prompts.chat MCP Setup Guide

Connect the 10,000-prompt open-source library to Claude Code so your agent finds the right prompt instead of you.

Why I bothered with this

I kept a notes file of prompts. Everybody does. Mine had about forty in it, half of them stale, and I still opened a browser tab every time I needed something outside that forty. prompts.chat has more than 10,000 prompts covering writing, coding, images, video and research, and the part that made me stop and install it is that the whole library runs as an MCP server. Claude Code can search it mid-task.

The project used to be called Awesome ChatGPT Prompts. It has been going since December 2022, it is past 140,000 GitHub stars, and the prompt data is CC0 1.0, which means public domain. You can copy, edit and ship the prompts commercially with no attribution. The site code itself is MIT.

What this guide covers: the config, the eight tools, the setup for both Claude Code and Codex, the five prompts I actually type, and the eight ways it goes wrong. Searching and reading needs no account and costs nothing. You only need an API key if you want to save prompts back.

01

How it actually works

1. The remote MCP server sits at https://prompts.chat/api/mcp and speaks JSON-RPC over HTTP. Once it is in your config, Claude Code sees eight new tools it can call by itself, the same way it can call Read or Bash.
2. When you ask for something prompt-shaped, the plugin's auto-activating skills fire and Claude calls search_prompts with your keywords, plus a category, tag or type filter if one fits. It gets back a list of matching prompts with their ids.
3. It picks one, calls get_prompt to pull the full text, substitutes the prompt's variables using what it already knows about your project, and runs it in the same session. You never see a clipboard.

02

The one thing almost everyone gets wrong

People install the plugin and then keep copy-pasting. The tools sit there unused, because installing a tool does not change a habit. The fix is embarrassingly small: once, at the start of a task, tell Claude out loud to check prompts.chat before writing a prompt from scratch. After three or four sessions it starts reaching for the library without being asked, because the auto-activating skills learn what prompt-shaped requests look like in your work.

WATCH OUT

Read a community prompt before you run it. Plenty of them were written for a chat window, so they assume you will paste text in or upload a file. Inside a coding agent with file access those instructions are wrong and will send Claude down a dead end. Ask it to strip the paste-and-upload language and rewrite the prompt for an agent that can read files directly.

03

The config and the commands, in full

SOURCE CODE

The whole project is open source, prompts included:

https://github.com/f/prompts.chat

CLAUDE CODE PLUGIN INSTALL (PASTE BOTH LINES)

```
/plugin marketplace add f/prompts.chat
/plugin install prompts.chat@prompts.chat
```

.MCP.JSON (REMOTE, RECOMMENDED)

```
{
  "mcpServers": {
    "prompts.chat": {
      "url": "https://prompts.chat/api/mcp"
    }
  }
}
```

.MCP.JSON (LOCAL, RUNS ON YOUR MACHINE)

```
{
  "mcpServers": {
    "prompts.chat": {
      "command": "npx",
      "args": ["-y", "prompts.chat", "mcp"]
    }
  }
}
```

SLASH COMMANDS THE PLUGIN ADDS

```
/prompts.chat:prompts code review
/prompts.chat:prompts writing assistant --category writing
/prompts.chat:prompts midjourney --type IMAGE
/prompts.chat:skills testing automation --category coding
```

> **SITES SHOWN IN THE VIDEO**
>
> - prompts.chat, browse the library
> **https://prompts.chat/**
> - MCP and API documentation
> **https://prompts.chat/docs/api**
> - Your API key, only needed for saving
> **https://prompts.chat/settings**

Set it up in Claude Code or Codex

1. Open Claude Code in whatever project folder you work in. This installs to your user config, so the folder does not matter much, but it does need to be a real project.
2. Run **/plugin marketplace add f/prompts.chat**. That registers the marketplace. If Claude Code says the command does not exist, your version is too old, so update it first.
3. Run **/plugin install prompts.chat@prompts.chat**. This installs the MCP server, two slash commands, a prompt-manager agent, a skill-manager agent and two auto-activating skills in one go.
4. Quit Claude Code completely and reopen it. MCP servers are read at startup, so a reload is not enough. Then run **/mcp** and confirm prompts.chat shows as connected.
5. If you would rather skip the plugin, paste the remote .mcp.json block above into your MCP config by hand instead. Same server, fewer extras.
6. For Codex there is no plugin marketplace, so hand-editing is the only route. Add the same mcpServers JSON to Codex's MCP config file, keeping the url form, and restart Codex. If you keep a project-level skills folder, put a one-line pointer in AGENTS.md so Codex knows the library exists.
7. Only if you plan to save prompts or skills back to the library: get an API key from https://prompts.chat/settings, then ask Claude to add the line **export PROMPTS_API_KEY=your_key_here** to your shell profile. Open a brand new terminal afterwards, otherwise the variable will not exist in the session you are in.
8. First test: type **/prompts.chat:prompts code review** and press enter.
9. A successful first run looks like a short list of prompt titles with ids next to them, and Claude offering to pull the full text of one. If you get an empty list or a connection error, jump to the fix table at the end.

Folder naming only matters if you also install skills from the library. If you do, the folder name under ~/.claude/skills/ has to match the name: field inside that skill's SKILL.md exactly, or Claude will never load it.

> 💡 **QUICK TIPS**
>
> - Leave it read-only for the first week. Searching, reading and improving prompts needs no key at all, and the save tools are the only ones that can write anything anywhere.
> - Want to browse without installing a thing? Run **npx prompts.chat** in a terminal. It opens the library as a CLI.
> - The local config is the one to reach for on a flaky connection or a locked-down network, since it runs the server on your own machine.

---

04 The commands I actually run

FIND THE PROMPT BEFORE WRITING ONE

```
Search prompts.chat for a prompt about [TASK]. Show me the top three with their ids, say which one fits my situation and why, then pull the full text of the one you pick and fill in its variables from what you already know about this project. Do not run it yet.
```

EXPAND A ROUGH IDEA

```
Here is my rough prompt: [PASTE YOUR ROUGH PROMPT]. Run improve_prompt on it, then show me what changed and why in a short list. Keep the result under [NUMBER] words and keep my original constraints intact.
```

IMAGE PROMPTS THAT MATCH MY BRAND

```
Search prompts.chat with --type IMAGE for prompts about [SUBJECT]. Give me three, rewrite each one for [IMAGE MODEL], and force my brand colours [HEX] and [HEX] into every version. No stock-photo language.
```

BORROW A SKILL INSTEAD OF BUILDING ONE

```
Search prompts.chat skills for [CAPABILITY]. Get the full skill with all its files and show me SKILL.md before you write anything to disk. Install it to ~/.claude/skills/ only after I say go.
```

MAKE IT A HABIT FOR THE SESSION

```
For the rest of this session, before you write any prompt from scratch, check prompts.chat first and tell me what you found. If nothing there fits, say so and write your own.
```

When it breaks

ERROR	FIX
/plugin marketplace add is not a known command	Your Claude Code is too old for plugin marketplaces. Update it, quit fully, reopen, then run the two install lines again.
Plugin installed but no prompts.chat tools appear	Quit Claude Code completely rather than reloading. MCP servers are only read at startup. Reopen, run /mcp, and check prompts.chat is listed as connected.
Claude never calls the tools on its own	Say it once at the top of the task: check prompts.chat before writing a prompt. The auto-activating skills fire on prompt-shaped requests, not on everything you type.
save_prompt or save_skill returns an authentication error	Those two need an API key. Get one from https://prompts.chat/settings and set PROMPTS_API_KEY. Search, get and improve work without any key.
PROMPTS_API_KEY is set but still rejected	You exported it in one terminal, or after Claude Code was already running. Put the export line in your shell profile, open a new terminal, and start Claude Code from there.
Connection to https://prompts.chat/api/mcp times out	Switch to the local config: command npx, args -y prompts.chat mcp. It runs the server on your machine and skips the remote endpoint entirely.
The prompt it pulled tells me to paste or upload something	That one was written for a chat interface. Ask Claude to strip the paste-and-upload steps and rewrite it for an agent that reads files directly.
Codex cannot see the server	Codex has no plugin marketplace. Add the mcpServers JSON to Codex's MCP config file by hand, keep the url form, and restart Codex.
An installed skill from the library never loads	The folder name under ~/.claude/skills/ has to match the name: field in that skill's SKILL.md exactly. Rename the folder and restart.