How to Build $10,000 Agentic Workflows (Claude Code Tutorial)

24 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=VFEPZE_WRFG
https://www.youtube.com/watch?v=vFepZE_wrfg

SUMMARY

Agentic workflows represent a significant evolution in the AI industry, projected to grow from $7 billion to $93 billion in the coming years. This shift is driven by companies seeking faster, more efficient automation solutions that can adapt to unexpected challenges, marking a departure from traditional automation methods. The video demonstrates how to build an agentic workflow using Claude Code, emphasizing the importance of understanding foundational automation concepts to effectively leverage these new tools.

- *The agentic AI market is expected to grow substantially, reaching $40-50 billion by 2030.*
- *Approximately 25% of enterprises are already piloting agentic workflows, with projections of 50% by 2027.*
- *Agentic workflows allow for self-healing capabilities during execution, adapting to issues in real-time.*
- *Traditional automation methods are limited and often require manual intervention when encountering unexpected scenarios.*
- *The WAT framework (Workflows, Agent, Tools) is essential for structuring agentic workflows effectively.*
- *Building agentic workflows requires clear communication and understanding of underlying automation principles.*
- *Businesses benefit more from identifying and solving operational constraints rather than simply implementing flashy AI solutions.*
- *Value-based pricing is recommended for service providers to reflect the true ROI of their automation solutions.*

Agentic workflows are not just a trend. They're the future of the AI industry. More and more companies are making the shift to agentic workflows, and this is just getting started because it's estimated that the AI agentic market is going from about $7 billion this year to around 93 billion in the next couple of years. So, I can tell you right now that knowing how to build agentic workflows is going to be one of the most valuable skills that you can have. So, in this video, I'm going to break down why you should be building agentic workflows, and then I'm going to actually build one live in front of you so you can see exactly how it works. And by the end, I'll show you how to actually sell these if you want to make some money with your skills. So, let's get into it. So, before we build anything, I want to show you why this all matters because it's not just hype. This is real money moving into real technology. Right now, the agentic AI market is sitting at around $8 billion. By 2030, that's expected to hit 40 to 50 billion. That's not just a small jump. That's an entire industry being built in front of our eyes. And it's not just projections. About 25% of enterprises are already deploying agentic pilots this year, and by 2027, that number will jump to 50%. So, half of major companies will be running some version of agentic workflows within the next 2 years. And with that comes massive budget allocations, new security requirements, and a ton of new opportunities for people who know how to build these systems. So, why is this happening now? What's driving the shift?

It comes down to pretty much one thing, which is companies are starting to hit that ceiling of what traditional automation can do, and they're starting to realize they could move a lot faster with more agentic workflows. If you've been building workflows in tools like end-to-end or Zapier, you know the drill. You map out every step, you connect the different nodes or blocks, you handle the edge cases yourself, and it works until it breaks because traditional workflows will break when they hit something unexpected. And when that happens, someone has to usually go in manually and fix that. And that's maintenance. That's time. That's ultimately money. Now, I do want to be real with you here because there's a lot of noise online about agentic workflows that makes it sound like they're just completely magic and they fix themselves forever, and that is partially true, but only in a specific context, at least right now. Cuz when you're actively working in a tool at Cloud Code and you trigger a workflow yourself and say, "Hey, go research these competitors and build me a report," the agent is sitting right there with you. So, if something breaks, the agent can catch it mid-run, it can adjust its approach, it can update its tools, and keep going. That self-healing piece is very, very real, and it's incredibly powerful while you're building and while you're iterating. But, once you deploy that workflow to run on its own, on a schedule or triggered by a webhook or something like that, that is when you're deploying the code, you're deploying the tools, not the actual agent itself. So, if you've seen my previous videos where we've used the WAT framework, we are basically deploying the W workflows and the T tools, but not the A agent. But, I'll cover this more in depth later during the live build if you're confused. What this means is that the self-healing ability ultimately goes away when the code is up in the cloud, you know, running automatically. And at that point it does behave more like a traditional automation, but that's really a good thing because automations are predictable, they're deterministic, and those types are the best ones. So, then where's the real advantage? Really, it sits in how you build. Traditional automation is like building a train track by hand. You're laying every rail, every switch, every connection all by yourself. Whereas, with Agentic workflows, it's like you're just telling a construction crew, "Hey, I need you to build a train track from here to there."

And then they build it for you. Meaning, if they hit a problem during construction, they would figure it out. So, you end up with a better train track, it's built faster with fewer mistakes because the agent handled the edge cases during the build process that you might have missed or not thought of. And then the idea is you battle test it before you ever actually deploy it. So, then you have a lot of confidence that it will always work. So, in our train analogy, before we deploy that train track, we would have like 10 different types of trains test drive on it. They would be different weights, different lengths, and maybe different wheels. And we want to make sure that our track can work for all different types of trains before we deploy it. And the reason this is actually possible now is because the technology is finally caught up. LLMs have gotten really reliable enough to use in production and we're not just playing around with chatbots anymore.

These models can reason, they can make decisions, and they can execute multi-step tasks with real consistency. On top of that, we've got things to use like skills or MCP or A2A. We've also got infrastructure like trigger.dev, modal, or Vercel that make deployment way simpler than it used to be. And most importantly, we've got tools like Cloud Code that make all of this accessible to non-developers. So, we can see that the market is absolutely shifting towards Agentic systems and the numbers back it up. But, here's the question that's probably on your mind. Does this mean everything that I've learned about N and N or traditional automation is useless?

Not even close. And let me explain why. The people who are going to struggle with agentic workflows are the

ones who completely skip those fundamentals and jump straight into Claude Code, tell to build something, and have no idea if what is being built is actually good. They probably don't know what a webhook is or how APIs work. They won't be able to spot when the agent made a bad decision because they've never built the thing manually themselves. Now, that's not to say that a beginner can't learn Claude Code. And because you actually understand how automations work under the hood, you can communicate precisely what you want much clearer. And you'll see once we hop into the live build just how important it is to actually be able to communicate really clearly. All right, so now you understand why agentic workflows are such a big deal. Now, I'm going to show you how simple it is to build an agentic workflow using Claude Code. And by the way, if you want to follow along with what I'm about to build, you can grab all the resources and files that you need completely for free in my free school community. The link for that is down in the description. All right, so we're now in Visual Studio Code, which is where we're going to actually use Claude Code.

Visual Studio Code is just an IDE or an integrated development environment, and that's where like I said we're going to be using Claude Code. So, if you don't have this up and running yet, all you have to do is go to a browser and type in Visual Studio Code and download the right one for your operating system. And then once you open that up, it should look like this. First thing you have to do is install the Claude Code extension.

So, you're going to go over here to the left-hand side. You're going to open up extensions, and then you will see right here Claude Code. Or if you don't, you will search Claude Code. And once you pull that open, it'll ask you to install it. And then once you do, it'll basically just prompt you to sign in with your Claude subscription. Now, you do have to be on a paid plan for Claude in order to access Claude Code. As you can see on the free version, you don't get Claude Code, but right here on the paid version you do. So, you can start off with the Pro plan at 17 bucks a month. And then if you need to, if you keep hitting limits, you can upgrade to Max, which honestly you probably will.

I'm on the Max plan, and it is an amazing return on my investment. So, I would just go with the Max. But anyways, you'll get authenticated, and then it will bring you back here. And now we can actually get going using Claude Code and building workflows. So, I'm going to close out of this screen, and what we're going to do now is we're going to open up a project. So, on this left-hand side, I'm going to go up here to explore, and this says you have not yet opened a folder, open a folder. So, essentially, when we're in Claude Code, we're going to be working within a certain folder, and that's kind of the way that I think of like, this is the project that we're working on. So, I'm going to click open folder, and I'm going to open up a blank project. So, you can see I'm in a folder called newsletters demo, and there's nothing in it. It's completely fresh. I'm going to click select folder, and now we can see we're in this project. I'm real quick just going to close out of this, and close out of this, just so we have a really clean interface to look at with not much going on, and I can explain what we're actually about to do. So, to make this as simple as possible, in Claude Code, we have an agent and we have files. That's it. The left hand side is where we see those files. We'll see different workflows, we'll see tools, we'll see all these other things.

And then on the right hand side, we'll have the Claude Code agent, and that's where we talk to it, we plan with it, it asks us questions, and it actually executes and writes the code or builds the workflow for us. So, if I switch

back into Visual Studio Code, and I double click right here, and then I open up this button that says Claude Code, this is where we actually open up the actual Claude Code agent right there.

So, I'll close out of this. You can see this is kind of what we're talking about now. Files on this side, there's nothing there yet, and the Claude Code agent right here. Now, what we have to do next is give Claude Code a claude.md file, which is basically just instructions for this specific project. And you can really just think of this as a system prompt. So, that way when we, the user, send a message to our Claude Code agent, it doesn't just process what we said and respond to us, but it also, every time, reads the claude.md file. So, this is where you're going to put important things like how the folders are laid out, you know, where to find your different files, what its end goal is, any frameworks you might be using. So, in this case, what we're going to be doing is we're going to be using a framework called WAT, which stands for workflows, agent, and tools. So, real quick, if you pop over to my free community, and you go to the classroom, and then you click on Claude Code, right here you'll see the WAT claude.md. And you can go ahead and download this file right here. Now, once you've downloaded that file, you can actually just drag it over here to the left hand side, and it should pop up as claude.md. And if you wanted to, you could read through this entire, basically, system prompt to see what I'm telling it about how to build workflows, how to build tools, how to keep learning, and how to, you know, set up its its folders and everything. But, I'm not going to read that all out right now. What I'm going to do is just basically tell Claude code to set up the project. Read the claw.md file and then set up the project and the structure and then we'll start building workflows together. So, I'm basically just going to shoot that off and it's going to go ahead and read that and get everything ready. So, we'll see soon on the left-hand side you've got all our different folders set up. But, while it's going through and doing that, let me just explain what these different things are. So, agent, that is the actual Claude code agent that we just talked to as you saw. And the agent utilizes workflows and tools to help us automate things. So, the first thing is workflows. These are markdown files, which you just saw similar to the claw.md and it looks like this. It's basically completely natural language.

You could read through every line and understand exactly what's going on. It just uses things like pound signs and, you know, dashes and asterisks to separate like what's a header and what's bold. It's to stress importance for the agent. Workflows are natural language processes instructions. So, right now let's just use an analogy of a recipe. The workflow is the recipe, so you'd have a workflow for how to bake a chocolate cake. And when you want to bake that chocolate cake, it's going to tell you what to do in certain order.

So, it's going to say preheat the oven to this, boil some water. I don't know why you'd boil water for a cake. Crack two eggs in a bowl, you know, measure out a cup of flour, whatever it is. And those are the tools. So, the tools are all of the ingredients, but without the structure of the workflow saying use tool one, then tool five, then tool seven, then tool 10, without the order and the structure, the tools are useless. So, basically the workflows tell the agent how to build the tools.

And what's really cool about both of these is as they're being built and as they're being used, the agent will improve them over time if it makes mistakes or if it learns things. So, that is why we use the WAT framework to build our workflows with Claude code. So, now that that's done, you can see that this is finished up and it's basically said, "Okay, the project is set up. Here's a summary of the structure. Here's what I understand.

We're going to be building workflows in this project likely around newsletter operations. WAT framework, I will act as the agent. I will read workflows. I will run tools. I will handle errors and improve the system. I've got Python ready to go and I'm going to store secrets in the .env." So, this is where we're going to put our API keys rather than putting them straight into Claude so that they could be exposed who knows where. Okay, so what I'm going to do is go ahead and do a {slash} clear just to get rid of this conversation and we can start fresh. And we're going to start to plan out this workflow that we want to build. Before we start planning, I'm going to switch this to plan mode. So you can see where I'm bypass permissions, you can go to ask before edits, you can go to edit automatically, but I want to go to plan mode. And it's really, really important, as we talked about earlier, to be able to communicate clearly what you want. And the cool thing about Claude Code is when we give it a plan, even if it's pretty ambiguous, it will come back and say, "Okay, in order for this to be good, I need to know X, Y, and Z." So I'm going to give it a fairly ambiguous prompt here and then you're going to see it ask us questions and plan out this workflow for us. "Hey Claude, I want to build a workflow which will basically be a newsletter automation. I want to be able to tell you that I need a newsletter about a certain topic and you will do research, you will structure it in HTML, you will make it look pretty, and you will also create a few infographics to go with it. So help me figure out what tech stack to use here and what else you might suggest that I haven't yet thought of." So I'm going to shoot that off.

Now, whether we're in plan mode or, you know, bypass permission, what happens is the agent starts thinking and it starts testing things out. So it's thinking, it's reading through files. It has this little thing that will say computing or deciphering or wobbling or whatever it is, just a bunch of little silly words. Um but that just basically shows you exactly what it's doing. All right, so we just hit the point where it's asking us some questions before it continues working on the plan. The first thing is, for research, do you want to add an external search API to pull in data? So what I'm going to do is say, "Yeah, sure, let's just go ahead and do Perplexity." For delivery, it asks if we want to use Beehiiv or if we just want to send the HTML file for now. I'm actually just going to go ahead and say, "Let's actually just send this over in Gmail." And now it asks us about brand assets, which is really cool. So if we want to, we can send over some brand guidelines or logos and stuff like that to make sure that the newsletters are always formatted and they feel on brand.

So I'm going to go ahead and say, "Yes, I will provide some brand assets." So then what happens is it comes back with a final plan. Let me just zoom out a little bit so we can actually see that a little bit better. We'll go ahead and see what it came up with. So newsletter automation workflow, we want to conduct research, generate HTML with polished visual design, create infographics to accompany the content. We've got a research layer, we've got the content generation, we've got the infographics.

So, it says that it could use data stats infographics or it could use SVG. Why not image generation? It's too unpredictable, it can't embed. I'm actually going to go ahead and say that I want it to use nano banana. So, I'm going to go ahead and type in here. For the infographics, I want you to generate AI images using nano banana. You can use a platform called key.ai. So, this just shows the importance of plan mode and reading through the plan so that you can make sure before it actually starts building everything, you like what it's going to do. Okay, so now that new plan has been done, you can see the tech stack is going to be research with perplexity, the content will be written with Claude, the infographics will be created with nano banana, we will write the email in HTML and then send that via Gmail. And it even comes up with a section here about things the user likely

hasn't considered. So, things like human review, subject line, metadata, brand consistency, all this type of stuff. Now, the last thing that I actually forgot to give it was my brand assets. So, what I'm going to do real quick before we accept or, you know, keep working on the plan, I'm going to create a new folder over here.

I'm going to call this brand {underscore} assets. And you can see I dragged in two things. I dragged in AIS.png, which is our logo, and I dragged in our brand guidelines. So, I want the newsletter to be formatted in this way. So, I'm going to click on no, keep planning and I'm going to tell Claude that it needs to use those two assets. So, what's cool is that I can actually directly tag them. So, I'm saying make sure the whole newsletter is branded based on my logo and brand guidelines. So, for logo, I'm going to do {at} and I'm going to type in AIS.

And you can see that it's going to show AIS.png. And then here I can do {at} AIS and I'm going to click on brand guidelines. So, now it's going to look at those exact two things and it's going to be able to make sure that the newsletter is branded. Okay, so this time the plan looks good to go and I'm just going to go ahead and auto accept and I'm going to turn on bypass permissions. So, it's going to build everything out, it's going to put the different files that we need and then we should be able to basically just add our API keys and then test it. So, what it's doing now is it creates a to-do list.

So, this is all of the things that it has to do and as it actually completes them, it crosses them out. So, it's really cool because you can work on else on a different screen and just kind of check in on Cloud Code to see where it's at and if it needs any help. Now, you guys may be wondering about this bypass permissions mode. If you don't see this, you just have to go to your settings. In your settings, search for Cloud Code and then right here you'll be able to see allow dangerously skip permissions, which turns on allow bypass permissions mode. Okay, so that is finished up. It's telling us here's what it built. So, it created two config files, which you can see right here. It's got newsletter style, which basically just shows like the colors and the text and the background and it's got recipients, which is where we need to add who this is actually being sent to. So, this is where we would add a huge list of, you know, our email list, basically. Then it created one, two, three, four, five different tools. If I open up those right here. The tools are research, generate infographic, assemble HTML, send via Gmail, and archive to sheets.

And here is what all of those five tools do. And then of course, we have the actual workflow right here, which is our markdown file, which basically shows step-by-step how to actually build the newsletter and what tools to use. And this is the complete natural language just explaining the process. So, now that those have all been created, the last thing that we have to do before we actually test it out is we have to give it credentials. So, Anthropic, Perplexity, Key.ai, and then our Gmail.

So, what I do is I go to Perplexity, grab my API key, I would come into the .env, and then over here it's created these placeholders. So, all that I have to do is paste in my API key right there and then hit save to make sure that all of this saves. So, I'm going to go ahead and do this now for my other API keys. Okay, so we've done everything that it told us to do. We've set up all our credentials, at least I hope we have. If we run into any errors, Cloud Code should fix it or tell us what to do.

What I'm going to do now is just kick off a prompt. Write me a newsletter about Agentyc AI. So, I literally just said write me a newsletter about Agentyc AI and that's it. What it's doing now is it's looking through the relevant workflows and tools and it's going to figure out what to do. Here you can see it said I found the newsletter workflow starting with step one, I'm going to do some research. You can see after that it's going to plan and generate the infographics, it's going to write the newsletter content, and then we actually have a human review point. So, it's going to get subject line approval and then if it's approved, it'll go ahead and send the newsletter. So, now your job at this point is just to watch it and to make sure it's doing everything right and if it runs into issues it should fix itself but sometimes it may need you to help steer in the right direction. The first test run is the only one where it's really like this because you have to see how it works but then after that you should be able to just trust that it's going to run pretty much perfectly every time. Here you can see we've already run into our first issue. There was a Unicode encoding issue but it's just going to go ahead and fix it and that's great because I don't really know what this means at all so I'm glad that it understands what to do. Nice so you can see it planned out three infographics. It's got a market growth, it's got Gartner road map and it's got impact metrics. So here's a good example. It was trying to generate those infographics using key.ai and it was getting an error. So what it did is it looked into the problem. It said let me investigate to find the correct endpoint. It did some web searching, it looked through the docs, it did multiple searches as you can see and it figured out that the endpoints have changed and now it's able to switch the tool so that it works this time. There we go. So it said I found the fix, here's the right endpoint, let me update the tool so that this doesn't happen again and now it just went ahead and fixed the tool. All right, at this point it did a human review step and we could obviously say we don't want this if we don't want it but for now let's just go ahead and see what it wants. It wants us to approve a subject line. It asks us to choose which one. I'll go ahead and send five and then we will see the final output. Okay so a few things happened and I'm actually glad they did so I can show you how you need to troubleshoot this. The first thing is we got the email but the HTML is all messed up. It came through with a background color but then all of this just is horrible. So we're going to have it fix this. The second thing is I gave it the wrong Google Sheet ID to archive to sheets because there was some sort of access issue. So I'm going to go ahead and fix that sheet ID and I'm just going to use my natural language to tell it that this is horrible. I've updated the sheet ID however the actual email that I received is completely awful. I can't read any of it. It doesn't even make any sense. Take a look and figure out what happened and try to send me it again. So it's going to go ahead and diagnose what happened and then hopefully send us a better version. So once again it found the issue, it found out exactly how to fix it and now it's updating the workflow in the tool so it doesn't happen again. Now of course Claude code's not perfect. You guys can see that in this demo but think about if you were doing this in something else like end-to-end and something that's a bit more manual and you were running into these issues, and you'd have to go back and fix all of the logic yourself and try to debug all this. I've literally just been telling it to fix it and then like doing other work or going in the other room and waiting for it to figure it out on its own. Okay, now it fixed everything, and if I go over to my email, we see this newsletter, What happens when AI stops waiting for instructions? We can see that we've got our logo up top. We've got AIS Intelligence Brief. It does think that it's June 2026, which is wrong, but we can obviously fix that very easily. But now we move into the actual newsletter.

And keep in mind, this started with one prompt that said, "Write me a newsletter about Agentic AI." That was it. Also, throughout the newsletter, pay attention to the fact that it's using our fonts, it's using our brand guidelines, our colors, all of that in this newsletter. So the first section is about the market landscape, an explosion that

cannot be ignored. I'm not going to go ahead and read all of this text. It would just take too long. A nano banana AI-generated image with text, with graphics, and this infographic is also adhering to our brand guidelines. In section two, we have architecture. We've got a little bit of a quote here. And if we keep scrolling down, we've got some more statistics. We've got section three. We've got another quote, and we have another infographic, once again, adhering to our brand guidelines and using a little logo up here as well. And that's pretty much how the rest of the newsletter goes. We've got section four, and we can see our third and final infographic that has a different version of the AIS logo as well as our brand guidelines. So this was literally iteration one. There's a lot of things that we can improve here, and all we would do is we'd open up Cloud Code, and we'd ask it to make it better using natural language. We can actually make sure that every infographic it creates uses our actual logo rather than prompting some sort of AIS logo in there. It ends with some key takeaways, and then we have It ends with some key takeaways. We have a call to action down here, and then all of the sources we could actually click on, and it would take us to that actual site where it pulled the data from. So that was version one of the newsletter, and I think that that's pretty solid. Now, the cool thing about these projects in Cloud Code is as you use them more, they get better and better. Because every time I run this workflow, it might find something else out, and it will update its Cloud MD, it'll update its workflows, it'll update its tools. As I give it more brand asset, as I give it more context and more knowledge, it just gets better and better. And then, once you really trust the actual workflows and tools, that's when you go ahead and you come back into this. And then, once you really trust the workflows and the tools you've created using Claude Code, you would basically take these two things and you would push those into like a GitHub repository and you'd sync those to something like trigger.dev or Modal in order to actually have them run every single Monday at 6:00 a.m. or daily, something like that. I'm not going to dive into that in this video, but if you want to see one where I did, then I'll tag one right up here. So, what you guys just saw here was me using hardly any prompting, just using my natural language, giving it a few logos and colors and then giving us a really, really good output for a newsletter.

Now, one thing that we didn't cover in this video, but we will be covering a lot more in the future, is how you could actually make your workflows even better and better. And that's the idea of using skills, whether it is a skill that you create yourself or whether it's a skill that someone else has already built. So, skills are basically just system prompts that you could load in when you need them. So, let's say you ask Claude for help. Hey, can you design me a website?

The agent will then check through all of the skills it has access to and it will see, based on all of these skills, does my current request require this? So, it's almost like the same way it decides if it should use a tool or not. So, for example, there is a front-end design skill that makes Claude Code so much better at designing websites. And so, if I'm ever building a project where I need it to be able to build websites, I would tell it to always invoke the front-end design skill. And the reason I'm bringing this up is because you can create your own. So, what I might do in this version is once I realize what I really like about how it creates newsletters, I will tell it to turn that into a skill. So, maybe it is the skill of making infographics look really, really polished with the AIS logo in the top left corner. And I could create that skill so that every time it needs to create a new infographic, it reads that first and then it makes its outputs a lot more consistent. So, I know that this seems a little bit intimidating at first, but hopefully you guys realize after watching this how easy it was for me to actually do this. Once again, with hardly any technical knowledge, we didn't set up any API calls, we didn't do anything like that. We just talked to it. But now, the question is, how do you actually turn a skill like this into income? So, this is something that I see all the

time. A business owner watches for YouTube videos or LinkedIn posts or whatever it is and sees flashy AI demo.

Maybe that's a voice agent or a really cool chatbot or a crazy-looking dashboard and they come to you or some sort of like you know AI agency and they say I want that. But when you actually sit down and you look at their business and their operations, that's not what they need at all. The real problem is that leads might be falling through the cracks or the onboarding is taking way too long or there's tons of manual data entry going on. Just think about it like plumbing. If you have a pipe that's clogged, it doesn't matter how much water you pour into it, it's not going to flow any faster if there's a clog.

Most businesses are out here trying to put as much water into the pipe as possible, hire more people, throwing AI at random problems. But what they actually need is someone who can come in, find the clog and then clear that and then start to add more water in. That's really the skill. And if you can cut through the noise and identify real constraints and unclog that pipe, that's worth way more than building some super flashy agent that looks cool but doesn't actually move the needle. The build itself is also not what businesses are paying for because building is getting easier and easier every day, which is good news but it also kind of brings about some panic because more people can spin up these automations much quicker and that's becoming a little bit more commoditized. So if you're trying to compete on I can build AI automations, you're going to be in a race to the bottom. What you need to do is act as the doctor, not the pharmacist. I've used this analogy a lot on my channel. A pharmacist just fills a prescription that someone else wrote but a doctor sits down with the patient, asks questions, runs diagnostics and figures out what's actually wrong before anything is then prescribed. That's the difference between someone who just builds workflows and someone that businesses will pay serious money to work with. So when you're talking to a business owner, you're not leading with I build agentic workflows in cloud code.

They don't care about that. You're leading with I can save you X amount of time per month. You're leading with I can save this process X percentage of errors. And that's exactly why you should not be pricing yourself hourly because if you can build something in 30 minutes that ends up saving the business, let's just say 20 hours a week, that's not a 30-minute job. That's tens of thousands of dollars in value over the course of a year. So if you price yourself at an hourly rate, you're putting a ceiling on your income and you're completely ignoring the value that you're actually delivering. Now, hourly can be fine early on when you're just getting started and you're building trust and you need to get your first few wins but once you can clearly show the ROI, the hours saved, the cost eliminated, the revenue generated, all that kind of stuff, then your pricing should really be reflecting that value, not your time. Trading time for money is not very scalable. So, here's a simple way to think about it. You sit down with a client and you figure out their process, and you calculate that this system is going to save them $10,000 a month. Now, let's say you charge $5,000 for that build. That should be a no-brainer for them. They make their money back in 2 weeks, and then everything else is just profit for the business. And it's also a great deal for you because that build might just have taken you a few days, maybe a few weeks.

That is basically value-based pricing. Everybody wins. Now, in terms of actually finding clients, I've done a full deep dive on that in another video, which I will go ahead and link right up here. But, at a high level, the approach is simple. You don't need a huge audience. You don't need to start a full-blown agency. You just need

to start conversations with the right people. You need to be transparent about what you're building and lead with how you can help them. Once you deliver the solution, you stick around because once that first system is running and they see the results, they're going to want more. They're going to want you to optimize the build, they're going to want you to expand on it, they're going to want you to help find new opportunities inside their business.

That's how a $3,000 build turns into a $50,000 a year relationship. But, the key there is that you have to be the one to track the metrics. You have to take ownership over that. You have to proactively show them the value that the system is actually adding. That's super, super important. And that's exactly the path: freelancer to consultant to trusted partner. You're not just building workflows, you're becoming the person businesses rely on to make their operations smarter. So, we just went from understanding what's happening in the agentic workflow market to actually building one live and seeing how to sell these systems for premium prices. Here's the thing though, this isn't just the future of automation, it's happening right now. Companies are already making the shift, and the demand for people who can build these systems is only going to grow. So, if you want to dive deeper into this kind of stuff, I've got a community with over a quarter million members where I share templates, resources, and all the files from videos just like this one. And if you're serious about making money with AI automation, if you want access to live Q&As with me, if you want direct support, and even job opportunities, then you can check out my paid community. Both of those links are in the description. But, that's going to do it for this one. If you guys enjoyed and learned something new, please give it a like. That really helps me out a ton.

And as always, I appreciate you guys making it to the end of the video. I'll see you on the next one. Thanks, everyone.