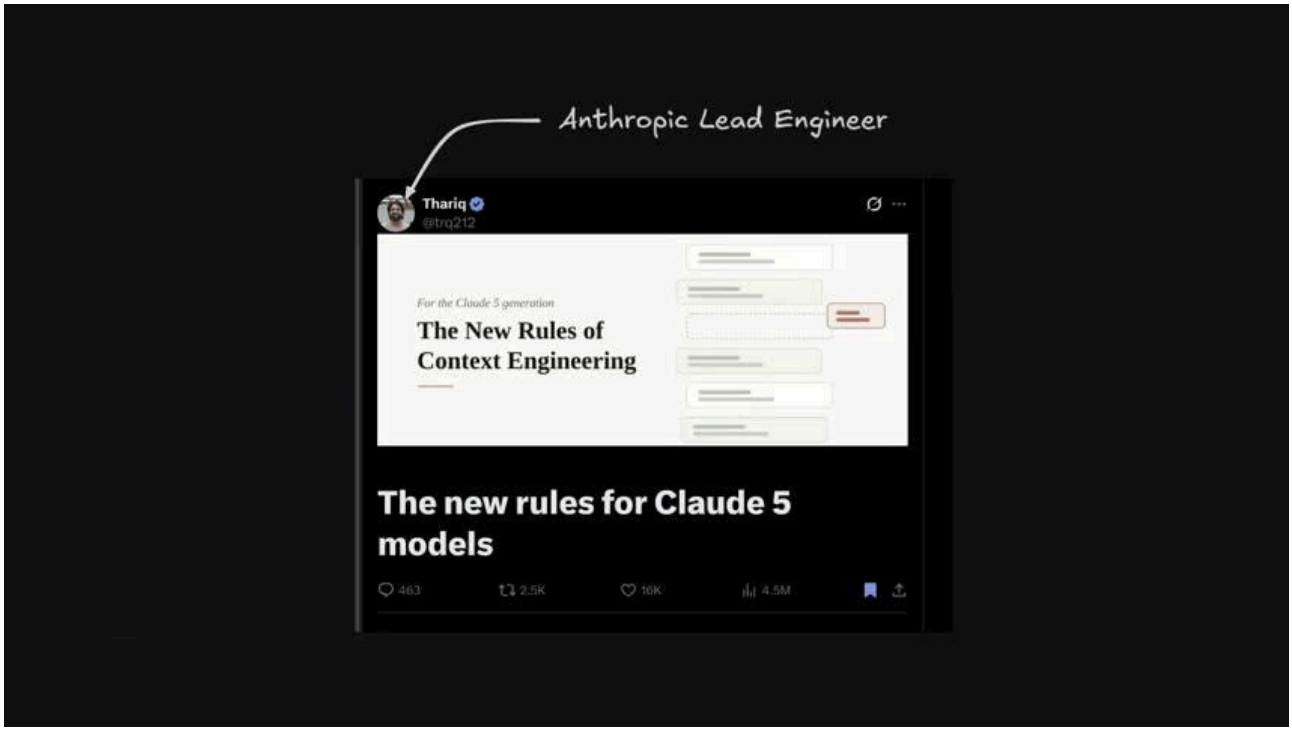
The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026

https://www.youtube.com/watch?v=8NSyI-npJCU

Introduction



at 0:02

Claude has evolved, with today's generation of Claude 5 models being a lot more powerful than everything that came before. But the way most people set up their agents and their operating systems hasn't caught up. So today I'll teach you this different framework for setting up your Agentic OS so you can get the full power from these new models — to make your systems faster, have your setup cost less, and ultimately be more productive than you ever thought possible.

I'll break this down into four simple parts so that by the end of it, you'll be using agents better than 99% of people. And if you're new, my name is Jay. I spent over a decade working with brands you may know, and I've been in AI since my master's in data science. Now I'm running an AI business and one of the largest AI communities globally. Let's dive into it.

The Agentic OS Dashboard

This is my Agentic OS, or more specifically, the virtual command center for my agentic operating system. From this one view, I can access summary information for the applications I use daily, like events in my calendar and time zones I care about, and a quick summary of my emails, including the messages Claude is flagging as needing my attention.

There are also quick links here to some of the micro-applications I created myself and use daily, which I'll

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

talk more about in a bit. I also have custom widgets, like this one for YouTube, because I do a lot of content. Here I also have a view of my routines and scheduled tasks, and which ones are going to fire at what time. And for specific skills where it makes sense to trigger them from this dashboard, I have this skills deck where I can adjust the effort level and the model to be used for a specific run, and run it straight from this dashboard.

Because these are widgets, and Claude Code is really good at creating these for me, you can freely adjust the size and placement of these widgets, and even create new ones depending on what you need. Lastly, because of the way I use Claude Code — where I consistently ask it to create artifacts for me — I had it make me this artifacts ring, where I can easily find the assets and artifacts it made for me in the past. So, for example, if I'm looking for artifacts for a client called THRO, I can search for that and open a specific HTML file it created for me back on the 5th of August.

at 2:04

Lastly, and very importantly, here in the center — if I click on that, it gives me access to my own second-brain system, which is crucial in the work I do because I need to visualize these systems so I can explain them better, and visually show the skills and other files in my workspace. I'll show more of that second brain in a bit, but this whole Agentic OS dashboard is great, especially if you're a visually motivated person like me. It also works well if you're serving clients or into AI consulting, because creating something like this for a client is a service you can package and sell.

To give a quick example, this one's a design mockup for a financial services firm here in Australia. Here's

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

another one for a company called Beto Green. The point is, if you have the foundations in place for your own personal Agentic OS, then with today's powerful AI models, it's quite easy to customize it for whichever client you're serving.

By the way, if you want to learn how to build and sell AI systems that businesses actually pay for, that's pretty much all we do over at the Rob & Nuggets community, where you get access to the Claude Living Master Class, which we update every week and takes you from zero to mastery with the latest in AI. You also get access to our Agents as a Service course, which walks you through how to actually get paid for the AI skills you're learning, and you get to be part of a genuinely great community of AI builders. You can see just some of the recent wins our members are getting from the program right here. So if you want to start earning from AI, check the pinned comment below. Now, back to the video.

Having a dashboard like this is great — it's visually appealing and lets you see all aspects of your work and business, which is why I use it as my homepage daily. But I'd say this visual interface captures only around 20 to 30% of the value of this agentic operating system, because the remaining 70% really lies in what's underneath.

The ARMS Framework

A big part of this operating system is how you've organized your context and your workspace, so that your system and your AI agent — in this case, Claude Code — works for you and not against you. There are many ways to organize your own operating system, but at least how I think of it personally, and how I teach it in our community, is via what I call the ARMS framework. It's simple to remember because it's like giving Claude, your AI agent employee, its own arms, its own workspace.

The core idea is: if you figure out how to best set up these four aspects of your OS, you'll be way ahead of 99% of other agentic AI users. Those four elements of the ARMS framework are the applications you use, the routines or scheduled tasks you run, your memory system, and the skills you have your agent use.

In my view, the best way to learn this is bottom-up. So for most users of these agentic AI systems, first you learn about skills, then you set up your own memory system. Once you're confident there, you rise up and can schedule your own routines, or even create your own applications or connectors for the applications you're using.

For the rest of this lesson, I'll go through this bottom-up, and for each one I'll share three levels of how you can use them, so you can freely skip ahead past the ones you already know — but I'll give you the depth of what I plan to cover in this video, so you'll pick up a few nuggets along the way you probably haven't known yet. By the end of it, if you watch the whole thing, you'll be able to level up your own agentic skills to the point that you can build out an Agentic OS similar to what I have here, customized for your setup.

Also, to make it easy for you, I've published a nine-page PDF guide — if you read through it, or just send it to your Claude Code, your agent will be able to guide you on how to set up this operating system as

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

well. You can grab that in the description below.

Skills, Level 1: Built-in Skills

Now let's start with the first element, which is skills. You've most likely encountered skills before, because they're basically shortcuts to your SOPs — your standard operating procedures. The basic principle is: when you find yourself prompting Claude for the same task twice, it's probably good to turn it into a skill.

At the very first level, if you're just getting started, the first skills you access are most likely the ones pre-built by Anthropic. You'd access those through the Claude Desktop app, under Customize — there's a section for skills where you can browse the ones Anthropic gives you. From here, you can see that one of the more popular skills from Anthropic is the skill creator skill.

Skills, Level 2: Creating Your Own Skills

The reason I generally advise people to create their own skills as well is that each of our work is really custom to us. When you get into the habit of creating your own skills, the faster you can get more refined and better results from Claude. You can find inspiration for skills everywhere. For example, a few days ago I found a post on X that looked like a really good tip to make your computer run a bit faster. So what I did was paste that whole tweet and then invoke the skill creator skill. If you send that, it lets Claude Code create that skill for you so you can test it out — which, at least for my workspace, has been quite effective. So if you've been having trouble with Claude Code or even Codex clogging up your systems and making them slower, this might be something to try.

Once you've created or tested out a few skills yourself, you start to realize there's a second level to this, because — contrary to what some people might believe — a skill file is actually not just the markdown file.

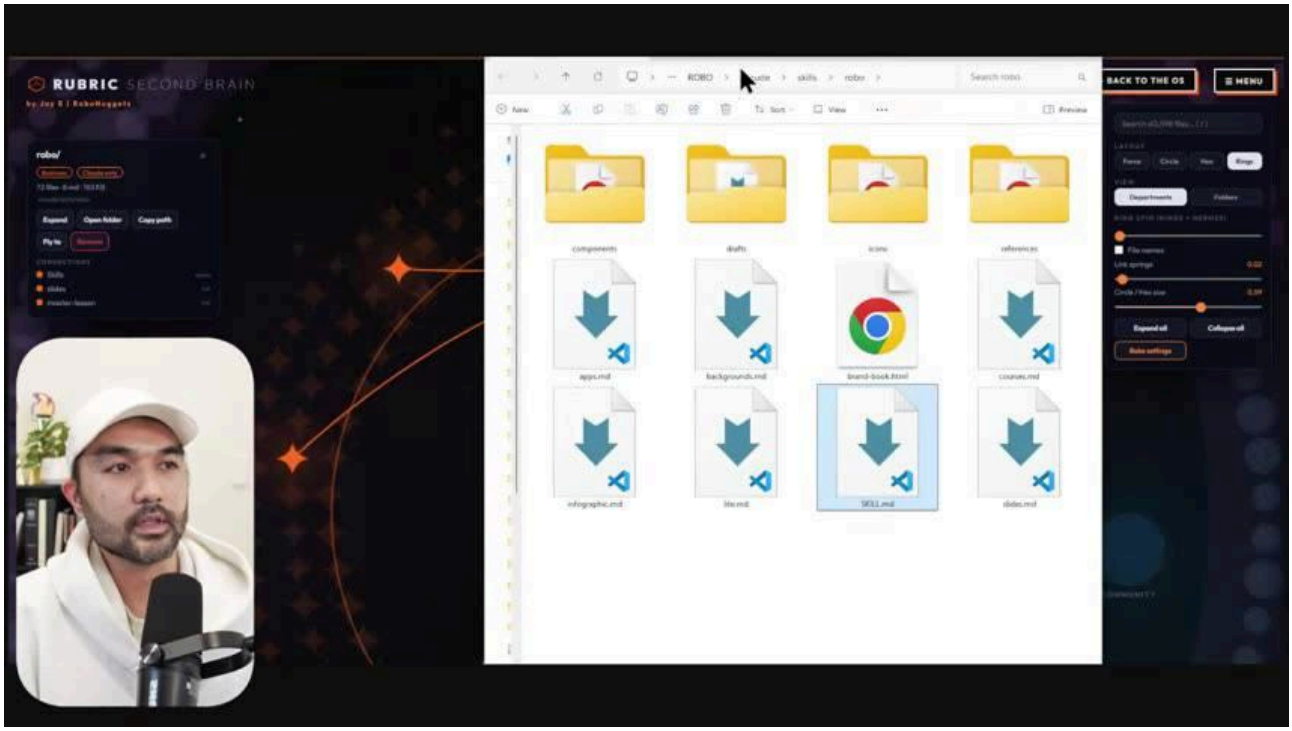
Rich References for Skills

Some of the most powerful skills you can add to your arsenal actually have rich references that they can pull from. To make that a bit clearer with an example, let's look at the cleanup skill I made earlier. This one's pretty thin and light because it only has this one SKILL.md, as you can see. If we open that, a SKILL.md is really just a markdown text file. It provides instructions to your Claude Code on how to execute this specific command.

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026

https://www.youtube.com/watch?v=8NSyI-npJCU



at 7:29

But let's say we look at a more complex skill, like this one for Robo. You can see that this skill, which I invoke with /robo, actually has multiple files connected to it. Let me open the folder so you can see better. In this folder there's the skill file, which is the markdown file that gives instructions to Claude Code on how to use this skill. If we go back here we can find that as well. And from this SKILL.md, you can see that it's essentially functioning as a router to these other reference files that also live inside that /robo skill folder.

The reason that's important for this skill specifically is that this is actually the design system I've been using for a lot of our videos and a lot of our company assets. If I open this brand HTML file, you can see it provides guidance on the Robo style — guidance on the fonts, guidance on the color palettes. Having visual references for your skills like this works really well, especially for skills that are meant for design.

For cases where you want the skill to do more complex tasks, you can enrich it and not limit yourself to just one SKILL.md to house everything as files. For example, this PDF guide I was talking about is well-designed and already knows our brand because I actually created it using that same well-built /robo skill. To make something like this, I just give a simple prompt: "make a PDF guide with /robo on how to set up an agentic OS." Then I answer a few questions Claude has so it's aligned with my intention. Because that /robo skill is already so well defined with a lot of visual artifacts and references, I get a well-designed PDF guide like this in just one or two prompts.

If you need a quick starter prompt to let Claude find your thick skills and turn them into ones with richer

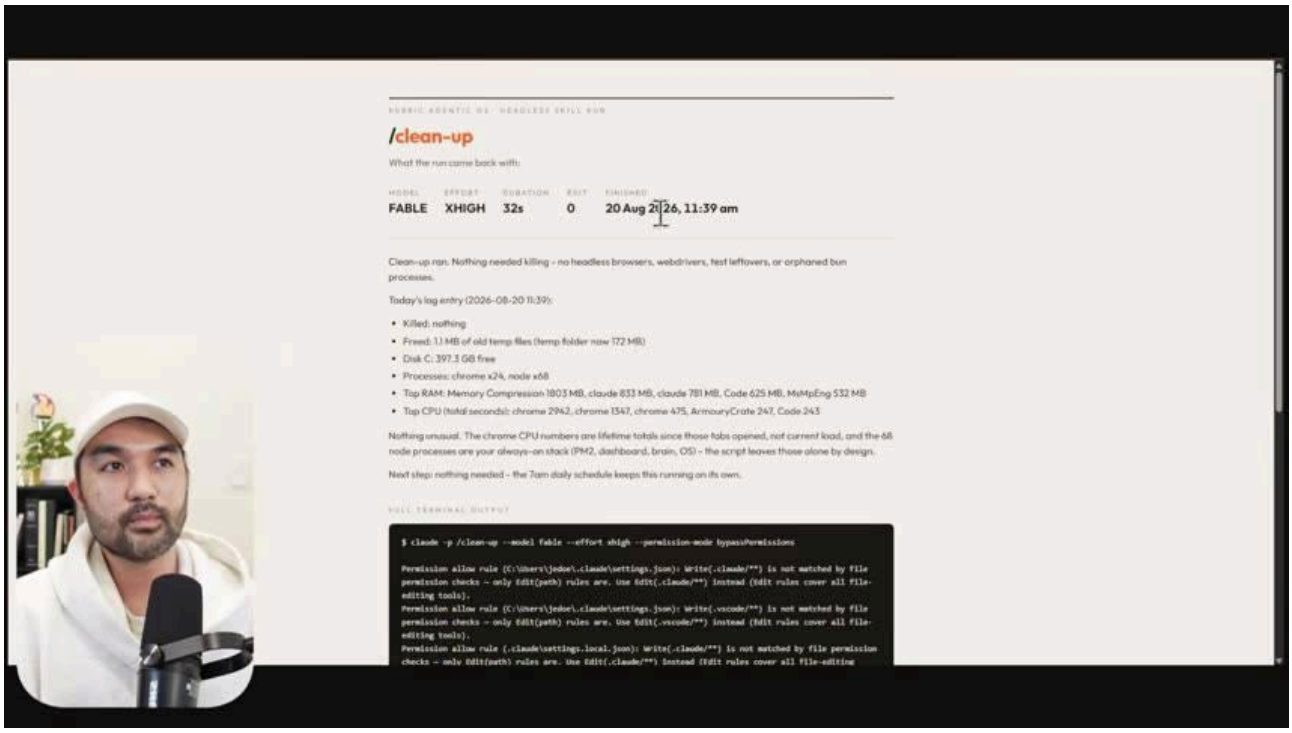
The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

references instead of stuffing everything into the SKILL.md, you can use this prompt — just screenshot it to kick-start that process.

Triggering Skills Outside of Chat Sessions

Once you have those, you can get to level three, which in some instances is actually useful: triggering skills even outside of your chat sessions. As one use case, I added that cleanup skill I created earlier to this skills deck so I can trigger it from this page instead of opening Claude Code in another terminal or chat session and typing /cleanup there — because most of the time I run this skill whenever I see my device slowing down.



at 9:37

When that's done, it also provides an output or report, similar to this one — if I open it, it just gives me a summary of the results of that skill run. The way you run any of your skills headlessly, meaning without opening up a chat session, is through a Claude feature called Claude -p. Without getting too technical, that essentially spins up a quick session that sends a one-shot prompt to Claude using the model and effort level you want. In this specific example, the only thing sent to Claude is the /cleanup command.

This becomes useful when you want to integrate your skills into dashboards like these, or even as part of internal applications you want to build for your team or company. And, like with most of the stuff I'll teach you today, the great thing about tools like Claude Code or Codex is that you don't need to learn any extra technical tooling to make this happen. As long as you're aware of this feature, Claude -p, you

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

can copy a prompt like this — just screenshot it and send it to your Claude Code — and that'll get you started with integrating any of your skills into your own operating system.

Memory

Now let's move to the next layer, which is memory. This matters because the more you use Claude Code, the more context and files you build up. At the very first level, what you really have is a workspace with a bunch of files. For my case, the folder on my computer that I set as my agentic operating system, or workspace, is called Robo. As you can see, there are already a lot of files and folders in here.

When you're just getting started and only have a few files, it'll mostly be fine, experience-wise. But the problem starts to arise when your context and files build up so much that it becomes harder for your agent to find things. For me personally, when I built out this second-brain system and pointed it at the Robo folder, that's when I found out I actually had something like 60,000 files already in there. You can imagine it's probably difficult for the agent to navigate through that, which has a direct impact on, one, how fast you can retrieve information from your workspace, and two, how quickly you burn through your plan's usage.

As soon as you start to see slowdowns in your systems retrieving memory, I advise people to move to level two: organizing your workspace in a way that's optimized for an agent. I say "optimized for an agent" because — if you're a millennial or Gen X — you probably remember when Windows or Mac operating systems first came out, and there was a period when you wanted to organize things neatly into folders, properly renaming files and folders so you, the person, could easily navigate the workspace and find what you needed.

In this new paradigm, where your agents are the ones operating on your files, you don't really need to pay as much attention to naming and navigability from within a traditional file-explorer view. With these agentic operating systems, at minimum what you should have in your workspace are what I like to call router files.

That's best illustrated in this second-brain system, where the center of it is your CLAUDE.md. That by itself is a router. My CLAUDE.md provides Claude context on the different departments I work with, so that when I work on content, it knows to operate within that set of files; when I work on my community, it knows those files are the relevant ones; and so on.

For each of these departments, I also have a dedicated router file. For example, if I search for the content.md file, you'll see it's just a list of skills and reference files, so that when I'm looking for something content-related, Claude can look at this list and immediately navigate to the files I want.

So instead of focusing on organizing your workspace in terms of folders and files — which is probably second nature for a lot of us who grew up with older, more traditional operating systems like Windows or Mac — remember that agents like Claude Code can parse through files at lightning speed. It's always much better to give them these router files so your agent can find what you need in the fastest way possible, with the least number of steps.

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

Again, you can start this setup just by talking to your agent. Copy this prompt and give it to your Claude Code, and it'll get you started setting up those router files. Then, once you're ready to upgrade your memory system, then you can...

Building a Visual Second Brain

Let's move on to level three, which is having your visual second brain system — the one I was showing you earlier. This is important for a lot of people because it allows you to see how all of your files and folders connect, and it also allows you to search for things faster.

You've already seen me do this throughout this video when I show my skills, or how my CLAUDE.md connects to different departments, or how these different files connect. That's really valuable for people who are more visual and understand things better when you show them in this format, versus taking them through the traditional file explorer view, which isn't the most engaging format.

The other thing is, let's say you want to go search for the cleanup skill. Usually with file explorer, it takes much longer to find. But earlier, as I illustrated, if I look for that cleanup skill from my OS, my second brain system, I can immediately find it and immediately preview it.

Routines: Level One (Local)

Now let's talk about routines. To me, this is the third layer, because once you've mastered skills and memory, that's really when you get the confidence to let your agent do tasks even without you monitoring it. That's the essence of routines — they're essentially just scheduled tasks.

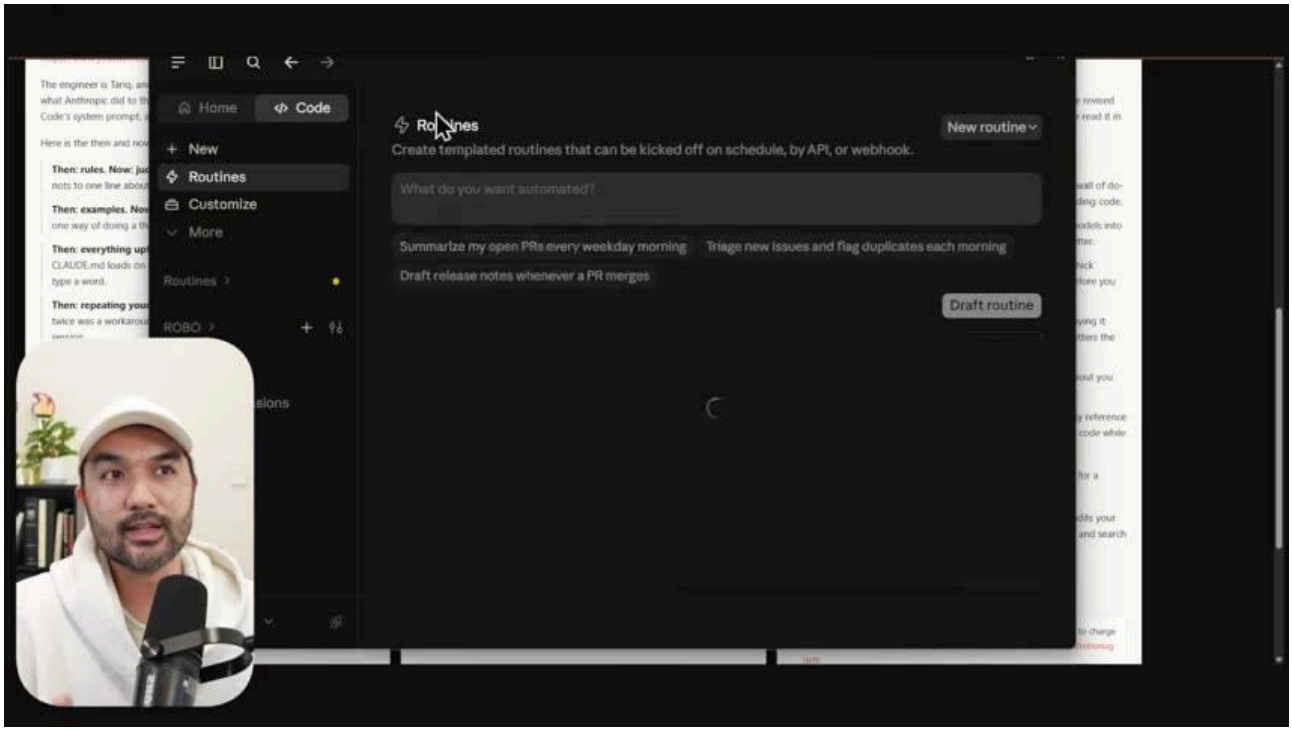
At the first level, this comes out of the box with Claude Code. In the Claude desktop app, if you head to Routines on the left sidebar, you can draft routines just by talking to Claude in natural language. I have a couple here — the one I use a lot is "YouTube to Substack daily." If you open that, you can see exactly when it repeats, which for this one is every day at 8:00 a.m.

In a nutshell, a routine is just a prompt that Claude sends to itself at the time you set. For this specific routine, it takes any new video on the channel and drafts it into a newsletter post in my tone of voice. When that runs in the background, it also puts the artifact in my Agentic OS. So let's say I had a previous video called "Six New Rules of Claude Code" — when it's time to review that, I just open it and it provides me with several drafts I can iterate on and review. Because this routine also uses a custom skill, I already have something like 70–80% confidence that it's within my tone of voice, and I only need to apply minor edits before it's ready for production.

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026

https://www.youtube.com/watch?v=8NSyI-npJCU



at 15:56

But if you notice, I only have a few routines set up in my Claude Code desktop app. That's because at level one, even though they're simple to set up, the key limitation with these local routines is that they only run while your computer is on.

Routines: Level Two (Cloud)

A lot of my routines are actually at level two, where I have scheduled tasks running in the cloud, so that even if my computer is off, I have the assurance that those tasks will still run without me. There are many solutions for this now. OpenClaw probably popularized it first. Grockbot is also pretty new, although it's paywalled at a fairly high price point at the moment.

The one I personally use is Hermes — I have several other tutorials on Hermes on this channel, or if you're part of the community, you can go through the Hermes Agent Masterclass to use it in the best way possible. The reason Hermes is so powerful is because it's 24/7, always on. It's always on because, for most people using Hermes, they give it its own computer — for myself, I give it a computer in the cloud. That's why, if you look at my routines board, most of the scheduled tasks I have are already loaded into my Hermes agent.

A key aspect most people miss: if your Hermes agent has its own computer, how can it access all the skills and context you've built up with Claude Code? There are many ways to do this, but for me, and for most people getting started, you can use a tool called Syncthing. Syncthing is a really good, free, open-

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

source piece of software that literally syncs things between your computers. If you point it to the workspace your Claude Code works on, and also install it on the computer your Hermes agent uses, you can sync the files you want, including all the skills and memory files you want to share with Hermes. As usual, all you need to get started is one single prompt, which you can use if you want to try it out.

Routines: Level Three (Your Own Server)

The next stage of routines is something I think is coming soon by default for Claude. I know some users tinkering a lot with this setup, where they get a VPS — a virtual private server, essentially a computer in the cloud — and install Claude Code onto it. All the files and context their Claude Code builds live in that space. In that format, you get the best of both worlds: routines that don't die out and actually run 24/7, and you're operating through just one agentic platform, which is Claude Code in this case, though you could also use Codex. You don't have to use an external tool like Syncthing to sync files between two different setups.

It's highly possible that OpenAI and Anthropic will offer something like this in the future, but because of file storage and security concerns, that's probably something to expect sometime down the road rather than right now.

Applications: Level One (Built-in Connectors)

Now let's go to the final element, which is applications. If you're trying to do any real work with your agents, you need to be able to connect to your apps. At the very first level, you connect to applications through the Claude Code desktop app — from there, if you go to Customize, under the Connectors section, you can find and browse the different applications you can connect to.

Applications: Level Two (Search Connectors)

In my view, this isn't the most efficient way to connect to applications, because at the second level, you can just have Claude Code connect to these applications itself, or search for what connectors exist, instead of you doing it manually. Personally, what I use to find connectors is a skill I built called Search Connectors. It searches the web for official connectors if any exist, or if there are none, for community-made connectors in the form of CLIs (command line interfaces), APIs, or MCPs — the three usual formats people use to connect AI agents to applications.

To give an example, I used Search Connectors for Adobe Premiere to check if there's an official one from Adobe. It also looks for anything community-made, and by the end provides a recommendation — right now, that's an open-source repo available on GitHub. If I want to use this connector, I just continue the session and ask Claude Code to scan it for safety and set it up so we can start using it.

Applications: Level Three (Build Your Own)

The great thing about these AI agents is that, at the next level, you can build your own connectors and

The NEW Agentic OS standard for Claude 5 Models is here (Full Breakdown)

Jay E | RoboNuggets | 21:38 | transcript saved 23 Aug 2026
https://www.youtube.com/watch?v=8NSyI-npJCU

your own applications. There are many ways to build your own connectors, but the one I personally use is the CLI Printing Press from Matt VH, co-founder of Lyft — I made a separate video on this if you want to check it out. Essentially, you can use it to create connectors for applications that don't have them. In that video, I created one for MyFitnessPal as well as for Skool, because no agentic connector really exists for those platforms.

And of course, nothing stops you from creating your own applications too. This whole dashboard is one type of application, but I've also built micro apps I use almost daily. I have an app where all the image and video generations you see on this channel and in our business end up in a masonry grid, so they're much easier to preview. And of course, I have that second brain system, which I've already shown quite a lot.

Building Your Own Landing Page for Visuals

Those Excalidraw illustrations that you're seeing, I actually have Claude build out this landing page where I can just copy in those artifacts that it creates for me and use them depending on the visual I'm trying to communicate. Whenever it makes sense, I encourage you to try creating your own applications yourself, because then you'll realize just how powerful these agentic platforms can be.

Wrapping Up

That is the ARMS framework in full, and I hope that was useful for you to craft your own perfect agentic operating system. If you want the full guide along with all of the prompts I shared here, so that you can start creating something like this for yourself or for a client, feel free to grab this PDF and send it to your Claude Code, which you can find down in the description.

That's it for this one. Thanks for watching till the end, as usual, and I'll see you all next time. Thanks.