

Ways to use AI: Chat, API, CLI, and Code Tools

Reading time: ~12 min · Prereqs: Lessons 01–02 · *No coding required to read this page.*

One building, four doors

When people talk about "using AI," they usually picture a chat window — you type something, the AI replies. That's real. But it's just one of four ways to reach the same underlying model. The AI itself hasn't changed. What changes is *how you get access to it*.

Think of Claude or ChatGPT as a brilliant expert sitting in a building. You can walk in the front door and talk face-to-face. Or you can send requests through a service window. Or you can phone in using a specific script. Or you can hire an assistant who goes in, does research, changes files on your behalf, and comes back with results. Same expert. Four very different kinds of access.

We call these four access methods **doorways**. This lesson covers each one — what it is, when to use it, and what skill level it actually requires.

Try the Doorway Picker below to get a personal recommendation before diving into the reading. It only takes 30 seconds.

Which AI doorway is right for you?

Answer 3 quick questions and get a personal recommendation.

Q1 — Are you just starting out with AI?

Yes, brand new

Sort of — tried it a few times

No, I have experience

Q2 — Do you need to do the same AI task many times automatically?

No, just occasionally

Yes, many times

Yes, on a schedule

Q3 — Do you want AI to read and edit files on your computer for you?

No

Sometimes

Yes, often

Find my doorway →

Try different answers

Doorway 1 — Chat (the front door)

A chat interface is a website or app where you type in a box and the AI replies.

Claude.ai, ChatGPT, and Gemini all work this way. You've probably already used one. No installation, no account with a credit card required on the free tier, no coding. It's as close to "just talking" as technology currently gets.

What can you actually do with chat? More than most people expect. Drafting emails, explaining complex documents, brainstorming ideas, summarizing meeting notes, translating text, generating code snippets, answering "what does this mean" questions — all of these work well. The AI sees exactly what you type in that conversation window, nothing more. It can't browse the internet (unless you enable a specific "search" feature), and it can't open files on your computer (unless you drag them into the chat). Everything happens in that conversation bubble.

The constraint of Chat is scale and repetition. If you have 800 customer emails and want a one-line summary of each, copy-pasting 800 times is not a plan. That's when the next doorway becomes relevant.

Start here. Spend at least a week using Chat before touching anything else. The muscle memory you build — how to ask clearly, how to refine a bad answer, when to trust and when to verify — transfers directly to every other doorway.

Doorway 2 — API (the kitchen window)

API stands for Application Programming Interface. The word is intimidating but the idea is simple: instead of *you* typing into a chat box, *your code* sends the message and gets the answer back automatically.

Here's the restaurant analogy that makes it click. The chat window is the dining room — you sit down, a waiter takes your order, brings it back. Lovely. The API is the kitchen's service window. You slide a structured ticket through: "here's the prompt, here's the model, here's the max length of the answer." The kitchen sends back a structured response. You can do this a thousand times in a loop. No waiter needed.

The API is how real products are built on top of AI. The "AI-powered writing assistant" in your favorite app? It's calling an API. The tool that automatically categorizes your support tickets? API. The nightly job that summarizes the day's news for your team? API.

To use the API you need to write a little code (Python is the most common choice), and you need an API key — a secret credential that identifies your account. Module 6 covers the hands-on steps in detail. For now, just know: API = "let my program talk to AI, automatically, at any scale."

API keys are like passwords with a credit card attached. If someone else gets your key, they can run AI requests on your bill. Keep it in an environment variable, never in code you share, never in a chat message. Module 6 covers this properly.

Doorway 3 — CLI (the terminal)

The CLI (Command-Line Interface) is the terminal — that black or white text window where you type commands instead of clicking buttons. Some AI tools are distributed as command-line programs you install and run there.

A real example: **Ollama** is a tool for running open-source AI models on your own computer. You install it, then you can type `ollama run llama3.2` and start a chat right in your terminal. No browser, no account, no internet after the initial download.

The CLI looks intimidating at first because it strips away every visual hint. But the mental model is simple: you type an instruction, you press Enter, something happens. It's not fundamentally harder than a chat window — it's just less decorated.

CLI access is most useful when you're comfortable enough with technology to want fast, scriptable tools. You'll get a gentle introduction to it in Module 4 when we install Ollama. For now, you don't need to touch it.

Doorway 4 — Code & Agentic Tools

This is the newest category and the one that most changes what AI can do for you. In Chat, API, and CLI, you're always the one deciding each next step — you ask a question, you get an answer, you decide what to do with it.

With an **agent**, you give a *goal*, and the AI figures out the steps. "Fix the bug in this file" isn't a single question — it requires reading the file, understanding the code, identifying the problem, writing a fix, testing it, and reporting back. An agentic tool does all of those steps. It uses tools (file access, the ability to run commands) on your behalf.

Examples of agentic tools: **Claude Code** (built into this learning platform, actually), **Cursor** (an AI-first code editor), **GitHub Copilot** (AI suggestions and chat inside VS Code). Each one connects an AI model to your actual codebase or file system.

Agent vs. Chat — visualized

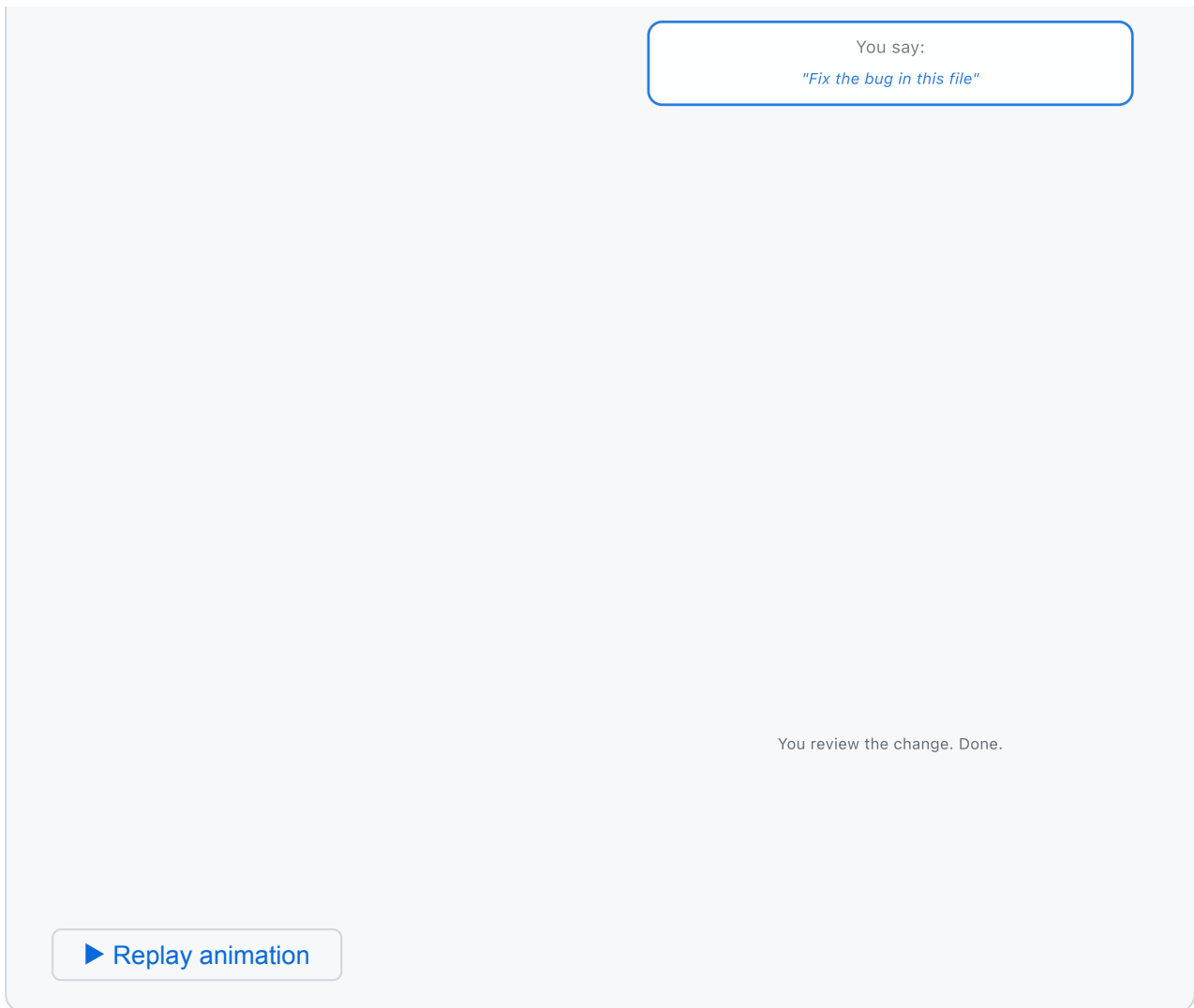
Watch how a one-shot Chat exchange compares to an agent working through multiple steps.

CHAT (ONE EXCHANGE)

You type:
"Summarize this article"

AGENT (MULTI-STEP)

Done. You take it from here.



The key difference isn't just convenience. It's *scope of action*. Chat gives advice. An agent *takes action*. That power comes with responsibility — a well-designed agent will show you what it's about to do and ask for confirmation. You should always review before saying yes.

Agents can change real files. They can delete code, rename things, run terminal commands. That's why they're powerful. And it's why Module 9 covers agent safety separately. For now, treat agents like a capable but new employee — supervise before trusting.

The comparison at a glance

The table below summarizes the four doorways. **Click any row** to expand it and see an example interaction.

Doorway	Skill needed	Install?	Use it for
►Chat	None	No	Everyday questions, writing, learning
►API	A little coding	A little	Automating tasks, building apps
►CLI	Comfort with commands	Yes	Local tools, quick experiments
►Code / Agentic Tools	Some technical comfort	Yes	Coding, multi-step tasks on real files

The beginner path — don't skip Chat

Here is a very clear recommendation: **start in Chat and stay there until Chat is not enough.**

Chat handles the vast majority of AI use cases. Writing assistance, question answering, summarization, code explanation, translation, brainstorming — all of this works right in the browser, free tier, no setup. Most people who "need" the API don't actually need it yet.

How do you know when Chat is not enough?

- **"I need to do this same thing 200 times"** → that's when you move to the API.
- **"I want it to read and edit files on my computer"** → that's when you explore Code tools.
- **"I want to run a model privately with no internet"** → that's when you install Ollama and touch the CLI.

There's no prize for using a more technical doorway than you need. The expert in that building gives you the same quality advice whether you walk through the front door or the service window. Use the front door until you have a reason not to.

Module 4 action plan: This week, open Claude.ai or ChatGPT and use it for three real tasks you'd normally do yourself (draft an email, look something up, explain a

concept). Notice where it helps, where it needs guidance, and where it gets things wrong. That hands-on experience is worth more than any reading.

Quick glossary

Chat interface

A browser-based or app-based conversation window. You type, AI replies. Claude.ai, ChatGPT, Gemini.

API (Application Programming Interface)

A way for your code to send requests to the AI automatically. Needs an API key and minimal coding.

API key

A secret credential that identifies your account with the AI provider. Treat it like a password.

CLI (Command-Line Interface)

The terminal. A text window where you type commands. Some AI tools (like Ollama) run here.

Agent / agentic tool

An AI that can take multi-step actions — read files, write code, run commands — on your behalf toward a goal.