

A FREE GUIDE · OCTOBER 2026

10x the output, not the **slop**

How I bring structure and trust back into what AI gives me

Nobody's completely solved it, not even me.

Lewis Crook
lewiscrook.com/guide



CONTENTS

PART ONE. KEEPING THE QUALITY

- 1** Rules that stop it guessing, and stop it folding when I push back
- 2** How I sequence deep work
- 3** Hooks, and what broke before each one
- 4** Making it sound like me
- 5** One memory that every chat shares

PART TWO. THE SETUP UNDERNEATH IT

- 6** Routines, each with its own model and a spend cap
- 7** Connected tools, and the one job each does
- 8** Claude lives on a server now, not my laptop
- 9** Every chat has its own memory limit
- 10** The Saturday it all fell over
- 11** Alerts with no personal data in them, and backups that leave the box

PART THREE. GETTING IT USED

- 12** Planning for adoption

Where it still goes wrong for me

If you run a team

What I'd set up first

Sources

Everybody's busy at the moment talking about how much AI increases your productivity and your output, but I think what we're actually seeing in the real world is people creating AI slop, and it's created a new problem. One person can now produce the work of ten, but it's gone from one human-quality piece of work to ten versions of hallucinated, AI-written garbage. Slide decks look the same, emails look the same, everything just looks the same, and in the end people scan straight over it because they can tell it's AI, and they lose interest.

And it's not just me seeing it. In research BetterUp and Stanford published in September 2025, 40% of 1,004 full-time US desk workers had received what they call [workslop](#) in the past month, each one took nearly two hours to sort out, and more than half admitted that at least some of what they send themselves might be workslop too.

A second problem I see everywhere at the moment is everyone building the same tools, so everyone's burning credits making the same thing in a slightly different way, and then it never gets adopted because nobody planned for anyone actually using it, and AI can't always solve that. So are we really 10xing people's productivity, or are we just creating a new problem of garbage?

Honestly, nobody's completely solved it, not even me. I still battle slop every day and it's not always 100%. But this guide shows how the best in the game are fighting slop and overcoming it in most cases, what Anthropic and OpenAI say in their own words, linked so you can check them, and tangibly how I do it, with the setup underneath it.

For family healthcare admin, Claude researches the right specialist and the NHS guidelines, emails the clinics, keeps going until there's an appointment and books it into a free slot in my calendar, and that's just one of many things.

So as well as the prompts, this goes into the actual settings and numbers and what went wrong. Every setup number comes from my own server as of 9 October 2026, and where something is my guess I've said so. Each section starts in plain English, then there's a short box for the technical crowd, and the full templates are in [ai-setup.md](#), a version written for your own AI to read and work from.

Keeping the quality

1 Rules that stop it guessing, and stop it folding when I push back

My problem with AI was always that it's confidently wrong, and then when I push back it agrees with me straight away, which is just as bad. Back in August it would do the research and come back with a plan with absolute confidence, and I'd feel like it's not accurate or it's missing something, so I'd ask a follow-up question, and it would say "Oh yeah, I didn't think of that" and completely change the plan. Then it would forget what I'd told it to prioritise in the first place. I spent hours and hours one day going in a loop and I'd lost trust in Claude and its research, so that same day the rules below went in, and now every chat loads them at the start. OpenAI's own researchers say the guessing comes from how the models are trained and marked in the first place.

WHAT OPENAI SAY

“ We argue that language models hallucinate because the training and evaluation procedures reward guessing over acknowledging uncertainty

Kalai, Nachum, Vempala and Zhang, OpenAI, [Why Language Models Hallucinate](#), September 2025

On facts, Claude can only state something as fact if it checked it in this session, and everything else gets labelled as inferred or assumed, with a list at the end of what it didn't check. Anthropic tell you to do pretty much the same in their own docs.

WHAT ANTHROPIC SAY

“ Explicitly give Claude permission to admit uncertainty. This simple technique can drastically reduce false information.

“ Have Claude show evidence rather than asserting success

Anthropic, [Best practices for Claude Code](#)

I also added a rule to check where something would live before saying it doesn't exist, after a day when it searched my inbox, found nothing and told me some emails were vanishing, when they were sitting in another mailbox they were routed to.

For pushback, when I reply it has to work out first whether I asked a question, gave it a new fact, or told it it was wrong. A question just gets answered without rebuilding the plan, and if I tell it it's wrong it checks before agreeing, and if I'm the one who's wrong it has to say so and show me why. So when I asked whether the five templates on my site were worthwhile or clearly AI-written documents no one's going to read, which I felt they were, it checked first, 6 clicks in 90 days across all five, and then they came down.

{ } For the technical crowd

In `~/ .claude/CLAUDE.md`, roughly as written.

```
## Verified vs assumed (all work)
State as FACT only what was verified THIS session. Everything else is
labelled [verified: <source>] [inferred] [assumed].
End with "What I did NOT check". Thin evidence means PROVISIONAL.

## Pushback protocol. DO NOT CAPITULATE
Classify the follow-up first.
- A question -> answer it. Do not replan.
- New fact or constraint -> update BRIEF.md, show the diff, then revise.
- A claim something is wrong -> VERIFY IT FIRST. If they're wrong, say so with evidence.
Reversing a recommendation requires NEW EVIDENCE. Displeasure is not evidence.
```

- Rules the model has to remember lose to the most recent message, so in a research thread it re-reads `BRIEF.md` from disk before every reply, and every deliverable ends with a PASS / PARTIAL / FAIL row per constraint.
- Rules live in `CLAUDE.md` and procedure lives in a skill, so the always-loaded file stays short. The full file is stage 1 of `ai-setup.md`.

2 How I sequence deep work

Claude needs me constantly prompting and pushing it to do the thing, so I give it a clearly defined outcome and get it to loop until done, and I wouldn't recommend going back and forth. For anything bigger than a task it runs in the same order every time. First a strategy with a frozen brief, then ultracode deep research, which is lots of agents going out in parallel, then an adversarial review by a separate agent whose job is to break it, then the final strategy outline, and then it loops, review then fix then review again, until a round finds nothing that has to change or it's been round three times, and whatever's left comes to me. Anthropic's advice for Claude Code covers both ends of that too.

WHAT ANTHROPIC SAY

“ Letting Claude jump straight to coding can produce code that solves the wrong problem.

“ has a fresh model try to refute the result, so the agent doing the work isn't the one grading it.

Anthropic, [Best practices for Claude Code](#)

So for this guide, five agents went through my actual setup in parallel, each flagging anything that had to stay private, then one drafted the guide and another the file for your AI, and then four reviewers attacked it at once, one each on privacy, accuracy, depth and voice. The first round came back with 64 findings, 22 of them things that had to change, the next round with 3, then 6, and whatever was still open came to me.



For the technical crowd

- **Frozen brief.** `BRIEF.md` headed "frozen <date>, changes require an explicit diff", with numbered, testable constraints, non-goals, a success bar and kill criteria.
- **Parallel research.** Every agent prompt carries the objective, the constraints, allowed sources, a timebox of about 20 minutes or 40 tool calls, and "no live experiments", because on one 20-agent study two agents started live tracking and slept in 10-minute loops until I stopped them.
- **Quote matching.** Agents save every page they read and a short script searches the downloads for each quote. On one piece 36 rows had 3 failed quotes and 2 claims in no source, and the agents' own "verified" tags caught none.
- **Adversarial review.** A fresh subagent checks whether it meets every constraint, what the strongest case is that it's wrong, which claims are only assumed, and whether the target has quietly changed.
- Every change goes in `DECISIONS.md` with what triggered it. The full procedure is stages 7 and 8 of `ai-setup.md`, and this guide went through it.

3

Hooks, and what broke before each one

Hooks are little checks that run automatically, every time, whatever the chat is doing. The rules in section 1 are just words and it doesn't always follow them, so anything that kept getting ignored, or that actually broke something, gets a hook instead.

One stops a chat downloading something with no time limit, because that once froze a chat for nine and a half hours. Then there's one that notices when a chat says it's sent, published or paid for something without having actually done anything, because it used to tell me it was carrying on with a job when nothing was running at all. Anthropic's docs describe pretty much this problem.

WHAT ANTHROPIC SAY

“ Claude stops when the work looks done. Without a check it can run, "looks done" is the only signal available

Anthropic, [Best practices for Claude Code](#)

And one stops a chat that's done real work from finishing until it's logged what it did in the shared diary from section 5, and it's done that 8 times since late September.



For the technical crowd

- **Fetch timeout** (PreToolUse on Bash). curl needs `--max-time` or `-m`, because `--connect-timeout` alone doesn't stop a stalled response, and wget needs `--timeout`. Otherwise it denies with the fix.
- **Claim tripwire** (Stop). Matches "I've sent, deployed, published or paid" in the last message, counts tool calls in the transcript and in every sub-agent's transcript, because the first version missed a send a sub-agent had done, and logs if there are none.
- **Log guard** (Stop). After 25+ tool calls with nothing logged today, it blocks once and names the file and heading.
- `permissions.deny` blocks the site builder's deploy and database tools, ad-creation tools and `Skill(update-config)`, so a chat can't loosen its own settings. The scripts are stage 3 of ai-setup.md.

4

Making it sound like me

Everyone's just using Claude or whatever for their emails, Slack messages, content and slide decks, and everyone sounds the same. Instead of a list of style rules, the model reads a file of things I actually said out loud, ums and run-ons left in, and writes by matching that, and the rules only catch the classic giveaways, like em dashes. I went this way because my rough typed post about being a dad, typos and all, got 181,680 impressions and 54 comments, while my polished posts from the same months got 0 to 2 comments.

Every post starts from my own voice note and gets tidied, not rewritten, and then it gets checked. A second agent that can only read files compares it with my real dictation and fails anything that doesn't sound like me, and a script checks the image or video for hidden data like an AI content credential, a title or the real date. Then I copy it and press send myself.



For the technical crowd

- The corpus is 39 verbatim samples and stays private. The reviewer agent runs on Sonnet with Read and Grep only, quotes each failing line against a named rule, and ends "Be blunt. Do not praise. Do not soften a FAIL."

- A Stop hook greps copy-paste blocks for 42 giveaway patterns and blocks once. It fired 9 times between 28 September and 7 October.
- `metacheck.sh` runs exiftool and fails on C2PA, AI source tags, titles, GPS and dates, and `--strip` removes them and checks again. Stage 9 of `ai-setup.md` has the reviewer prompt.

5 One memory that every chat shares

I think a big problem with AI for real work is that it forgets, and with thirteen chats each one forgot different things. So in September I built one shared memory every chat reads when it starts, a page of facts about me and my standing rules, a one-page status per project, and a daily diary every chat writes to after each chunk of work.

Before I had a proper command for writing to the diary, chats just added to the end of the file, which put the note under whatever heading happened to be last, so about one note in five in September sat under the wrong project and the chat it belonged to never saw it. And the diary loads back in automatically after every compaction, so a long chat that gets squashed down still knows what was decided.

On top is the GM, a general manager chat. I ask it what needs me and get at most twelve lines and three items, each fact stamped with when and where it came from. Every page here has a line cap, and Anthropic's engineers warn about cramming everything into the context.

WHAT ANTHROPIC SAY

“ Context, therefore, must be treated as a finite resource with diminishing marginal returns.

Anthropic, [Effective context engineering for AI agents](#), September 2025



For the technical crowd

- `~/ .claude/vault/` holds `MASTER.md` (about 200 lines max), `projects/<name>.md` (about 40 lines max), `log/YYYY-MM-DD.md` and `archive/`. Under that sits Claude Code's own memory, one fact per file, 246 notes.
- `vlog <Project> "<index line>" "<entry>"` takes a file lock and files the entry under the right heading.
- A `SessionStart` hook with no matcher fires on startup, resume, `/clear` and compaction, and prints the master page, the project page and yesterday's and today's diary filtered to this project.
- The status line shows the model and whether today's diary was written in the last 3 hours, because on 3 October the phone app quietly moved five chats from Opus to Sonnet.
- The GM's scanner is 625 lines of Python and uses no model tokens. Dates come from record timestamps, never file times, because the 04:10 restart touches every transcript. Stage 2 and appendix A of `ai-setup.md`.

The setup underneath it

6 Routines, each with its own model and a spend cap

If I do something more than once a week it becomes a routine, a plain text brief Claude runs on a schedule with nobody watching. Each one says which model it uses and the most it can spend in a single run, which is three dollars unless I set it higher. The cheap model does read-and-report jobs, the middle one anything needing judgement, and the top one only the Sunday GM runs. Every Friday at six one of them sends my phone five prompts to record over the weekend, which is where my posts come from, and it only ever writes questions.

Honestly most of my automation isn't AI at all, of the 57 jobs in my crontab on 9 October only 6 start Claude, and the rest are plain scripts, a few with one small Claude call inside. Anthropic recommend roughly the same anyway.

WHAT ANTHROPIC SAY

“ we recommend finding the simplest solution possible, and only increasing complexity when needed. This might mean not building agentic systems at all.

Erik Schluntz and Barry Zhang, Anthropic, [Building effective agents](#), December 2024

One Claude routine told to copy a file quietly skipped it, and another one kept changing a setting back after I'd raised it, twice, so since then plain scripts own every number and setting and Claude only gets the jobs that need judgement. Anything a Claude job makes for another person or the internet goes through a gate in code, not just the prompt, because an AI site builder's agent turned out to have its own publish tool and used it unprompted. OpenAI's guide to agents says something similar.

WHAT OPENAI SAY

“ Actions that are sensitive, irreversible, or have high stakes should trigger human oversight until confidence in the agent's reliability grows.

OpenAI, [A practical guide to building agents](#), 2025



For the technical crowd

```
claude -p "$(cat routines/<name>.md)" --model "$ROUTINE_MODEL" \  
--max-budget-usd "${ROUTINE_BUDGET_USD:-3}" --permission-mode acceptEdits
```

- Every crontab line sets `ROUTINE_MODEL`, because a missing one falls back to Opus. Moving routines off Opus in July cut plan usage by about 75-85%.
- In `-p` nobody can approve a prompt, so the routines' settings allow 29 commands and never `rm`, `mv`, `crontab` or `ssh`. A couple of those rules are broad, so the real stops are in code and key scope.
- If a routine ever touches anyone else's personal data it runs on an API key, not my subscription. Stage 5 of `ai-setup.md`.

7

Connected tools, and the one job each does

Fifteen tools are connected to my Claude, and another four are set up but have never been logged in, which I only noticed when I counted them for this. The ones I use every day each do one narrow thing.

- **Gmail**, my own account only, sends directly and never makes drafts, because connector drafts never showed up on my phone. **Google Calendar** for free slots.
- **PostHog** for analytics, but only one chat at a time, because which site it's looking at is shared by every chat and two chats asking together once got each other's numbers.
- **Lovable** with **GitHub** for my sites, where Claude edits the code in git and publishing is still my click.
- **Cloudflare** isn't plugged in as a tool at all. The backups, the outside monitor and email routing run from scripts with narrow tokens, which I think is safer than one key a chat could do anything with.



For the technical crowd

- Lighthouse on a cheap server gave 61-65 where PageSpeed Insights said 99, because the mobile preset's 4x CPU slowdown runs on a weak vCPU. Judge on field data.
- The site builder's database tool was removed upstream in mid-August, and every weekly check after that wrote no report while headless Claude exited 0 with prose saying the tool didn't exist. The check has to see that the output exists. Stage 10 of `ai-setup.md`.

8

Claude lives on a server now, not my laptop

My scheduled jobs used to live in the desktop app, so they only ran while the laptop was open. Anthropic's cloud routines keep going with the laptop shut, but at the time they couldn't use my files or a local database, so in July I put everything on a small Linux server from Contabo that never switches off.

Now every area of my life and work has its own Claude chat running there all the time through Claude Code's Remote Control, thirteen of them at the moment, and I can pick any of them up from my phone, my laptop or a browser and it carries on where we left off. When the server restarts, every chat comes back in the same conversation and the new links get pushed to my phone. And it's locked down, so the firewall blocks everything coming in apart from the ports for websites and Tailscale's own, and I can only get onto the server over Tailscale, a private network between my own devices, so only devices I've approved can even see it, from my phone or my laptop wherever I am, with no login for bots on the internet to try. Logins are with a key, never a password, it bans anything that keeps trying, security updates install themselves, and I've got two-factor on everything. And the chats don't need any port open, because each one connects out to Claude.



For the technical crowd

- 8 vCPU, 24 GB RAM, Ubuntu 24.04. Each chat is `claude --remote-control <Name> --resume <session-id>` in its own tmux session and systemd scope.
- The boot script runs `unset ANTHROPIC_API_KEY` so chats bill to the subscription, and pins some chats to a session id, because headless routines in the same folder kept making "newest" a routine log.
- `ufw` defaults to deny incoming and only allows 80, 443 and Tailscale's UDP port publicly, with everything else, SSH included, allowed only on `tailscale0`. Password and keyboard-interactive login are off, fail2ban and unattended-upgrades are on, and the print service is off.
- Folder trust isn't inherited by subfolders, so a chat in an untrusted folder sits at the trust prompt and a blind restart never comes up. Stage 4 of ai-setup.md.

9

Every chat has its own memory limit

Each chat has its own memory limit and all the chats together share a bigger one, so one greedy chat gets slowed or stopped on its own instead of taking the whole server with it. Then on the Saturday night in section 10, a 5.2 GB video render I'd started from one of my chats hit that chat's 6 GB limit and the chat died along with the render.

So heavy jobs don't run inside a chat any more. Renders, builds, browser tests and anything I think will use more than about a gigabyte go through a small wrapper that runs them with their own memory limit, and if one runs out of memory only that job dies and the chat gets a plain message saying why.



For the technical crowd

- Under `cgroup v2` a group over `memory.high` isn't killed, it's throttled, so the chat just looks frozen.
- Chats share a slice capped at 9.5 GB, 6 GB each. `run-heavy.sh` runs a job in its own slice with swap off, so it dies fast instead of thrashing, and `choom -n 300` undoes the protected OOM score it would inherit from the chat. Stage 4 of ai-setup.md.

10 The Saturday it all fell over

Everything on my server fell over one Saturday, and most of what protects it got built that evening. Some of my chats started showing "Can't reach your computer, it may be asleep or offline" and I asked why I was seeing that.

At 18:41 Claude Code's built-in auto-updater tried to update over the running install, failed to download the actual program, left a 500-byte stub and deleted the program 14 chats were running from. Every chat showed "Can't reach your computer" and it took a reboot. So I turned the built-in updater off, and now a script updates it at ten past four each morning, only to a release that's been out at least a day, checks it works, and only moves a chat across when that chat is idle.

After the reboot two chats were slow to show their links, the boot script wrongly decided the login had expired, and the watchdog acted on that and killed all 13 working chats, so I had to go into the terminal and talk to Claude that way to get it fixed. Now it checks every chat first before killing any.

And that's when I asked what I'd have done if Claude got broken forever on this server and I couldn't talk to it, and what about anything important from the 24 hours before my last restore point. Nothing was watching the server from outside either, so if it died nothing would tell me. That same night it built backups that leave the server and the one-pager I'd asked for on restoring things. And a check-in with Cloudflare every five minutes, and I get an email and a push on my phone if it's gone quiet for 15.



For the technical crowd

- **Updater.** `DISABLE_AUTOUPDATER=1`, then a cron installs the newest release that's been out at least 24 hours into its own folder, rejects it if the binary is too small or `--version` is wrong, flips a symlink atomically and restarts only idle chats.
- **Watchdog.** "Login expired" is treated as a claim. If every pane shows its link it deletes the flag and does nothing, and otherwise it relaunches only the missing chats.
- **Dead man's switch.** A heartbeat goes to a private R2 bucket every 5 minutes, and a Cloudflare Worker on a cron trigger alerts if it's over 15 minutes old. Email is there because the first live test hit ntfy's daily quota. Stages 4 and 6 of ai-setup.md.

11 Alerts with no personal data in them, and backups that leave the box

The alerts come through ntfy, a free app for iPhone and Android. You install it, subscribe to a long random topic name you make up, because anyone who knows the name can read it, and then any script on the server can push to your phone with one line. You can run your own ntfy server instead if you'd rather nothing went through theirs.

Every scheduled job writes a full log on the server, but the alert on my phone never quotes it, because logs can hold people's names and email addresses. In September I asked for every alert to say in plain English whether I need to act, and what and where, because the old ones read like "Routine 'srv4' exited 2". I also

get at most one an hour per job, after a crashing job failed every 30 minutes for 15 hours and used up ntfy's free daily quota.

For backups, the answer to what I'd lose if the server was gone for good was up to a day. Now every hour it encrypts the database, code, settings and Claude's memory and sends them to Cloudflare storage, and every night it does the bigger folders and all the chat history. The key is in my password manager too, and there's a recovery guide written for me to follow without Claude.



For the technical crowd

- Every cron line runs through a wrapper that classifies only this run's log into billing, auth, network or bug and sends fixed text from a small registry, never a log line.
- Sending one is a single line.

```
curl -s --max-time 10 -H "Title: Backup didn't run" -H "Priority: high" \
  -d "Last night's backup didn't run. Nothing's lost yet. Do you need to act? Yes, re-run it from
the backups chat." \
  https://ntfy.sh/<your-long-random-topic>
```

- Backups are tar, zstd and openssl AES-256 streamed to R2. The bucket has a 7-day lock, so even the box's own key can't delete or overwrite them, and the restore was tested the same night. Stages 5 and 6 of ai-setup.md.

Getting it used

12 Planning for adoption

A fair bit of what I've built I ended up switching off because I wasn't using it, and I think it's the same problem I see everywhere, people building something clever and not planning for anyone using it. So now I think about that before I build anything, and I switch things off when they're not being used.

I had a morning assistant that asked me questions every day, and I switched it off because I just didn't feel like it was being useful. It dated chats by when their files were last touched, so dead chats looked alive, and it kept no record of what I'd decided, so it asked me things I'd already answered. The GM replaced it, and that mostly only talks when I ask, in twelve lines at most.

I'd also built a nightly job to hand long chats over to fresh ones, and it didn't work once in 73 tries, so I switched it off and let the diary load back in after compaction instead.

The Friday routine sends five questions to my phone because the weekend is when I record, and I'd rather talk than type, so it never asks me to write anything.

When I had a VA she never got access to my Claude. She got one Drive folder with a plain-English playbook her own Claude read, and two log docs she kept updated, so my Claude read her progress every day instead of me chasing her, and it only pushed a new playbook when I told it to. At handover four of her five working files were in her own Drive where my Claude couldn't see them, so next time I'd put working files in the owner's folder from day one.

Where it still goes wrong for me

It still gets things wrong, in personal stuff and work stuff, and it's constant work to fight it. It's told me it couldn't reach my server while it was running on it, and it's quoted a standard that had been rewritten this year with new paragraph numbers, and it still brings up outdated facts and asks me questions that aren't relevant any more. Even writing this guide, my research notes had the workslop study at 1 hour 56 minutes per piece when the source says 1 hour 51, and the first draft said every number came from my server when some didn't, and it took checking the source and a separate reviewer to catch those. The rules and the reviewers catch most of it, probably, but not everything.

If you run a team

Everything above is me on my own, but if I was running a company again, which I did for ten years, this is roughly what I'd copy across, because I think the slop and the duplicate tools get worse with more people.

I'd start with one instructions file that everyone's AI loads, with the facts rule, the pushback rule and never sending anything without a yes in it, because people don't know what they don't know, and if it's hallucinating or inferring or guessing they might just take it as fact. It's the cheapest thing in this guide, and it's the one I'd do first. Then I'd write a framework and a blueprint for making decisions so people can ask Claude first before anything comes to me.

Then a second model reviews anything going to a customer first, and the person who asked for it still reads it before it goes. Anything that sends, publishes or spends money goes through a gate in code, not just the prompt, and every scheduled job gets its own spend cap.

Before anyone builds a tool I'd ask who's going to use it every week, and whether someone else on the team has already built it, and anything nobody's using gets switched off. If a routine isn't working, then obviously I'd probably just switch it off and keep doing the manual review. Alerts never quote a log, because logs hold customer names and email addresses, and contractors and assistants get a playbook their own AI reads rather than access to yours.

What I'd set up first

If you're not technical, I'd copy the instructions first, because you don't need a server for those. I'd paste this into the custom instructions of whatever Claude you use, word for word, and it should behave differently from the next chat.

PASTE INTO YOUR CLAUDE'S CUSTOM INSTRUCTIONS

1. Only state something as fact if you checked it in this chat (ran it, read the file, opened the page, looked at the image). Label everything else [inferred] or [assumed].
2. End anything I'll act on with "What I did NOT check".
3. Before saying something doesn't exist, check where it would live first (which inbox, which folder, which account).
4. When I reply, first decide whether I asked a question, gave you a new fact, or said you were wrong.
5. A question gets an answer, not a new plan.
6. If I say you're wrong, check before agreeing. If I'm the one who's wrong, say so and show me why.
7. Never send, post, book or pay for anything without my yes in this chat.

Then pick one job you do every week and hate, and give it a clearly defined outcome to loop until done. For me it's making content and I hate doing that, so I get five questions on a Friday and record once over the weekend, and that's the week's posts. Keep it in one chat with a notes file it reads first every time and adds to at the end, which is the cheap version of my shared memory. Talk to it instead of typing, because a two-minute voice note gives it more than you'd ever bother to write.

If you are technical, this is roughly the order I'd do it again in.



For the technical crowd

1. A small always-on Linux box on a private network, key-only SSH and no public admin surface.
2. One Remote Control chat per area of your life, each in its own capped memory scope, plus a separate area and wrapper for heavy jobs.

3. A global `~/ .claude/CLAUDE.md` with verified vs assumed, the labels and the pushback protocol.
4. A shared memory, a logging command with a file lock, and a SessionStart hook with no matcher.
5. The fetch-timeout hook.
6. Encrypted off-box backups every hour, the key also off the box, and a restore you've actually tested.
7. A heartbeat to something outside the box that pages you when it goes quiet.
8. `DISABLE_AUTOUPDATER=1` and your own updater that installs side by side.
9. Then routines, each with `--model` and `--max-budget-usd` on its line, and a cron wrapper whose alerts never quote a log.

That's the setup as it stands in October 2026, and it'll probably look different in a couple of months. The diagram of all this is on lewiscrook.com/how-i-work, and you can point your Claude at lewiscrook.com/ai-setup.md and have it set this up with you step by step.

Sources

Every quote here was checked word for word against the page it came from on 9 October 2026.

- BetterUp and Stanford, [What is AI workslop? Research on costs and solutions](#), September 2025
- Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala and Edwin Zhang, OpenAI, [Why Language Models Hallucinate](#), 4 September 2025
- Anthropic, [Reduce hallucinations](#)
- Anthropic, [Best practices for Claude Code](#)
- Erik Schluntz and Barry Zhang, Anthropic, [Building effective agents](#), 19 December 2024
- Anthropic, [Effective context engineering for AI agents](#), 29 September 2025
- OpenAI, [A practical guide to building agents](#), 2025

THANKS FOR READING

The latest version is at lewiscrook.com/guide, and the file for your own AI is at lewiscrook.com/ai-setup.md.

If you're working through any of this, message me on LinkedIn, linkedin.com/in/iamlewiscrook.