what must stay as it is.

◆

12 When a fix breaks something that worked before, go back to the last version that worked and try again from there.

◆

13 The model loses track of early decisions, and a new conversation starts from nothing.

◆

14 A journal fixes both: paste it at the top of a new chat and the work picks up where it stopped, even weeks later.

◆

15 Without it, the next conversation suggests the same rejected idea again.

◆

16 Everything on the list in Pick your AI applies twice over when you code, because code gets copied, shared and uploaded.

◆

17 A key or a password pasted into the chat can end up written into the code.

◆

18 A tool that moves money gets built and tested against a copy or a test mode first, and it pays nothing until its stop works.

◆

19 Everything below came out of that loop: a short brief, one small step at a time, a look at the result, then the next step.

◆

20 You judge it by what it does on screen.

◆

21 All three were built with an AI, step by step, and none of them shows its code here.

◆

22 It buys a gift from my Throne wishlist again and again, with the card of the sub who installs it.

◆

23 Dumb sends a USD 5 gift, Addicted a USD 10 gift and Surrender a USD 20 gift, and no price is shown on screen.

◆

24 The stronger the mode, the more its switch moves and glows.

◆

25 While it runs, my wishlist page plays effects: a spiral, images of me, short lines of text one at a time, faint clips with their sound and an ambient track.

◆

26 Every USD 10 paid adds a level, and levels unlock image sets: Normal from the start, Censor at level 2, Armpits at level 6 and Feets at level 10.

◆

27 It loops through the checkout on its own: empty the cart, add the gift, check out, pick the card, press Pay Now, start again.

◆

28 If no card is ready after 10 seconds, it pauses and waits for the sub to press Pay Now by hand.

◆

29 A payment with an unknown result, or a bank check, stops everything and says why.

◆

30 Everything stays in the sub's browser.

◆

31 The effects follow the reduced-motion setting and never flash more than three times a second.

◆

32 Only removing the extension stops it.

◆

33 The first step was reading the real checkout page without paying anything, and every payment test ran on a copy of that page.

◆

34 And Surrender has no stop: only removing the extension ends it.

◆

35 That's how it is today, and I'm telling you so you can judge it the way you'd judge anyone else's tool.

◆

36 Throne's help pages say the wishlist can't be used to exchange goods or services.

◆

What types of items can I add to my wishlist? (https://help.throne.com/en/articles/15808792-what-types-of-items-can-i-add-to-my-wishlist)

Archive (https://web.archive.org/web/20261019011312/https://help.throne.com/en/articles/15808792-what-types-of-items-can-i-add-to-my-wishlist)

Accessed 3 November 2026

37 They also say a profile can't offer content or services in exchange for gifts.

◆

What's allowed in my profile text and images? (https://help.throne.com/en/articles/15810290-what-s-allowed-in-my-profile-text-and-images)

Archive (https://web.archive.org/web/20261026034619/https://help.throne.com/en/articles/15810290-what-s-allowed-in-my-profile-text-and-images)

Accessed 3 November 2026

38 Read the rules of the platform before you build for it.

◆

39 It runs on fake money, asks for your limit first, shows the total all the time, and stops at the limit by itself.

◆

40 Then lower the limit while it runs and watch it stop.

◆

41 With a USD 50 limit and the USD 20 gift, it sends two gifts and stops at USD 40, because a third would pass the limit.

◆

42 A real tool would also take the total from the payment provider, which a page can’t do.

◆

Nothing here is financial, legal or medical advice for your situation.