

The Map — time-saving edition

A short slice of The Operator's Map guidebook: how to find the work to hand off, the words you need, three builds that give you time back, and what to do when it breaks.

The Map is the guidebook I wrote for the people who went through The Operator's Map. This is a slice of it. Only the parts about getting your time back.

Four parts. The three builds in Part 3 are on the same page you got this from, ready to download.

If a word still doesn't land, do what I do. Paste it into your AI and say: *explain it to me like I'm in seventh grade*.

— Waz

Part 1 — Find the Work to Hand Off

Defense before offense

Defense gives you your time back and protects you. Offense spends that time on being known. Defense goes first.

If you do not get your defense straight first, you'll find yourself rushing. So start with the work that's already eating your week: the inbox, the meeting follow-ups, the month-end paperwork.

Notice what frustrates you every day

That's the job to hand off.

Decide what matters, then say it out loud and let the AI write it down. That's where we all skip the work. Nobody writes down what matters.

Then name the leak. Not "email is a mess." One stream, measured in minutes per week. Write down the rules first: who sends it, what the subject says, when it usually comes. Once you can write the rule, a machine can follow it.

Sort your data before any AI touches it: ones, twos, threes

Three tiers of data.

- **Ones, keep AI away.** The things you'd get sued over. Account numbers, Social Security numbers, card numbers, anything that identifies a real person.
- **Twos, think it through first.** Your team's working stuff.
- **Threes, safe to use.** Things you'd share anyway.

Don't rush to sort everything. Twos should be your biggest column, because you don't know everything yet. And if a tool hasn't passed your vetting, it touches threes only. Full stop.

Decide how much it's allowed to do: read, write, never

Three levels of permission. Look only. Draft or change. Never touch.

Be careful giving write access with a vague instruction. Someone in the room put it best: "Organize" could mean delete.

Keep a human in the loop

A person checks it before anything goes out. Usually that person is you.

Every build in this edition drafts, labels or logs. None of them send on their own. And don't pay an AI for a job code could do: sorting, filtering, "skip anything already in the sheet." Code does it the same way every time, for free.

Watch for tools that send on their own. Some call recorders will email your call recaps to everyone on the call. Find the setting and turn it off.

Part 2 — The Few Words You Need

Every entry does two things and stops. What it is. The mistake everyone makes with it.

Workflow A chain of blocks that does one job, start to finish. Legos for adults. *The mistake:* Five half-built ones. One finished thing beats five started.

Trigger The first block. The thing that starts the run. A new email, a time on the clock, a message from another app. *The mistake:* Two triggers on one canvas. It doesn't know where to start. One trigger per workflow.

LLM The AI brain. Claude, ChatGPT, Gemini. n8n, the automation tool, is the pipes. It moves the data. The LLM does the thinking. *The mistake:* Picking a model you know nothing about. Every model you send data to sees that data.

Agent An AI step that holds your instructions in plain English and hands the thinking to an LLM. Give it tools and it picks which one to use. I call it a brain with fingers. *The mistake:* Running it with no chat model plugged in. It won't move. There's no gas in the tank.

Prompt The plain-English instructions you give the AI. A prompt is code. Treat it that way. *The mistake:* Leaving something out. The AI fills the gap with the most common answer, not yours. My daughter brought the garbage can in, full, because nobody said to check it was empty.

Code node A block that runs a few lines of JavaScript. You describe what you want in English. The LLM writes the code. *The mistake:* Thinking you have to write it. You don't. You have to read it, a little.

Credential The saved login that lets n8n act as you inside one account. Gmail, Google Sheets, Claude. *The mistake:* Expecting a shared workflow to carry its credentials. It doesn't. They're stripped on export. You'll see red triangles until you add your own.

API key, and the two wallets Your secret pass. It proves the request is you, and it sends you the bill. Your monthly chat plan and your API credits are two different wallets. n8n spends from the second one. You preload it. *The mistake:* Not saving the key the second you make it. You will not see that key again.

Execution and token (*the two meters*) n8n charges per run, top to bottom. That's an execution. The AI company charges per token, every time the brain thinks. *The mistake:* Checking for new mail every minute. You burn runs for nothing. Every hour, or once a day, is plenty.

Webhook A doorway one app sends a message through so another app picks it up and acts. It's how a call recorder hands a transcript to n8n. *The mistake:* Leaving it open. Have it reject anything without the right secret.

Source ID The unique ID on a record that tells the system "I've seen this one before." *The mistake:* Leaving it off. Without it, you get 55 copies of the same client because they email you a lot.

Part 3 — Three Builds That Give You Time Back

Every recipe is the same four sentences. What starts it. The middle. Where it lands. What it must remember.

All three are on the same page as this guide, as ready-to-import files, each with a two-page setup guide. You'll see red triangles when you import them. That's normal. Credentials don't travel. Open each red block and pick your own login.

Before you build: four habits that save you the rebuild

- **A status before a name.** Before you even think about first name, add a status column. That's what turns a dead spreadsheet into a smart one.

- **One ID per record.** Every row gets a Source ID. That's how the system recognizes the same thing the second time.
- **One naming rule, and you don't break it.** "Subject" and "subject" are different words to a machine. I watched a build fail on a capital S.
- **Time.** Stamp when every row came in. That's how it knows what's new, and how you compare this week to last.

And start with fiction. Have the AI make up fifty fake rows in the same shape as your real data. Build against that. Connect the real thing last, when it works.

Meeting Notes → Action Items

You walk away with: every action item from every meeting, with who owns it, in one sheet.

- **What starts it:** new meeting notes landing in your Google Drive. It finds them by name.
- **The middle:** Claude reads each note and pulls out every action item and its owner. Notes it has already read get skipped.
- **Where it lands:** one Google Sheet of open action items, plus an email to you with the list grouped by person.
- **What it must remember:** what's already on the sheet, so an item that's still open never gets logged twice. Mark a row Done and it stops counting as open.

Inbox To-Do + Industry Digest

You walk away with: the email that's actually your job as a to-do list, and one industry digest a week instead of a pile of newsletters.

- **What starts it:** a new email in your inbox, checked every five minutes.
- **The middle:** senders you list never reach the AI, and attachments are never opened. Everything else gets one AI call that sorts it into three buckets: my job, industry report, or other.
- **Where it lands:** "my job" becomes a row on a To-Do sheet. Industry reports get summarized onto an Industry Log. Friday at 7am, one email to you: open to-dos on top, then the week's reports grouped by theme.
- **What it must remember:** which bucket each email went to, so Friday's digest is built from the log, not from a fresh read of your inbox.

Hours → Invoices and Chasers

You walk away with: every client's invoice drafted on the 1st, and a polite reminder drafted for anything overdue.

- **What starts it:** you log hours on an Hours tab as you work. On the 1st, it gathers last month's unbilled hours by client.

- **The middle:** hours times rate. That's plain arithmetic, so no AI touches it. Then the invoice email gets written with its line items.
- **Where it lands:** in your Drafts. You press send. The invoice gets logged and those hours get marked billed.
- **What it must remember:** what's been billed and what's been paid. Weekdays at 9am, anything sent and past due that hasn't had a reminder in seven days gets one drafted. The first one is gentle. After two weeks overdue, it says "following up again." Nothing goes to a client until you've looked at it.

One rule for all three. Build one case first, the hardest one with the most controls, then copy it. Start on one inbox, or one client, before you go near the whole thing.

Part 4 — When It Breaks

It will break. Mine did, in front of the room, most weeks. That's the job.

It's like Jay-Z said. I did it wrong, so you don't have to.

Three rules before the table.

1. **Red means stop.** A red block is where the data stopped flowing. Start there, not at the top.
2. **Two minutes, then stop.** If you've stared at it for two minutes, screenshot the error and the data coming in. Send both to your AI and say what you expected to see.
3. **Pin it.** Once a step works, pin its output. Then you can fix the next step without pulling fresh data every time.

It won't start, or nothing shows up

What you see	Why	Fix
Red triangles all over a workflow you imported	Credentials are stripped when a workflow is shared.	Open each red block and pick your own login. Swap anything that was mine, like a label name.
Claude key "couldn't connect," or "not scoped to a workspace"	The key was made under the wrong workspace, or there are no API credits on it.	Make a new key under Default workspace . Preload credits. Your monthly plan doesn't cover this.
Your test email never shows up	The trigger only reads unread mail, and you opened it.	Send yourself a fresh one. Or turn off the unread filter while testing.
A weekly run only caught 10 emails	"Max per poll" defaults to 10.	Raise it. 50 is the top.

Turn on Always Output Data.

What you see	Why	Fix
Everything after a Filter goes quiet	The filter found nothing, so it passed nothing.	
The agent does nothing	No chat model plugged in, or the chat model has no login.	Plug in the model. Attach your key.

It runs, but the answer is wrong or something feels off

What you see	Why	Fix
Code says a field isn't there, but you can see it	Capitals or spaces don't match. <code>subject</code> won't find <code>Subject</code> .	Make the column name and the code match exactly.
Code from your AI fails on real data	It guessed what your data looks like.	Screenshot the error and the data coming in. Send both back.
The same thing gets logged again and again	Nothing checks what's already there.	Read the sheet first. Give every row a Source ID.
Your report arrives as ugly plain text	Wrong field mapped, or no formatting.	Map the agent's output . Ask your AI for HTML.
Your execution count is racing up	It's checking every minute.	Every hour or once a day is plenty.
Your AI bill is climbing	You're paying a model to do what code could do.	Filter with Code first. Only send the AI what needs judgment.
You're thinking about adding a Delete step	Don't.	Tag it, park it, review it monthly. n8n shouldn't delete anything.

Where the Full Map Goes Deeper

This is a slice. The full Map is the guidebook the people in the program build from, and it goes a lot further than this.

It has every word I said in the room, not just these eleven. Every n8n block we used, and how to read a Code node without writing one. The tool stack, and which tier of your data each tool may touch. All eight builds as recipes, plus eight more from the capstone round. How to find the twenty people you want to know. And the full break table, with the one-pagers behind it.

But the guidebook isn't the program. The program is the live builds. Every week we build one together, on real tools. It breaks in front of everyone, and we fix it in front of everyone. Then office

hours, where your build is the one on the screen. And a room of operators doing the same thing you are, which is where most of the good ideas came from.

The test at the end is simple. Can it run without you?

If this slice saved you some time and you want the rest, join the Cohort 02 interest list at **theoperatorsmap.com/inside**. I'll let you know when the next cohort opens.

— Waz