The Claude Cheat Sheet

Everything from today's session. Keep it next to your keyboard, steal the prompts on the last page.

00 THE GOLDEN RULE

> When Claude says it can't do something, that is almost never an intelligence problem. It's a tool problem. Don't accept the no. Ask: "What tool or access would you need to do this?" It will usually tell you, and often install it itself.

01 INSTALL CLAUDE CODE

Open the terminal first. Mac: press Cmd+Space, type "Terminal", hit Return. Windows: press the Windows key, type "PowerShell", hit Enter.

Then paste one line. No developer tools needed. Log in with your normal Claude account (any paid plan).

```
$ curl -fsSL https://claude.ai/install.sh | bash
```

Windows (PowerShell):
`irm https://claude.ai/install.ps1 | iex`

Mac, don't skip: when it finishes it prints a line starting with `echo`. Copy that whole line, paste it back in, hit Return. Then type `claude` in any folder and just talk.

First thing to do: run prompt P1 on the last page so typing `c` launches Claude.

02 GIT & GITHUB

> **Git is a save system for a whole folder. Every commit is a save point you can always return to.**

GitHub is the photo album in the cloud: backup, sharing by link, and the socket everything else plugs into (your site host watches it and redeploys on every push). You'll never type a git command. Claude does the git.

03 SKILLS

A skill is a recipe card: a folder with a `SKILL.md` saying "when I ask for X, do it exactly this way."

- Drop into `~/.claude/skills/`, or say "install this skill from this GitHub link"
- Project skills live in the repo and travel with it
- Bundles install via `/plugin`

Anything you've told Claude twice should become a skill.

04 THE FOUR DOORS INTO YOUR OTHER APPS, WEBSITES, AND ACCOUNTS

- **MCP**: the standard plug. Connect once (`/mcp`), Claude gets buttons for that product
- **API + access tokens**: a revocable special password; Claude writes the code
- **Browser control**: Claude drives your real Chrome. If you can click it, Claude can
- **Dedicated accounts**: give Claude its own login, scoped and shut-off-able

05 EMAIL THAT SENDS ITSELF

- The official Gmail connector **drafts but won't send**. That's a guardrail, not a wall
- Real sending: your own Gmail API token, or Apple Mail on a Mac
- The method: **draft-first two weeks**, then auto-send one narrow category, then widen
- Feed it real emails you sent so it writes in your voice

06 LIVING IN THE TERMINAL

- **Installed something?** Quit Claude and start it again. New skills, plugins, and MCPs load at startup
- **Pick up where you left off:** `claude --continue`, or `claude --resume` to choose from a list
- **Before you quit**, ask: "write me a prompt I can paste into a fresh session to continue this work." Paste it in the new one
- **Drag any file onto the terminal window** and it types the path for you
- **Windows:** can't paste an image, but **Ctrl+Shift+C** in File Explorer copies the file path. Paste that. Mac: Cmd+Option+C

Why Use the Terminal

The Bicycle and the Motorcycle

The app and the terminal both get you there. One takes your effort all day. The other takes ten minutes.

> The Claude app is a brilliant talking head. It can tell you exactly what to do, and then you go do all of it yourself. The terminal gives Claude arms. Same brain, but now it can actually do the work.

A WHAT THE ARMS CAN DO

- Search and read every file on your computer
- See what's on your desktop and organize the whole machine
- Edit videos, resize photos, convert anything into anything
- Diagnose and fix things that are broken. A permanent IT partner, on call, no ticket queue
- Build and ship real things: websites, documents, apps, automations that run while you sleep

In the app you receive advice. In the terminal you receive **finished work**.

B YOU OWN EVERY BIT OF THE WORK

In the app, your work lives inside a chat window. Lose the thread, lose the login, lose the work.

In the terminal, everything Claude produces is a **real file on your computer**. The chat can end; the work stays. Tomorrow you can switch models, bring in a different AI, even run one locally someday, and every bit of what was built is still sitting right there, yours forever, no login required.

C IT COMPOUNDS

A chat forgets. Your computer doesn't.

In the terminal, Claude writes what it learns into files that persist: your preferences, your projects, your history, your standards. Every session starts smarter than the last, because the last one left notes.

The app gives you conversations. The terminal builds you an **asset that compounds**.

D ABOUT THE RISK

Yes, more power means more responsibility, the way a motorcycle asks more respect than a bicycle.

But the risk is managed for you: Claude **asks permission** before touching anything that matters, and git puts a save point behind every change, so nothing is ever truly lost.

At our level, this is a solved problem. Ride the motorcycle.

> Bicycles are for rides around the block. Motorcycles are for actually going somewhere.

06 PICK THE MODEL, SET THE EFFORT

MODEL	REACH FOR IT WHEN	EVERYDAY EFFORT	CRANK IT TO
Haiku	Chores: renames, bulk edits, quick lookups	low	medium when the output still matters
Sonnet	The daily driver for most building	medium	high for real work sessions
Opus	Hard debugging, big features	high	xhigh for architecture and thorny bugs
Fable 5	Long autonomous jobs, deepest reasoning	high	max for the one problem nothing else solved

Switch with `/model`, dial with `/effort`. Effort is how long it thinks per step; model is the ceiling.

> Match the model to the stakes, effort to the difficulty. When output disappoints, raise a dial before you blame Claude.

07 WHICH PLAN

Pro is priced for chatting. Agentic work re-sends the whole conversation every step; one serious session can out-eat a month of chat.

PLAN	REALITY
Pro $20	Learn on it. Caps will interrupt real work
Max $100	About 5x. The floor for daily business use
Max $200	About 20x. Agents running all day

Upgrade the day a cap costs you more than the upgrade does.

08 ULTRACODE: THE SWARM WORD

Put the word `ultracode` in your prompt and Claude stops being one worker and becomes a fleet: it plans the job, fans out parallel subagents (searchers, builders, and skeptics that double-check every finding), then merges the results.

For audits, deep research, and "be exhaustive" jobs. It spends serious usage on purpose. Sibling: `/code-review ultra`, a multi-agent review of a branch or PR.

09 CLAUDE WHILE YOU SLEEP

- **Cloud sessions** (claude.ai/code): run on Anthropic's machines, laptop closed. `claude --cloud "task"` sends work up, `/teleport` pulls it back
- **Routines** (`/schedule`): every morning, every hour, or webhook-triggered
- **Phone**: the Claude app starts and monitors cloud sessions, pings you when done
- Real terminal from your phone: Tailscale + SSH app + tmux to your Mac

10 COMMAND QUICK REFERENCE

`/model`	Switch models
`/effort`	Thinking depth, low to max
`/mcp`	Connect your other tools
`/plugin`	Install skill bundles
`/schedule`	Recurring cloud jobs
`/fast`	Faster Opus output, ~2x cost
`/loop`	Repeat a task on a schedule: `/loop 5m`
`/remote-control`	Drive this session from your phone
`ultracode`	Say it in the prompt: swarm mode

11 THE TOOLKIT

github.com/chrismikemagic/**claude-toolkit** · All free, open source, MIT licensed.

context-forge	Sharpens the request going in	**memory-palace**	Keeps what Claude learns as a wiki
better-opus-4.8	Structures the work, review gates	**claude-setup-optimizer**	Trims setup, searchable history

12 EVERY LINK YOU NEED

- **Docs**: code.claude.com/docs · **Cloud**: claude.ai/code
- **Browser extension**: claude.ai/chrome. Install, open it, log in, Authorize, then set it to act without asking
- **Connect apps (MCP)**: claude.ai/settings/connectors
- **API keys**: platform.openai.com/api-keys · platform.claude.com · github.com/settings/tokens
- **Tip**: an OpenAI key gives Claude image generation
- **Ship sites**: github.com + netlify.com · **Phone**: tailscale.com + termius.com

Copy, Paste, Watch It Work

Type these into Claude Code word for word. Each one changes how it works for you from that moment on.

P1 THE ONE-LETTER LAUNCH

```
Set up a shell alias so that when I type c in the terminal, it launches Claude with permission prompts skipped. Make it permanent, confirm it works, and briefly tell me what to be careful about when running this way.
```

Skipping permission prompts means Claude acts without asking each time. Speed for trust: use it in your own projects, not in folders you can't afford to break.

P2 STUDY MY BUSINESS, THEN RUN WITH IT

```
Find the Booked Solid folder on my computer and unzip anything that needs unzipping. Study everything inside it: every system, process, procedure, piece of knowledge, and tool. Save what you learn to your memory so it applies to all of my future Claude projects. From now on, when I ask you to complete a task, do your best to finish it on your own without my input: find or build the tools, MCPs, and skills you need, do your own research, and only come back to me when a decision is genuinely mine to make.
```

This is the delegation switch. One paste turns Claude from an assistant that asks into an operator that finishes.

P3 YOUR OWN MISSION CONTROL

```
Build me a personal mission-control dashboard as a beautifully designed single-page website. Before designing anything, study my brand: look at my website, my logos, my photos, and any brand files on this computer, and match my colors, fonts, and overall aesthetic. The dashboard should show my projects and what you're currently working on, stay current by reading simple status files that you keep updated, and have buttons for my most common requests. Host it so I can open it any time, and tell me the one link to check each morning.
```

Pairs beautifully with Obsidian as your knowledge base: download free at **obsidian.md/download**

P4 THE PLAIN-ENGLISH SWITCH

```
I am not very technical. From now on, explain ideas and instructions in plain language that a non-technical person can understand: short steps, no jargon, and when a technical term is unavoidable, define it in one simple sentence. Before doing anything complicated, tell me what you're about to do and why in a sentence or two. Remember this as a permanent preference.
```

Claude meets you at whatever level you ask for. Nobody is too non-technical for this to work.

P5 THE KEY VAULT

```
From now on, any time I share a password, authorization token, personal access token, or API key with you, store it in my computer's keychain so it is never lost and always accessible in a secure way. When you need it later, read it from the keychain instead of asking me again. Never write secrets into plain-text files, into code, or into anything that gets pushed to GitHub. Remember this as a permanent rule.
```

Paste a key once, never hunt for it again. The keychain is encrypted by your computer; a note on your desktop is not.

Want My Personal Dashboard?

One command center for everything you are working on, built from your own files. The prompt is free and it is waiting for you.

◆ GET THE PROMPT

chrismichael.site/claude-tools

Open that page, hit **Copy the prompt**, then paste it into Claude Code running in your home folder. It asks you questions before it writes a line of code, and the answers are what make the result yours instead of a template.

A WHAT IT BUILDS

- **A live knowledge graph of your notes**, drawn from the real links between your files. Hubs grow, hovering lights up neighbours, clicking opens the note
- **Your actual projects**, pulled from your repos with true last-commit dates
- **Calendar, metrics, and quick links** to everything you open daily
- **A command palette** on Cmd K across every note, view and project

B WHY IT IS DIFFERENT

Most dashboard prompts hand you a pretty template full of invented numbers. Most of this one is instructions to **go find your real data first**, verify its own work in a browser before handing it over, and tell you plainly which figures are samples.

> No dead buttons, no placeholder panels, and it runs with the wifi off.

C THE REST OF THE TOOLKIT, FREE ON GITHUB

- **claude-toolkit** the index, all four in one place
- **context-forge** rambling ask becomes an engineered brief
- **better-opus-4.8** plan, review and QA loop
- **memory-palace** Claude's memory as a wiki it maintains
- **claude-setup-optimizer** trims setup, searchable history
- All at **github.com/chrismikemagic**

◆ EVERYTHING FROM TODAY, IN ONE PLACE

- **This cheat sheet and the full playbook**: chrismichael.site, in Papers and documents
- **The dashboard prompt**: chrismichael.site/claude-tools
- **My free toolkit**: github.com/chrismikemagic/claude-toolkit
- **Work with me**: chris@bookchrismichael.com · chrismichael.site/#consult

THE PART I DID NOT TEACH TODAY

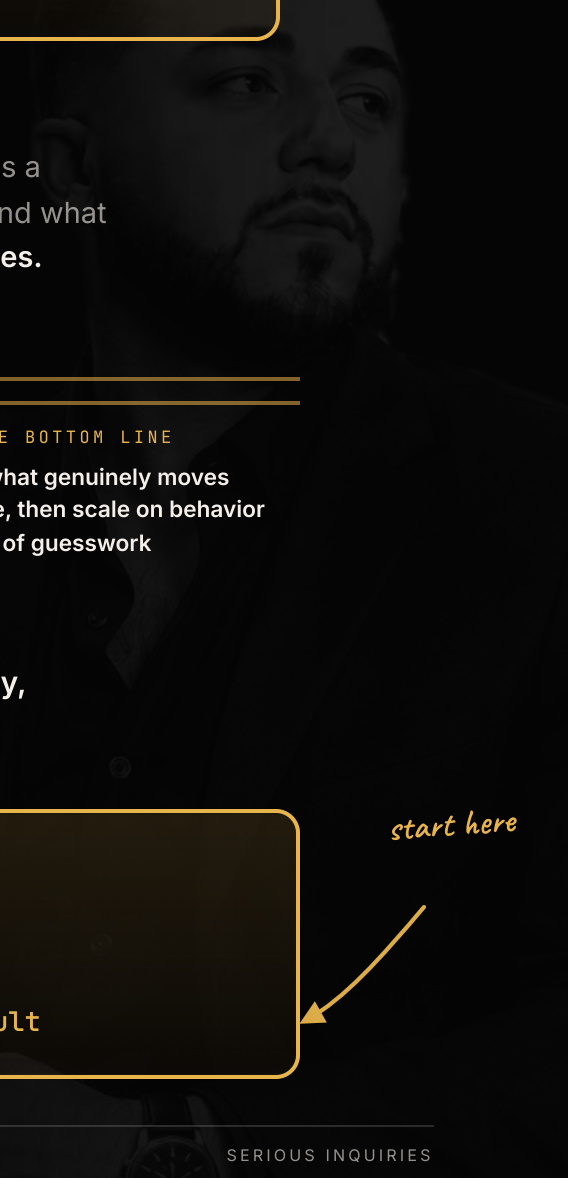
What I Build For Companies, I Can Build For You

read this part before anything else

WHO THIS IS FOR

If you run a team, and the business is doing $200k or more.

Below that line, what you already have is enough. **Today's material plus your own effort gets you to $200k, and I would rather you keep your money.** Hire me when the bottleneck stops being effort and starts being strategy.



Today was tools, and tools are the easy part. What decides who wins a category is behavior: how you build authority, how people decide, and what actually moves your bottom line. **That is the work I do with companies.**

01 AUTHORITY

Become the obvious choice in your space instead of one of the options

02 DECISION MAKING

The psychology behind how your buyers, your team, and your partners actually decide

03 THE BOTTOM LINE

Learn what genuinely moves revenue, then scale on behavior instead of guesswork

I train you and your team to be serious contenders in any industry, and I build you into a proper CEO.

start here

Your business should be outperforming everyone in your category. Let's make it.

EMAIL ME

chris@bookchrismichael.com

CONSULTING

chrismichael.site/#consult