Trained on what was.

The tools you'll use today learned from everything people already made. They don't know what you're about to make. That gap is your job.

A record nobody could hear

In June 2001, Reprise Records turned down Wilco's *Yankee Hotel Foxtrot*. The band streamed it free on their website on September 29, 2001, and more than 50,000 people came to listen, about eight times their normal traffic. Nonesuch signed it that November and released it on April 23, 2002.

The title comes from a shortwave numbers station: a voice reading "Yankee… Hotel… Foxtrot" out of the static, recorded for the Conet Project. You hear it at the end of "Poor Places."

Jay's reading: it's a record about a new condition, more noise and less signal, and the people judging it couldn't hear it because their ears were trained on what had sold before.

A console nobody had built for

Jay's handheld console has a knob, two buttons and a touchscreen. You talk to it; a game appears. The models reached for what they'd seen: old arcade games. Of the 22 games on the shelf in September, **16 ignored the knob**. That was the tell. Jay played all 22, and **none of them were fun**.

Browser games it has seen by the pile, already sorted by people who played them, so its average is already fun. Knob games: none, and nobody had sorted them yet. Average lives in the past. New lives in the future.

So Jay kept the fun ones, each with a written *why*, and wrote a taste file:

```
Knob-first games.
Alive on the first frame.
Sound on everything. Silence reads as broken.
```

Those kept games went back in as examples. His kids playtested them and ranked them by favorites. Nobody could compute that. Somebody had to play them.

Why the gap exists

FROM YOUR AI FLUENCY SHEET	WHAT IT MEANS FOR YOU
Pre-training	It learned from text people already wrote. It's very good at what was.
Knowledge cutoff	Its picture of the world stops at a date. Your business didn't.
Hallucination	When it doesn't know, it can still sound sure. You're the check.

The point. Judgment, taste and wisdom matter more now, not less. The first filter, before you point AI at anything: would I know a good answer if I saw one? If not, learn enough to judge before you hand it over.

The loop you'll practice all day

YOU	WHAT IT LOOKS LIKE	THE 4D
Describe	Say what you want and what done looks like	Description
Watch	Read the plan and each prompt before saying yes	Delegation
Check	Ask where a number came from. Check one by hand.	Discernment
Redirect	Say what's wrong in one plain sentence. Keep or toss.	Diligence

Rules of the circle

1 · Practice folder, fake data. All day. Your own notes at 3:50.	**2 · Short rounds.** The clock is on the slide. When it beeps, look up.	**3 · Partner.** Reads prompts with you.	**4 · Check before you trust.** End every problem with one thing you checked.

Sources: Wikipedia, "Yankee Hotel Foxtrot" · Anthropic, AI Fluency cheat sheet and framework (Dakan, Feller & Anthropic, CC BY-NC-SA 4.0); 4D mapping is Jay's · Console numbers: Jay's notes, Sept–Oct 2026.

SKILL CARD · BOTH DAYS

TICK EACH ONE WHEN YOU HAVE *DONE* IT, NOT DESCRIBED IT

NAME _____

PARTNER _____

FRIDAY · YOUR OWN WORK

01	Read a permission prompt out loud before saying yes	9:00	☐
02	Redirect it mid-task in one plain sentence	9:00	☐
03	Write a brief your partner can say back	10:00	☐
04	Plan before building, and cut the plan smaller	10:00	☐
05	Catch a made-up fact	11:00	☐
06	Make it show where a number came from	11:00	☐
07	Sort data green / yellow / red	11:00	☐
08	Write a CLAUDE.md and see the difference	1:00	☐
09	Make a skill you can run with one command	2:00	☐
10	Ship v1: the brief is in your inbox	3:15	☐

SATURDAY · YOUR ORGANIZATION

11	Map a process and name its owner	11:00	☐
12	Route each step to frontier, open, local or a person	11:00	☐
13	Write a lane card for one team's agent	1:00	☐
14	Survive the lane test	1:00	☐
15	Name who owns the output and who signs off	1:00	☐
16	Commit to one build by Nov 15	2:50	☐

BRIEF CARD · 10:00 FILL THIS IN BEFORE YOU TYPE

GOAL what this is for, in one line

WHAT I'LL GIVE IT files, folders, examples

WHAT I WANT BACK file type, length, format

DONE LOOKS LIKE how I'll know it worked

DON'T the thing it must never do

PARTNER'S ONE QUESTION

BRIEF CARD · SPARE FOR ROUND 2 OR TONIGHT

GOAL what this is for, in one line

WHAT I'LL GIVE IT files, folders, examples

WHAT I WANT BACK file type, length, format

DONE LOOKS LIKE how I'll know it worked

DON'T the thing it must never do

PARTNER'S ONE QUESTION

HOMEWORK CARD · OVER DINNER · BRING IT AT 8:30 SATURDAY

FEEDS JEFF'S 10:15 DISCUSSION AND JAY'S 11:00 MAPPING

ONE PROCESS AT YOUR ORGANIZATION THAT WASTES PEOPLE'S TIME

ITS STEPS, IN ORDER who does what, roughly

WHO OWNS IT a name or a title

HOW OFTEN IT RUNS daily, weekly, monthly

HOURS IT EATS PER MONTH a guess is fine; write GUESS

WHAT DATA DOES IT TOUCH? clients, money, people, public stuff

Claude Code, on one page.

You type in plain English. It reads, writes and runs things in the folder you started it in. You stay in charge by reading before you say yes.

Start, stop, undo

`claude`	Type it in Terminal or PowerShell, inside the kit folder. Or double-click **Start Claude Here** in that folder; it opens a terminal already inside.
`Shift` + `Tab`	Plan mode on and off. In plan mode it reads and proposes; it doesn't change anything.
`Ctrl` + `C`	Stop what it's doing right now.
`Esc` `Esc`	On an empty prompt: go back to an earlier point.
`/rewind`	Undo. Restores files and/or the conversation.
`/clear`	Fresh start. Use it between jobs, and after two failed fixes.

It runs in auto mode. It acts without asking for most things. When it does stop to ask, read the prompt out loud to your partner before you say yes.

Typing tricks

`Shift` + `Enter` or `\` then `Enter`	New line without sending.
`@`	Point at a file: `@receipts.pdf`
`!`	Run a command yourself: `!ls`
`Ctrl` + `R` or `↑`	Find something you typed earlier.

Slash commands worth knowing

`/memory`	Edit your CLAUDE.md by hand (this afternoon).
`/skills`	List your skills (2:00 today).
`/help`	Everything it can do, always current.

How to talk to it

WHEN YOU WANT…	SAY SOMETHING LIKE
A plan first	"Before you change anything, tell me your plan in five steps."
Questions first	"Ask me anything you need to know first, one question at a time."
A smaller version	"What's the smallest version that would still help me?"
Proof	"Where did that number come from? Quote the line."
Its doubts	"What did you assume? What would make this wrong?"
A fix	"That's wrong because ___. Fix only that."
Safety	"Don't delete anything. Make a copy first."
It to remember	"Remember that I always want ___." or `/memory`

Where things live

`ai-workshop-kit/`	The kit folder, wherever you unzipped it. Everything today lives inside it.
`CLAUDE.md`	Notes it reads every time it starts in that folder.
`~/.claude/CLAUDE.md`	Notes it reads everywhere, for you.
`~/.claude/skills/`	Your skills. Each is a folder with a `SKILL.md`.
`~/.claude/projects/`	Its record of your conversations, kept 30 days on your laptop.

What our account promises

You're signed in through the Connolly Center's commercial account. Under commercial terms Anthropic doesn't train on your prompts or files, and keeps them 30 days. The free and personal chat apps work differently. Still: no client secrets, no passwords, no Social Security numbers.

Checked against code.claude.com docs (interactive mode, commands, memory, skills, permission modes, data usage) on Oct 13, 2026. Versions change fast; `/help` is always current.

A site about you. Then a folder to clean.

One project all day. It starts with a website about you, and every step after this adds a page to it. First, a short demo from Jake, who spent the term building for a whole team.

Before you start: open your Terminal or PowerShell and type `claude`, inside the `ai-workshop-kit` folder (or double-click **Start Claude Here** in that folder; it opens a terminal already inside it). Stay in this folder all day: your site, the practice folder, your rules and your skills all live here.

8:55 · A WEBSITE ABOUT YOU

Your first app.

```
Make me a one-page website about me, in a folder called site.
Ask me five questions first. Then build it and open it.
```

In place of "me," give it your name and a little about who you are. If and when it asks for permission, **read the prompt out loud to your partner** before you say yes.

Then redirect it twice, just by talking: *"warmer colors," "add a section for what I'm proud of."*

Notice: that was the loop. You described, you watched, you checked a prompt, you redirected. Everything today is that, bigger.

HOW TO READ A PERMISSION PROMPT

1. **What** does it want to do? Make a file, change one, run a command?
2. **Where?** Inside your workshop folder, or somewhere else?
3. **Would you let a new hire do it** on their first day?

You can say yes once, yes for the session, or no and tell it what to do instead. "No, do this instead" is the best redirect there is. You'll turn the prompts off eventually. Not before you can read one without thinking.

9:25 · CLEAN THE FOLDER

practice/ · a consultant's working folder, everything invented.

Dana Kessler runs Kessler Advisory, a one-person consulting shop in Roanoke. This is her folder after a busy fall: copies of copies, shorthand notes, a couple of things that don't agree.

```
Look in practice. Sort it into folders that make sense. Delete nothing.
Show me your plan before you move anything, then list every move.
Tell me anything that looks wrong, duplicated or out of date.
```

This folder is yours for the rest of the day. Every step from here builds on it.

Gallery lap · 9:38 · 5 min. Read the next table's folder plan. Same folder, same prompt. What did theirs catch that yours missed?

Skill Card: 01 read a permission prompt · 02 redirect mid-task.

Your two redirects

Write the exact sentences you used. Which one worked the first time?

One prompt, three questions

Pick one permission prompt you read out loud. What did it want to do, where, and would a new hire get a yes?

What looked wrong in the folder?

What did it flag as duplicated or out of date? Write the rule you'd give it next time, in one sentence.

Steal one idea

On the gallery lap, write down one thing their folder plan did that yours should.

The morning rhythm (show a little, then you build a little) is borrowed from Jay's Playgramming course at DePaul (GAM 240). Kessler Advisory and its clients are invented; the cities and industries are real.

Put the folder on your site.

One page on your site that sums up the practice folder: clients, money in, open items, what looks wrong. The brief is where you decide what "summarize" means.

> **The AI "does not know which ones are important to you here and now."**

Kent Beck, "Taming the Genie," on constraints. His point: a persona changes the style you get; an explicit constraint changes the structure.

Five lines, for your page

Same folder for everyone, so the briefs differ in what you decide matters.

LINE	QUESTIONS TO ANSWER FOR YOUR PAGE	WHY IT MATTERS
Goal	Who's it for: you on Monday, or a client? What should it let them do?	Guides small choices.
What I'll give it	Which files in `practice/`? All, or the CSVs and notes? Which invoices file?	Point at the real files.
What I want back	A new page in `site/`. A client table? Totals on top? Open items?	Format is a decision.
Done looks like	A number you can check by hand, like September's invoices.	How you'll check it.
Don't	Don't change `practice/`? Don't pick when numbers disagree?	At least one. People skip it.

Three moves

WRITE THE BRIEF

Write the brief on your card.

Pen, not laptop. Five lines, plain English. If a line is blank, that's the thing you haven't decided yet. Then your partner reads it and asks **one** question. Fix the brief so it doesn't need asking.

PLAN, DON'T BUILD

Plan, don't build.

Shift+Tab to **plan** mode. Type your brief, then add:

```
Ask me anything you need first. Then give me a plan.
Don't build anything yet.
```

Read the plan. Cut it to the **smallest version that would still help you**.

BUILD IT

Now build it: the folder, on your site.

Shift+Tab out of plan mode, then: `Go ahead with the plan.` Partner check at 10 minutes.

The cut test

Ask it: **"What's the smallest version of this plan that would still help me on Monday?"** Then cross out anything that fails all three:

- ☐ I'd use it in the first week.
- ☐ I can check whether it's right.
- ☐ It doesn't need a system I don't control (logins, other people's databases).

A good first version: one table of clients with what each was invoiced, and nothing else. Open items and "what looks wrong" come next, after you've checked two numbers by hand.

> **Practice data all day.** The folder is invented, so nothing real goes to the model. Your own files come Monday, after you've sorted them with the traffic light at 11:42. **Skill Card:** 03 write a brief · 04 plan before building.

Your first version

After the cut test, write what you're actually building in the next 20 minutes. One sentence.

At the 10-minute partner check

What's on screen? What did your partner ask?

What the plan got wrong

Two volunteers share this at the end. What did its plan assume that wasn't true about the folder?

Save for later

What you cut today that you want back. Later today: an Ask page, then this whole page on one command.

Sources: Kent Beck, "Taming the Genie like Kent Beck," newsletter.kentbeck.com · Claude Code best practices (explore, then plan, then build), code.claude.com/docs/en/best-practices. Kessler Advisory and its clients are invented.

Ask the folder. Prove it.

It's fluent whether it's right or wrong. Fluent and wrong is the expensive combination, so you learn to ask for proof.

The habit

1. **Ask for the receipt.** "Where exactly does that number come from? Quote the row." beats "are you sure?"

Why you check

It sounds just as sure of itself whether it's right or wrong. Confidence isn't evidence. The file is.

11:08 · ASK THE FOLDER, WITH RECEIPTS

Four questions it can only answer from practice/.

```
Using only what's in practice: 1) How much did I invoice in September?
2) Which client's notes disagree with its contract?
3) Which client email never got a reply? 4) What's on my calendar next week?
For every answer, quote the file and the line it came from.
Tell me what you're not sure about.
```

Check every answer against the file before you believe it. If one is wrong, that's the lesson.

The wrong answer	What the file says

11:27 · A WAY TO TALK TO THE FOLDER

The Ask page.

```
Add an "Ask" page to my site: a search box that searches every file in
practice and shows the matching lines with the file name next to each one.
No server, just a page I can open. Then add the four answers from before,
each with its source, as the examples on that page.
```

Try a question it can't answer. What does the page do? What should it do?

11:42 · TRAFFIC LIGHT

Would you put this in?

Green = public or made-up: your website, a published report, the practice folder. **Yellow** = internal, not sensitive: meeting notes, drafts, your own calendar; commercial accounts only. **Red** = client data, health, Social Security numbers, passwords, anything under NDA; not today, maybe not ever. Mark each one, then argue with your partner about two of them.

DATA	G	Y	R	DATA	G	Y	R
1 · Your company's public web page	☐	☐	☐	7 · A patient's visit notes	☐	☐	☐
2 · Minutes from a public board meeting	☐	☐	☐	8 · An industry report you paid for	☐	☐	☐
3 · A client's Social Security number	☐	☐	☐	9 · Your customer list with emails	☐	☐	☐
4 · Your own calendar for next week	☐	☐	☐	10 · Your notes on a book you read	☐	☐	☐
5 · A press release before it goes out	☐	☐	☐	11 · The office Wi-Fi and bank passwords	☐	☐	☐
6 · Everyone's salaries	☐	☐	☐	12 · A contract with a confidentiality clause	☐	☐	☐

OUR WORKSHOP ACCOUNT (COMMERCIAL TERMS)	EVERYTHING ELSE
Anthropic doesn't train on your prompts or files unless you opt in. Kept 30 days. Your laptop keeps its own copy of conversations for 30 days in `~/.claude/projects/`.	The free and personal chat apps are a different contract, and other companies\' apps have their own rules. More on Saturday.

How your words travel

- What it learned from (training) stopped at a date. What you type is read fresh each time and isn't added to its training on our workshop accounts.
- Your prompt goes from your laptop to Anthropic's servers, encrypted, and the answer comes back.
- A copy of each session also sits on your laptop for 30 days.

Source: Claude Code data-usage docs, code.claude.com/docs/en/data-usage

Saturday adds a fourth option for red data: a model that runs on a machine you own, with the Wi-Fi off. **Skill Card:** 05 catch a made-up fact · 06 make it show its sources · 07 sort data by traffic light.

11:55 · Before lunch.

```
Write a file called where-i-stopped.md: what we built,
what works, what doesn't, and what I should do first after lunch.
```

Then: `Remember three things about me that you should know every time we work together.` It writes them into a file it reads on every start. That's memory, part one.

Teach it. It forgets, and it follows rules.

Every new conversation starts from zero. CLAUDE.md is the note it reads first, every time: your rules and your memory in one file. Then the outside world comes in.

1:00 · WHERE WERE WE?

Start fresh. Open claude.

```
Read where-i-stopped.md and tell me where we are.
```

The file you wrote before lunch is the only memory it has of this morning.

1:12 · YOUR RULES

CLAUDE.md for this folder.

```
Write a CLAUDE.md for this folder. Interview me first: who I am, what my
site is for, how I like things written. Then add three rules it must never
break: never invent a number, never delete a file, never send anything
anywhere without asking me.
```

Read what it wrote. Cut anything that isn't true. Then: `/clear` and ask it who you are.

Notice: the three nevers are the ones the rest of the day leans on. No invented numbers (the web step). No deleting (this morning's rule, now written down). No sending without asking (the last step).

1:20 · BEFORE AND AFTER

Same words, with your rules and without.

```
/clear
Write a short welcome paragraph for the top of my site.
```

Compare it with the one it wrote this morning. Partners blind-pick the better one. Then find one thing you'd still fix, and fix it the lasting way (see the correction habit, page 2).

1:30 · PULL THE OUTSIDE WORLD IN

Enrich the folder from the web.

```
For each client in clients.csv, find two public facts from the
web about its city or its industry: one local, one about the industry.
Give the URL for each. Add an "In context" line to each client on my
site, and mark every fact UNVERIFIED until I say otherwise.
```

Then check **one** fact by hand, in the browser. Tell it: *"verified"* or *"wrong, fix it."* Clients are invented; cities and industries are real. That's why this step is safe today.

A real-looking example

Dave is invented. Notice how specific it is, and how short.

Where it lives

```
# About me
Dave Hollins, partner at a 12-person commercial real estate
brokerage in Richmond. Mostly office and light industrial.
I'm not technical. Explain steps in plain English.

# Who reads my work
- Clients: owners and CFOs. Busy. Want the number first.
- My two associates, Priya and Tom. They do the follow-up.
- Our managing partner, who wants one page, never more.

# How I like things
- Lead with the answer, then the reasons.
- Dollars rounded to the nearest thousand: $1.2M, $85K.
- Square feet with commas: 14,500 SF.
- Dates like Oct 16, 2026.
- Tables over paragraphs when comparing properties.
- Emails under 150 words. No "I hope this finds you well."
- Sign emails "Dave."

# My files
- Comps live in ~/Documents/comps as CSVs, one per quarter.
- Never edit a file in ~/Documents/comps. Work on a copy.

# Checking
- When you use a number from a file, say which file and row.
- If two sources disagree, show me both. Don't pick.
- If you're guessing, say GUESS next to it.

# Never
- Never send anything. Draft it; I send it.
- Never put a client's name in a file name.
```

26 lines. Every line is something a new hire would need on day one.

`./CLAUDE.md`	This folder only. Today's file.
`~/.claude/CLAUDE.md`	Everywhere you start Claude. Move the "about me" part here once you like it.

When the two disagree, the one closer to your folder wins. Anthropic suggests keeping these files short, under about 200 lines; under 40 is better for a start.

What makes a good line

VAGUE (CUT IT)	SPECIFIC (KEEP IT)
Be professional.	Emails under 150 words. Sign them "Dave."
Be accurate.	When you use a number from a file, say which file and row.
Be careful with files.	Never edit anything in ~/Documents/comps. Work on a copy.

THE CORRECTION HABIT · ALL AFTERNOON

Every mistake becomes a line.

> Add a line to my CLAUDE.md so this doesn't happen again.

A good CLAUDE.md grows one correction at a time.

Lines you added today

Skill Card: 08 write a CLAUDE.md.

Sources: Claude Code memory docs, code.claude.com/docs/en/memory · Claude Code best practices, code.claude.com/docs/en/best-practices ("keep CLAUDE.md short"). Dave and his brokerage are invented.

Anything you'll do twice, make it a command.

CLAUDE.md is who you are. A skill is a recipe with a name on it: one job, done the same way every time, on one command.

2:10 · TRANSFORM IT WITH A SKILL

/brief

```
Make me a skill called brief. When I run it, it reads everything in
practice and writes a one-page Monday brief: money in this month, open
items with who owes what, anything that looks wrong, and three things to
do first. Save it as brief.md, add it to my site as a page, and open it.
```

Then run it: `/brief`. Then change one file in `practice/` and run it again.

What you changed in practice/

What changed in the brief

What a skill is

A folder with one file in it, `SKILL.md`. The top says its name and when to use it. The rest is the recipe: steps, format, and how to check the result. Yours live in `~/.claude/skills/`, for example `~/.claude/skills/monthly-update/SKILL.md`. Run one by typing `/monthly-update`. Or just ask for the thing, and it notices.

Did the "looks wrong" part of your brief catch this morning's traps? If it invented a number, the fix is a line in the skill, not a complaint.

Demo · a playbook becomes a skill. One page of "what we insist on" turned into `/contract-check`. Watch for what the contract mentions that isn't in the folder. In your kit: `field-demos/contract-check`.

What a SKILL.md looks like

An invented example. Yours will be different.

```
---
name: brief
description: Reads everything in practice and writes a
  one-page Monday brief. Use when I ask for the brief
  or "what's going on in the folder."
---

1. Read every file in practice/. Use nothing else.
   Never delete a file.
2. Write brief.md with four headings: Money in this
   month · Open items · Looks wrong · Do first.
3. Under Open items, say who owes what to whom.
4. Every number gets the file and line it came from.
5. If two files disagree, show both. Don't pick.
6. Three "Do first" items, no more. One page.
7. Add it to the site as a page and open it.
```

Steps 4 and 5 are mistakes you already caught this morning. Every mistake you've seen earns a line.

2:25 · A WALKTHROUGH THAT TALKS

Narrate it.

```
Add a "Walkthrough" page to my site with a Play button. It should step
through every page on the site and read a one-sentence explanation of
each out loud, using the browser's built-in voice. If ffmpeg is installed
on this laptop, also export the same walkthrough as walkthrough.mp4.
```

Press play. Then: *"the voice is too fast,"* *"skip the Ask page."* Most laptops won't have ffmpeg. That's fine; the page is the walkthrough.

2:40 · SWAP LAPTOPS

Your partner runs your /brief without your help.

Sit on your hands. Every time they hesitate, ask a question, or it does something odd, write a row. The last column is the line you'll add to SKILL.md.

WHERE THEY GOT CONFUSED	WHAT IT DID	LINE TO ADD TO SKILL.MD

Why bother? Saturday's question is "could a colleague run it?" This page is your first answer. Keep it.

Skill Card: 09 make a skill. Done when the second /brief run shows the change.

Skills you want next

Chores you'd turn into skills next week, once you're home.

Sources: Claude Code skills docs, code.claude.com/docs/en/skills (file layout, name and description frontmatter, running as /name)

Send it. Then your own notes. Then Monday.

The brief leaves your laptop. Then your own notes become Monday's first build. Fifteen minutes, then twenty more.

3:15 · SEND IT TO YOURSELF

The first time it reaches outside the folder.

```
Send me brief.md. Open a draft in my mail app addressed to me with the
brief as the body. On a Mac, also text it to my phone through Messages.
If Google Drive is on this laptop, put a copy in it. Ask me before each one.
```

Three prompts, maybe four. **Read every one out loud** before you say yes. Any one of the three counts.

☐ The brief is in your inbox. ☐ You read every prompt out loud. Both ticked: that is **10 · ship v1**.

3:50 · YOUR OWN NOTES

The last step is you.

```
Here's a photo of the notes I took today. Read them. Tell me what I
learned, what I'm still unsure about, and propose three things I could
build on Monday for my real work. Then build the first piece of the
first one, here, as a new page on my site.
```

Photograph your notes. AirDrop or email the photo into this folder. The handwriting doesn't have to be good. Of the three ideas, build the one where you'd know a good answer if you saw one.

Alright. What does this mean for you?

1. **Pick a project.** The second thing on your Skill Card. Monday, rough, in a folder.
2. **Teach it who you are.** A CLAUDE.md, a skill, and a correction line every time it's wrong.
3. **Make a culture of making and risk taking.** Start with the homework card below.

Show-and-tell · 4:10

Ninety seconds each. **Open your site.** Practice on your partner first.

BEAT	SAY
1 · What it was this morning	The one-page site you made before the folder showed up.
2 · What it does now	Click through it. Don't explain the code.
3 · What surprised you	Something it got wrong, or something it did that you didn't expect.

Homework, over dinner

Fill in the **homework card**: one process at your organization that wastes people's time, its steps, how often it runs, and who owns it. A name, not a department.

Bring it at 8:30. It goes two places:

- **10:15, Jeff's discussion.** Where AI is showing up at work, where it's stalled, and why.
- **11:00, Jay's routing.** Map it step by step, decide which steps go to which kind of model, put a rough monthly cost on it.