A PRACTICAL GUIDE TO LEARNING AI

Start here.

A few things I wish I had understood sooner.

By Tim Dillard · First edition

I started with ChatGPT for emails and everyday questions. Eventually I was building automations and trying to connect whole systems. I learned a lot. I also built things I did not need.

If you have asked, "Where do I even start?" this is for you. These are the lessons I would put in your hands first.

Pick one real task. Try a lesson. See what happens. Come back when the next problem shows up.

Tim

YOUR FIRST MOVE

Open the AI tool you already use. Pick one task from this week that took more effort than it should have. Start there.

THE PATH

6 stages · 24 short lessons

01 **Start**	02 **Explain**	03 **Save**	04 **Try**	05 **Repeat**	06 **Refine**

Read the first stage and try it today. The rest can wait until you need it.

START

Start with something real.

01

You do not need to understand all of AI. You need one place where a little help would matter.

Define the problem together.

You do not have to know the answer before you ask for help. Describe what is happening and what you wish were different. Let AI help you work out the problem before either of you decides what to build. “I keep losing track of follow-ups” is a better starting point than “build me an operating system.”

Make the first win small.

Pick one recurring task: preparing for a meeting, sorting rough notes, or drafting a reply. A useful result today teaches you more than a huge plan you never use. You can stop at a simple solution if it does the job.

Decide what better looks like.

Before you begin, choose one way to judge the result. Less time rewriting? Fewer forgotten actions? A clearer decision? If you cannot say what should improve, it is easy to mistake a polished answer for progress.

Keep ownership of the direction.

AI is very good at suggesting the next thing. That does not make it your next thing. I let both ChatGPT and Claude lead me into building more than I needed. Come back to your question: does this help with the problem I actually came here to solve?

TRY THIS

Pick something you did this week that took more effort than it should have. Describe it in a few sentences. Do not ask for an app, an agent, or a full system yet.

A starting prompt

I want help with one real problem: [describe it]. Right now I handle it by [explain]. I would like [what should improve]. Help me check whether I am naming the right problem. Ask one useful question at a time. Do not design a system yet.

↳ *Once the problem is clearer, the next job is helping AI understand your version of it.*

EXPLAIN

Give it what only you know.

02

The useful details are often the ones you leave out because they seem obvious to you.

Talk it through like you would with a new hire.

Voice helped me get more of my thinking out than a blank page did. Explain what you do, what goes wrong, and how you make the call. Type if that works better for you. The point is to get the real context out, not to find a perfect prompt.

Show what good looks like.

Give it an example you like and explain why. For an email, that might mean your usual tone, the relationship, and what the reader needs to do next. Add an example that misses the mark if you have one. Examples make words like “professional” much less vague.

Ask for pushback, then check it.

Tell AI to find weak assumptions and missing information. When something sounds wrong, say why. But do not accept an answer just because it agrees with you, or because it argues confidently. Check important claims against the source or what actually happened.

Get the judgment out of your head.

If you want help making a repeated decision, explain how you make it now. What matters? What disqualifies an option? What changes your mind? Turn those answers into a short checklist and try it on a few examples before asking AI to apply it broadly.

TRY THIS

Explain your task out loud for two minutes, or write a rough paragraph. Add one good example. Ask AI to summarize what it understands, then correct the summary.

A starting prompt

Here is the context and an example: [add them]. Tell me what you understand about the task, what good looks like, and what you are still assuming. Keep it short so I can correct it.

↳ *When you finally get the explanation right, save it. You should not have to start over next time.*

SAVE

Make the work survive the chat.

03

A useful conversation becomes more valuable when you can find it, reuse it, and correct it later.

Save the useful part outside the conversation.

I had work buried in chats that I could not find when I needed it. Save the final brief, decision, or instructions as a file in a place you recognize. A clearly named document is enough to start. You do not need an elaborate knowledge system.

Give each important thing one main home.

If three documents all claim to be current, you have three opportunities to use the wrong one. Choose the main copy and label older versions. Keep your reusable instructions separate from this week's task list. They change for different reasons.

Leave a handoff for your future self.

At the end of a useful session, save what you decided, why, what remains uncertain, and the next action. A summary that only says what you discussed is not enough. The next session needs to know where the work stands.

Decide what you are comfortable sharing.

Start with examples you are allowed to use. Leave passwords and access keys out of chat. Check the rules before adding somebody else's private information or workplace data. An invented example can help you learn the workflow without exposing the real record.

TRY THIS

Save a one-page task brief with a clear name and date. Start a fresh conversation with that brief and see whether you can continue without explaining everything again.

A starting prompt

Create a short handoff I can save. Include the task, the important context, decisions and reasons, open questions, and the next action. Separate confirmed facts from assumptions.

↳ *Now you have something reusable. The next step is finding the simplest way to put it to work.*

TRY

Try the simple version first.

04

Building became easier. Learning when to build became more important.

Check what you already have.

Before adding a tool, look at the features and services you already use. The right answer may be a template, a setting, a smaller process, or help from the person who owns the system. Recommending an existing solution is still useful work.

Remove unnecessary work before automating it.

When a task feels heavy, ask why it exists and what creates it. Could you collect the information once instead of retyping it? Could clearer instructions prevent the cleanup? Sometimes the best automation is a step you no longer need.

Prove it by hand before putting it on repeat.

Run the process on one real example while you are there to inspect it. Correct what fails. Then try a different example, including an awkward one. A repeatable process gives you something worth automating. An untested idea gives you something to learn from first.

Choose tools by the job, not by loyalty.

I have used different AI tools as my needs changed. You do not need to copy my whole setup. Start with one you can access and learn it on your task. Switch when you can name a real gap, not just because another tool is getting attention.

TRY THIS

Use the same task brief on a straightforward example and a messy one. Note where each result needs correction. Decide whether the process is useful enough to repeat.

A starting prompt

Before we build anything, what is the simplest way to handle this with what I already use: [list tools or methods]? Consider removing steps, using an existing feature, or doing a small manual test. Explain the tradeoff and recommend one next move.

↳ *One good result is encouraging. Repeated, dependable results are a different test.*

REPEAT

Make useful work repeatable.

05

This is where I had to learn the difference between describing a system and having one that did the work.

Written down is not the same as working.

A set of agent instructions tells AI how to do a job when it is used. It does not prove anything is running on a schedule. I learned to distinguish a plan, a manual run, and an actual repeated workflow. Ask what starts the work and watch it happen.

Follow the result all the way through.

Check the place where you actually use the output, not just the place where it was created. Did the note land in the right document? Is it current? Can you act on it? I wasted time changing local work without first finding the process that was producing the real result.

Connect the handoffs, not just the tools.

One tool may find information and another may help you use it. The important part is what passes between them and who takes the next action. Add a coordinator only when those handoffs need one. A bigger collection of agents is not automatically a better system.

Keep a person in charge of the exceptions.

Decide what can happen automatically, what needs your review, and what should stop when information is missing. For messages, that might mean letting AI prepare a draft while you approve the content and recipient. Make those boundaries clear before turning on repetition.

TRY THIS

Sketch one workflow in plain words: what starts it, what information it uses, what it produces, where it goes, and who checks it. Circle any handoff you cannot explain yet.

A starting prompt

Map this workflow from beginning to end in plain language. Distinguish what already works from what is only planned. Show where a person needs to review it and what should happen if a step fails.

↳ *A workflow can function perfectly and still be too much trouble to use. That is the next thing to check.*

REFINE

Keep what helps. Learn from the rest. 06

The goal is a better way to do your work. The system should earn the time you give it.

Make the solution fit your day.

If you are already overwhelmed, a complicated plan is a poor fix. Start where you naturally do the work. Keep the next action small and the useful result visible. Less friction matters more than an impressive list of features.

Direct the output, including how it looks.

Tell AI who will read it, what they need to notice, and how it should sound. I learned that generic language and generic design could weaken good work. Read it as the person receiving it. Would you understand it, trust it, and use it?

Record the correction, not just the success.

When something fails, save the specific lesson and update the relevant instructions. If the same lesson appears again, strengthen the existing note instead of adding another copy. That is how your working process improves: you carry the correction into the next attempt.

Pause when building stops helping.

I have needed to step away to see what I actually wanted. Keep asking whether this saves effort, improves a decision, or makes something useful happen. You can simplify, change direction, or stop. The goal is not to become the caretaker of something you did not need.

TRY THIS

After a few uses, compare the result with your original goal. Keep one thing that helped, change one thing that added friction, and save one lesson you want to remember.

A starting prompt

Help me review this experiment. I wanted [goal]. What actually happened was [result]. What should I keep, simplify, or stop? Separate what we know from what needs another test. End with one small next action.

↳ *Then return to the next real problem. You now have a better starting point.*

PUT IT TO WORK

One task.
One useful change.

You do not need my setup. You need a starting point that matters to you.

A small experiment

1. Choose one task and say what you want to improve.
2. Give AI the context and work through one example.
3. Use the result. Notice what helped and what needed fixing.
4. Save the useful part, then try it again.

Tell me what happened.

This is a first edition. I want to know which lessons hold up when somebody else tries them.

Which lesson helped? Where did you get stuck? What did you still need explained? Did this save effort or just add another step?

Send me your experience, including the part that did not work. That is how this guide gets better.

You bring the experience.
Let AI help you put it to work.

Adapted from my working notes and lessons learned building with AI. The exercises are starting points, not promises. Keep what proves useful in your own work.

TIM DILLARD / START HERE

First edition · September 2026