Build with a real toolkit.

Why serious vibe coding starts with VS Code on your own machine — and the freedom it gives you.

01 — LOCAL FIRST

Why work with VS Code on your own computer?

Many platforms let you write code right in the browser, like Replit. But local software like VS Code is the professional standard — for three practical reasons.

01

Ownership

Your code doesn't live in a third party's cloud that can shut down or raise prices. The files are your digital asset, saved on your own disk, fully under your control.

02

Speed & privacy

The software runs locally, so typing and searching never wait on your internet. Sensitive code and confidential ideas stay on your machine.

03

Version control

Git is like “Google Docs for code,” only smarter — a full history of every change. Broke something? One click rolls back to yesterday's working version, and teammates can build in parallel.

• 02 — OPEN BY DESIGN

Freedom to choose your AI

In today's AI market the leader changes every few months — OpenAI, then Anthropic (maker of Claude), then a new open-source model. Your tools shouldn't lock you in.

CLOSED PLATFORM

One AI, and you're a hostage

Build inside a closed platform wired to a single AI, and you're tied to that one company's pricing, limits, and roadmap.

VS CODE

Open through extensions

Connect your workspace to any model through extensions:

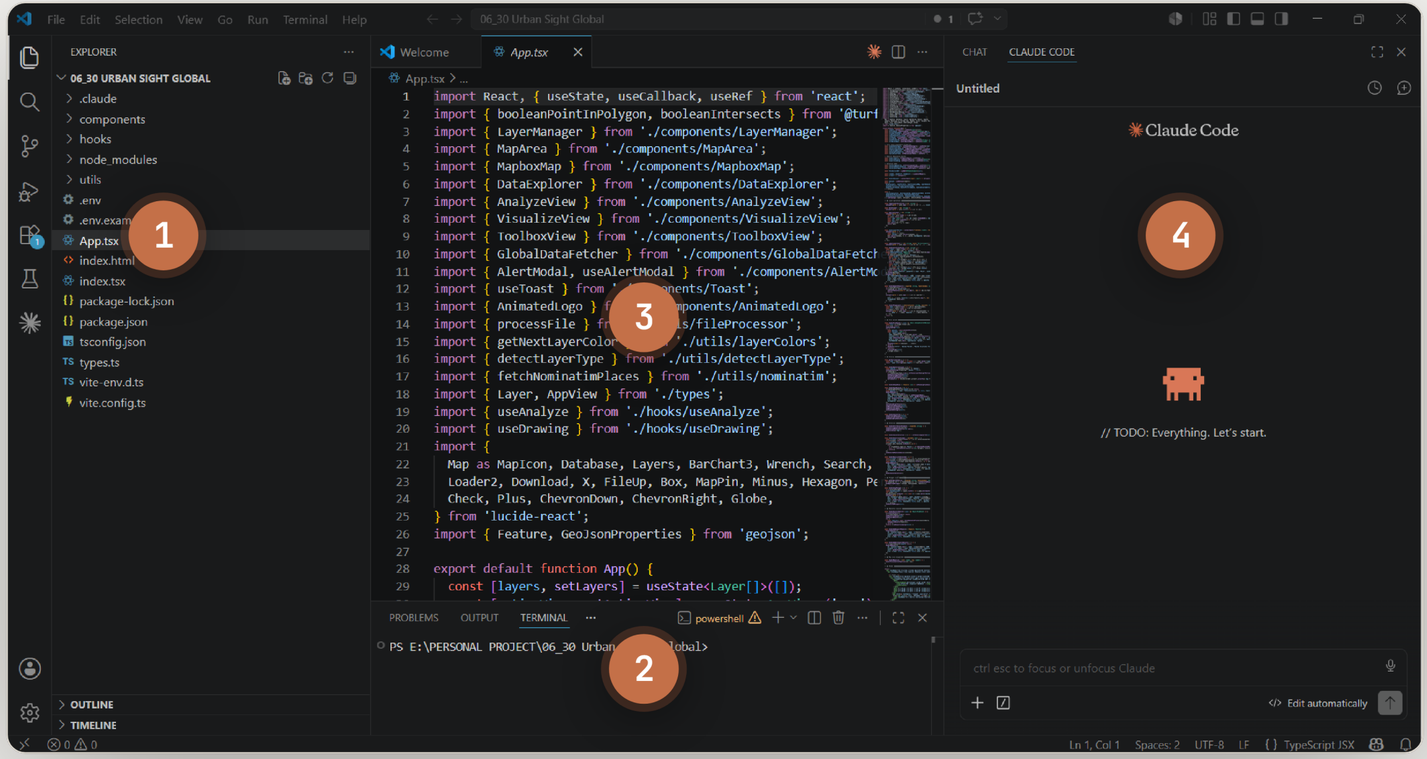
`Claude Dev` `Continue` `Cursor` `Roo Code`

A better model ships tomorrow? Just swap the `API key` in a lightweight extension — no code changes, no moving files.

• 03 — THE WORKSPACE

A tour of the VS Code workspace

It looks busy at first, but it's really just four areas working together. Here's the lay of the land before we walk through each one.



01 Sidebar
Left · your files

02 Terminal
Bottom · run commands

03 Editor
Center · your code

04 AI chat
Right · your assistant

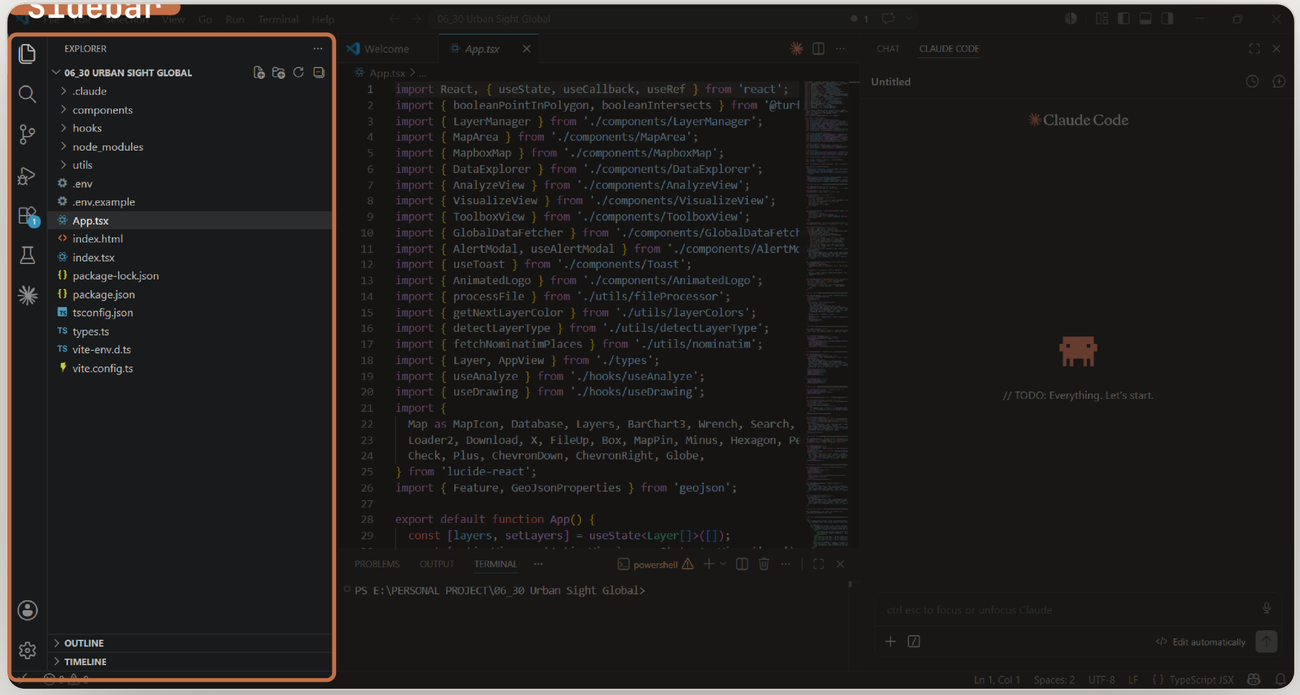
THE WORKSPACE · 01

The sidebar

left

The narrow strip of icons and the panel beside it are your map of the project. The icons switch tools; the panel shows what you're working on.

- Files — browse and open every file in your project.
- One click each for Search, version control (Git), Extensions, and Run & Debug.



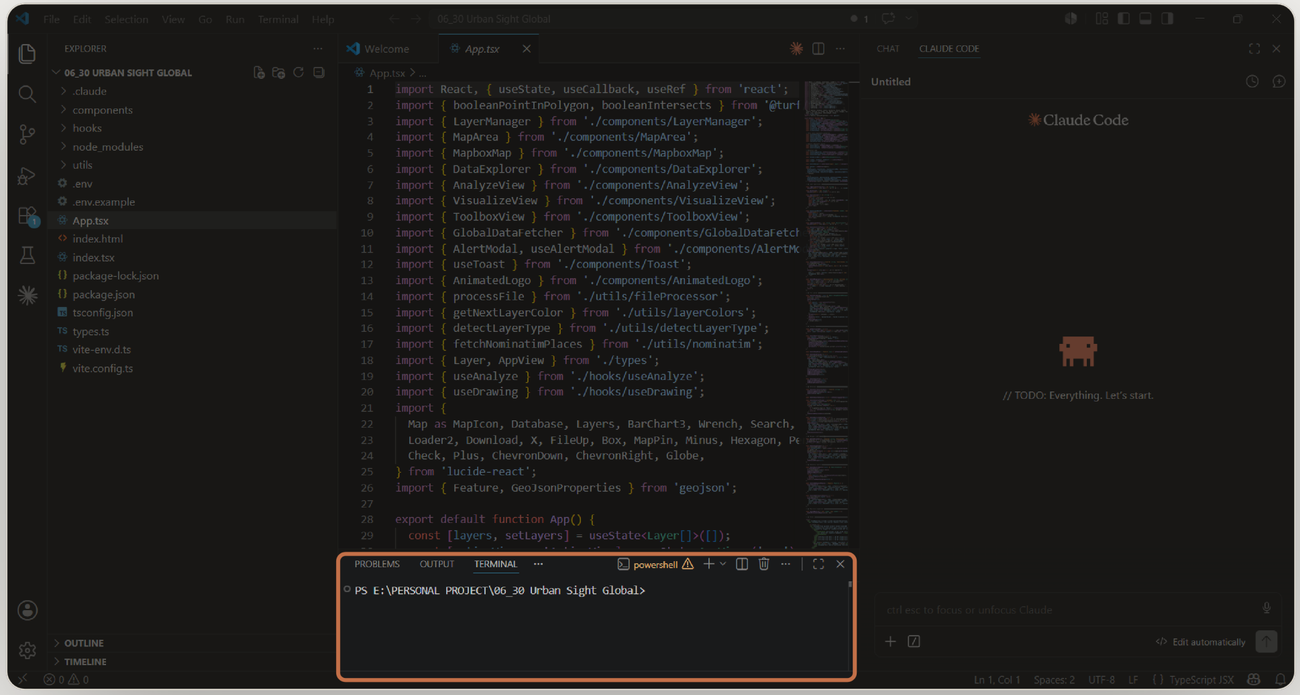
THE WORKSPACE · 02

The terminal

bottom

A command line built right into the window. Run your project, install packages, and use Git without ever leaving VS Code.

- Start your app, watch its output, and stop it — all in one place.
- It opens inside your project folder, so commands just work.



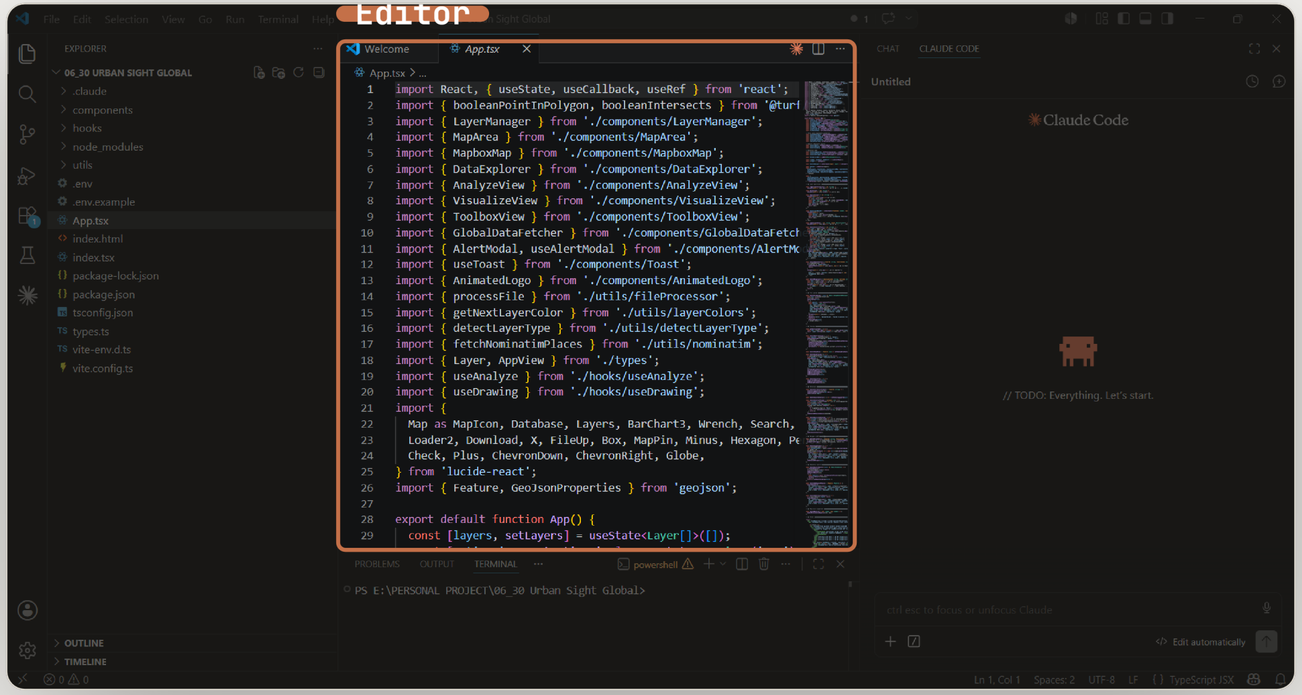
THE WORKSPACE · 03

The editor

center

The main stage, where you read and write code. Files open as tabs across the top, so you can jump between them in an instant.

- Color-coded syntax makes code easy to scan and understand.
- Keep several files open as tabs and switch with a click.



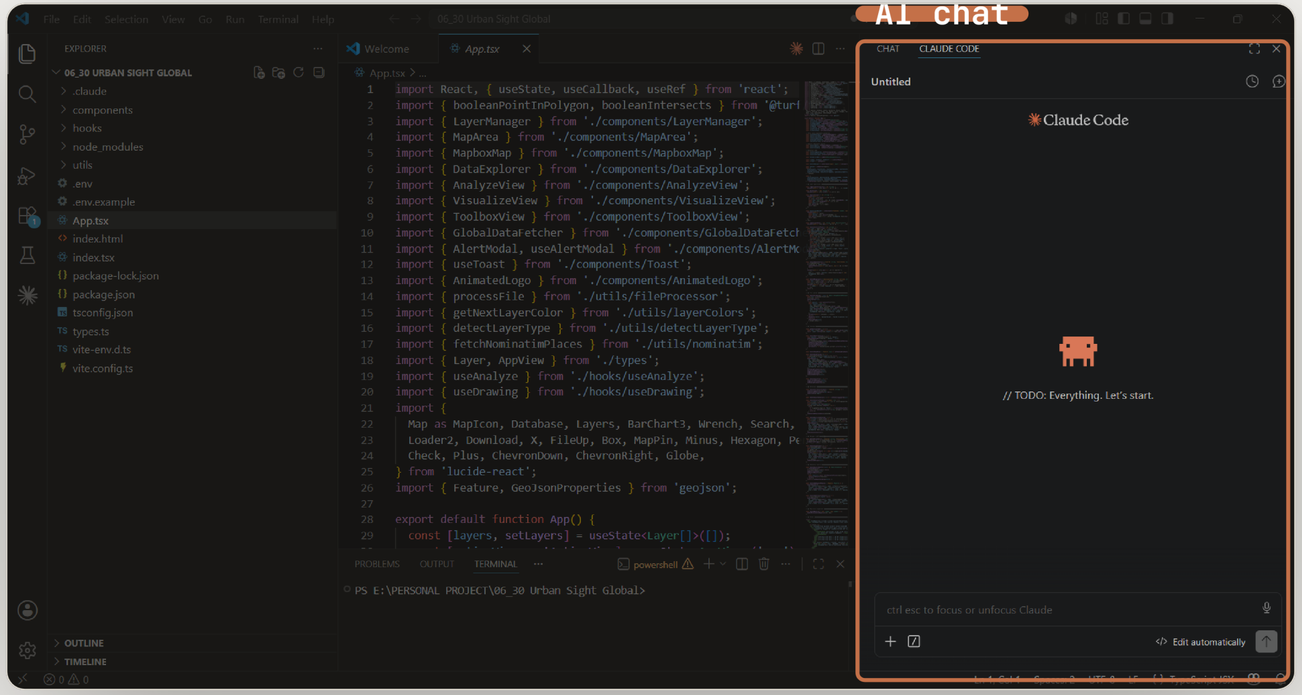
THE WORKSPACE · 04

The AI chat

right

Your AI assistant sits in a panel next to your code. Describe what you want in plain language and it edits the files directly.

- It can see your project, so its answers fit your actual code.
- Swap which AI powers it whenever a better model comes along.



04 — PRICING

What the AI engines cost

Because VS Code connects to any model, you choose how much to spend. Here's the ladder most people climb — from free, to a few dollars, to today's strongest.

Free

GitHub Copilot

`older GPT-class model`

Comes with any free GitHub account — the simplest way to switch it on.

Fine for getting started and seeing how it feels.

Free · students

GitHub Student

`wider choice of models`

Free for verified students, with access to a broader set of engines.

Usage caps apply — check your current monthly limit.

~$2 · pay-as-you-go

DeepSeek

`DeepSeek (China)`

Add it yourself and top up credits — a couple of dollars buys many hours.

Cheap and surprisingly capable for everyday work.

RECOMMENDED

$20 / month

Claude

`Claude`

The current sweet spot for coding — strong on complex builds and simple edits alike.

Best balance of quality and cost as of this talk.

• 05 — GETTING STARTED

Building your first project

With everything connected, here's how to actually begin — and the one habit that makes complex builds possible.

01

Open a new folder

Make an empty folder and open it in VS Code. That folder is your project.

02

Start with small requests

Ask the AI for one small, clear thing at a time — and check each result before moving on.

03

Split the big idea

A complex site or platform comes together only by breaking the idea into many small requests.

05 — WHEN SOMETHING BREAKS

See the problem, then fix it

When your site misbehaves, the browser already knows why — it just keeps the explanation hidden until you ask for it.

`Press F12`

01 Open the developer tools

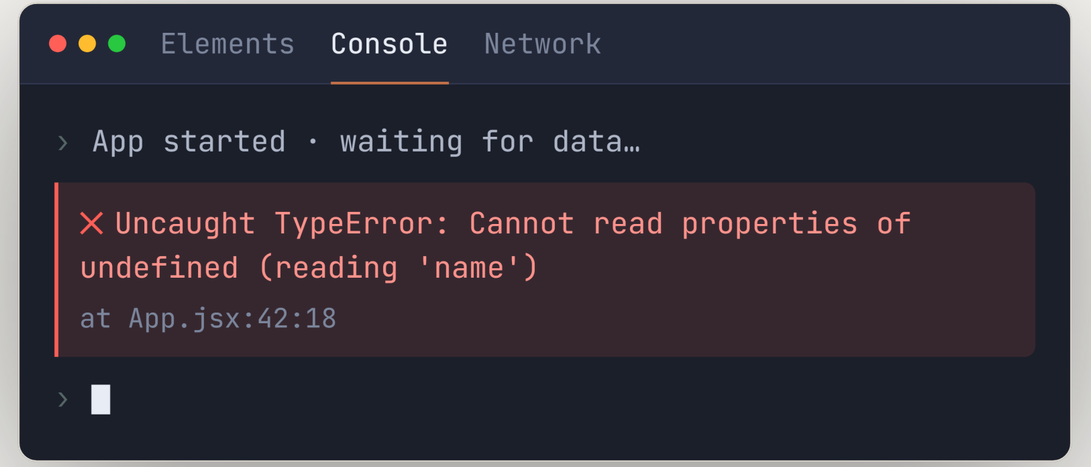
Hit F12 in the browser and pick the Console tab. This is where the page reports what's going wrong.

02 Find the red error

Errors show up in red, with a message and the file and line where it happened.

03 Copy it straight to Claude

Select the red text, copy-paste it to Claude, and ask it to fix the problem. The error message is all the context it needs.



SECTION

A few concepts worth knowing

Before you build something bigger, three terms come up again and again.

01 Sites vs platforms 02 GitHub 03 Vite & the dev server

THE BASICS · 01

Start with plain HTML

HTML is the base layer of the web — the simplest thing to build, with almost no setup to get going. Whatever little setup there is, Claude runs it for you.

Going further

Platforms need more

A real platform means extra installs. Your project folder grows as they pile up — but that's exactly what unlocks complex features.

Pulling in data

What an API is

An API is how your app fetches data that's protected or paid — a bridge to outside services and their information.

* "Installing" here isn't downloading from a website. In the terminal you type one line — like `npm install` — and it fetches the code libraries and helper tools your project needs.

GitHub

Where your code lives in the cloud — and how teams work on the same project without overwriting each other. Programmers reach for it constantly.

Push

git push sends your latest changes up to GitHub, so they're saved and shared with everyone.

Pull

git pull brings down the newest changes others have made, so your copy stays up to date.

Fork

A fork is your own copy of someone else's project to build on — without touching the original.

THE BASICS · 03

Vite & the dev server

Vite runs a small web server on your own machine, so you can see your site while you build it.

```
$ npm run dev
```

Run this once to start the server. Open the address it prints, and your site is live on your computer — refreshing automatically every time you save.

06 — BACKEND, HANDLED

Building a platform without a backend

Traditionally, an app with real users meant two teams. Today, one of them comes ready-made.

FRONTEND

What you build

The part users see and touch — buttons, layout, graphs. You shape it together with the AI.

Your job: how it looks and behaves — then ask the AI to connect a button to your database.

BACKEND

Handled for you

Server, database, security, cloud — provided out of the box by BaaS platforms, which the AI wires up in a few lines of code.

`Supabase` `Firebase`

A `generous free tier` — perfect for early startups and student projects.

07 — THE TOOLKIT

A few tools worth knowing

As your project grows, a handful of services do the heavy lifting — most with a free tier to start.

Vercel

Hosting — publish your site to the web in a click, with a live URL.

CLOUDFLARE

Speed & security — a fast, protective layer in front of your site.

clerk

Accounts — sign-up and login, without building auth yourself.

PostHog

Analytics — see how people actually use what you've built.

PART TWO

02

Let's actually build it

From here on it's hands-on — the exact steps, from signing up to putting your site online.

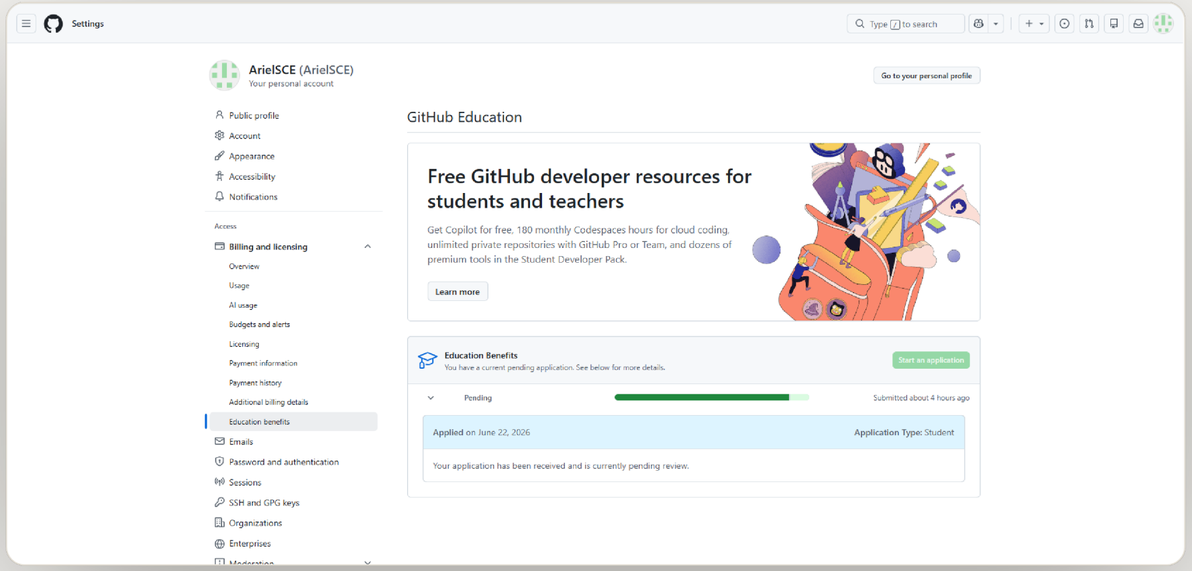
STEP 01

Create your GitHub account

GitHub is the home base for everything that follows — your account, your code, and your free AI access all start here.

01 Sign up for a free account at **github.com**.

02 A student? Apply for **GitHub Student** — same account, with extra tools and a wider choice of AI models for free.



STEP 02

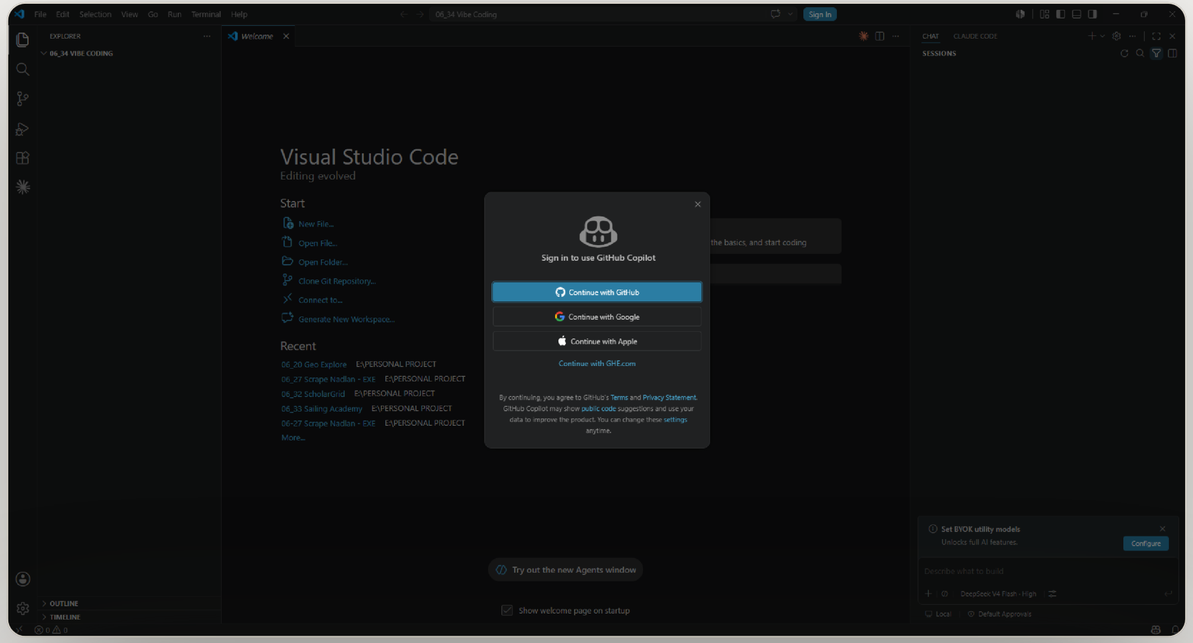
Open VS Code & connect your account

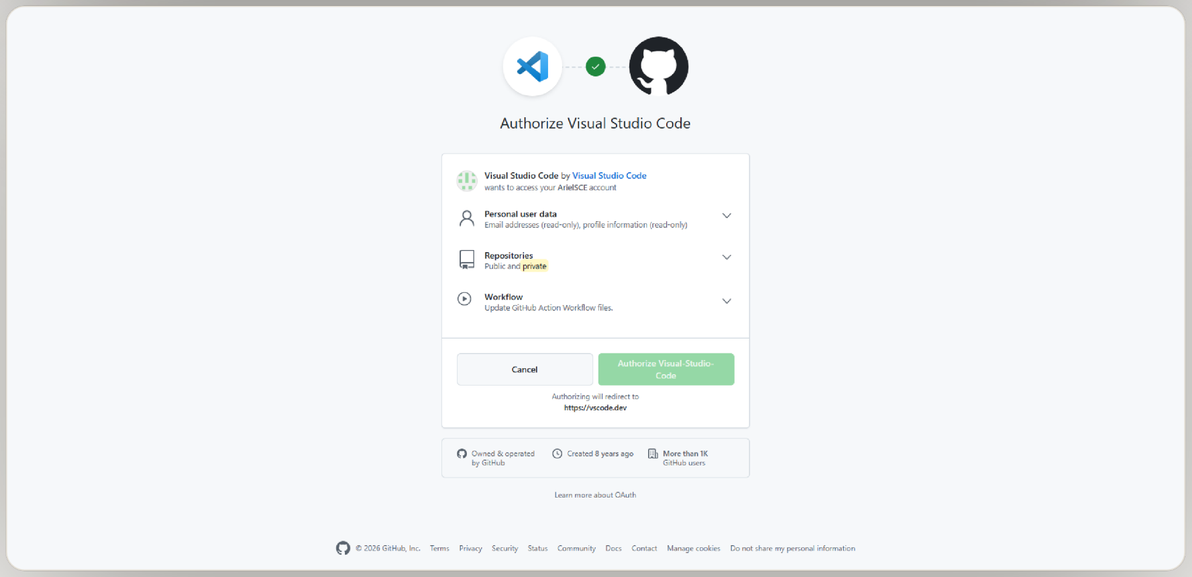
Launch VS Code and sign in with the GitHub account you just made. That link is what unlocks your AI assistant inside the editor.

01 Open VS Code and find the **Accounts** menu.

02 Sign in with **GitHub** and approve in the browser.

03 You're connected — the AI panel is ready to use.





STEP 03 · YOUR FIRST PROMPT

Ask for your first site

Type a plain-language request in the AI panel. Keep it simple — a basic HTML site is the perfect way to learn the rhythm of vibe coding.

YOU → CLAUDE

```
Create a simple HTML website for a personal recipe page. One page, a title, a short intro, and a list of three favorite recipes. Keep the code plain HTML and CSS so it's easy to read.
```

Claude writes the files and runs any setup for you. Open the preview to see it live — then ask for one small change at a time to keep going.

STEP 04 · PUT IT ONLINE

Push it to GitHub & go live

Once you're happy with it, one more prompt saves your code to GitHub and publishes it to a real web address you can share.

YOU → CLAUDE

```
Push this project to a new GitHub repository, then deploy it so it's available online with a public link I can share.
```

Your code now lives safely on GitHub and on the web. Every future change can be pushed the same way — saved and published in one step.