How I got 30k+ GitHub stars as a nontechnical builder: vibe coding techniques and principles

45 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=JVB2F_HYKXA
https://www.youtube.com/watch?v=jvB2F_hYkXA

SUMMARY

The presentation explores the concept of coding as a storytelling medium, emphasizing the speaker's unique perspective as someone from a humanities background. They discuss their journey into coding, the projects they've developed, and how AI tools have democratized coding for non-technical individuals, enabling them to create impactful products.

- *The speaker initially resisted learning to code, believing it was reserved for technical professionals, but has since embraced coding as a form of self-expression.*
- *They launched several projects, including "Front-End Slides," which allows users to create visually appealing presentations using HTML, gaining significant popularity on GitHub.*
- *The speaker highlights the importance of aesthetics in design, advocating for the use of templates and design guidelines to avoid "AI slop."*
- *They emphasize the value of building small, opinionated products that resonate with users rather than generic solutions.*
- *The speaker encourages iterative development based on user behavior and feedback, stressing the importance of understanding actual usage patterns.*
- *They propose that coding is evolving from a profession to a skill accessible to everyone, similar to writing, and can enhance various roles beyond traditional engineering.*
- *The presentation concludes with a call to action for individuals to express their stories through coding, leveraging AI tools to bring their ideas to life.*

The theme of my presentation is code as a medium for storytelling. I think because I don't come from a traditional background, I think of coding from a different lens from most professional engineers or programmers. And so today I'll talk about a few things. I'll go through uh my own background, how I started VIP coding and the projects I've built and how I built them and then why it matters. So first where I started. So I actually never thought of myself as a coder. I studied humanities in school. I never took CS.

Actually, in 2023, I wrote a blog post called why I'm not learning to code. That kind of went viral. So, at that point, I was like, I'm proudly a humanities person working in tech. And I think tech needs more humanities people with a different lens. I was challenging the traditional belief that only like technical people can work in tech. But a lot changed in the past year. So, I started vibe coding maybe a year or two ago. I first tried a lot of the web- based tools, but at that time the models weren't good enough. So I couldn't complete projects. It was mostly like demos or little tools for myself. I couldn't really build for other people.

But I think January this year was a turning point. So in January, I started using cloud code and other coding agents. And I was able to ship products end to end to users other than myself. I've somehow got accumulated like almost more than 30,000 stars on GitHub with more than five projects with over a thousand stars which like beyond my belief because you as you can see in this chart like before this January I've never even used GitHub. I wasn't even sure like how GitHub worked. But since January I think January was a watershed moment for AI coding. it really a lot of the nontechnical people could really get their hands on coding and start building uh real projects. I'll go through a few of the projects I build. The first one is front- end slides. So the deck you're seeing right now is built from this skill. So this is a skill that allows you to build presentations using HTML.

So leveraging the coding agents front end capabilities. So replacing the traditional PowerPoints. This was inspired by the fact that I think the coding agents are very very good at front-end design. So I I was thinking how can we apply it to an area that also has to do with design but it's not traditionally touched by code. So I realized that HTML is a great canvas for the model to express itself. What you're seeing right now is like just a bunch of the templates I've built for the HTML slides and you they they look really beautiful. I don't think they look like the traditional slides. I actually give them a lot of references from like artworks and very unique color combinations. So, I wanted to create more different looks from the traditional slides. This is my most popular skill so far. So, it has more than 21,000 stars on GitHub. I actually didn't expect it to take off, but I think slides is such a universal thing that a lot of people need. with this skill you can make them really really quickly without having to you know do any of the pixel pushing yourself. So this is the from the user experience how it works. So first the agent like cloud code will actually start with ask you a few questions like what is the purpose and use case for your deck roughly how long do you want it to be? Do you have the content ready or should I help you write them? And then it will ask you like is it speakerled or more for reading like so because it determines the density of the slides and then it will actually build you three different style previews for for you to choose from. So it will build three different designs for the covers. So this is to let the user have more say in how the design is made and then it will go and build the full slide. And then after that you have the option to edit it or you can publish it to a URL like the the deck I have right now is like deployed to Versell and then I had it point to my own domain. So it become becomes kind of a website that you can easily share with people. And what's nice is you can also track it because it's a website. So you can actually know how many people have opened it etc. So, and then I realized HTML is actually a really great medium for slides because first of all, AI models kind of speak HTML as a native language because it's all over their training data. Like a lot most of the models are trained on tons of web pages.

So, they are very very good at arranging text and images on screen. And then also websites and slides have a lot in common. They're basically just text and images arranged in like re rectangular boxes and sections. And also if you make it in HTML it becomes interactive and sharable. So for example for this slide I have there are a lot of cool interactions like as you can see my cursor is turned into a laser uh pointer and then on the bottom you have these like navigation dots that allow you to easily jump to different slides and also hover to see the title. And then you have interactions like uh you can these animation effects or you can click to enlarge any of the media. So these are all the cool things you could do on the web. I wanted to talk a little bit about aesthetics because I think that is the key to this skill. I spent a lot of time on figuring out how to make the slides look stunning or like at least not AI slop. And the solution I came up with was templates but not the traditional templates. So, if you

think about traditional PowerPoint templates, they're just PowerPoint files that you fill content in. At first, I tried making the templates like HTML files.

But then I realized the model would overfit the template. So, it would just take whatever arrangement or layout the template had and then try to squeeze the content in that layout which compromised the content quality. So, then I heard I read this awesome concept called design. MD. So there there's a great concept called design.mmd which is basically a markdown file that describes the design of a kind of a set of design tokens for a web page. So things like the color combination the the fonts the vibe and overall uh design guidelines etc. So it's just a markdown file and so I realized actually the model can do a better job with just uh this markdown file rather than the HTML file. So that's uh what I ended up using. So basically if you look at the the templates, every folder is just a markdown file. Um and then just from this markdown file, the model can do a pretty good job replicating the design system. How to prevent AI slop in design. So I think the first thing is to use cloud code because cloud is a lot better than codeex or other models when it comes to front-end design. So I in my day-to-day I use clot both cloud code and codeex a lot. But if it's anything that has to do with design, I always choose cloth because it's just a lot better and also give a lot of opinionated references. So these are some of the examples of the kind of images I would collect um on like Pinterest or other places. Um I would just save them in the folder and then let Claude reference them when making designs. Codify design guidelines as design like I just showed. Um and also I think we need to know what AI slop is before we can prevent it. So I think last year AI slop looked like this. So you have a lot of like purple gradients but this year AI slop is a lot looking like this. So it's like this look is very clawed. So it's just like instrumental serif like itallic. So I think the goalpost for as always constantly moving forward. uh because once everyone starts using the same tools like claw design like you just see a lot of designs that look very similar to each other and then people get tired of it. So um I think we just need to know what like what this commonly designed slob looks like and then just ban them. And I I like to make playgrounds. So what what I mean by playgrounds is basically an HTML file where you can see a lot of different options for for different things. So, for example, this is a playground for fonts for this deck. So, basically just have Claude make a bunch of different options for you to compare. Um, and then this is a playground for colors. So, you could just click through the different combinations and get a feel for which one you like. And this is a playground for animations. So you can view all the types of cool stuff it can do on the basically let it show all the tricks it can do. So then you can kind of pick and choose what you want to use kind of like a menu.

So I think if you're not sure what you want, uh just have claw mix make a playground with lots of different options and then choose from there. And also I realized SVG could be pretty useful. So, this this was a website I made. It's like a list of my favorite restaurants and shops in Tokyo cuz I visit a lot. The way I made it was actually uh there's a thing called Google Takeout where you can actually take out all of your personal data from Google products. So, you could actually export all your saved locations in Google Maps um as like a Excel spreadsheet. And I give that spreadsheet to Clock Hole to turn into a website.

Uh, but what I really love about it is these cool illustrations like this cat and like uh these uh like this little illustrations for the different types of restaurants and stuff. So these were generated as SVGs using a product called Quiver AI. Um, so Quiver specializes in like AI generated SVGs, but I really like how intricate it is and how it kind of seamlessly blends into the overall vibes of the design. So I think SVGs is a very underrated um

form factor that we can blend into you know HTML designs.

And then for this deck itself uh the way I made it was first I I just had a folder of the various images and videos cuz I I did have a lot of media I wanted to embed. So I just put them in a folder and then I give it like very specific instruction on what I want to include in each slide. So this was like the the first prompt I gave it. And then of course it took a lot of rounds of different iterations, but I did have a very specific idea of what I want um in each slide. And then I I tell Claude how to use the different media assets. Love the playground idea. Would love to learn more about that. Yeah. So for playground is actually super simple. Literally I just tell Claude like I want to explore different fonts. Make me a playground.

that show me the different options. Um, and it will do that with like actual content that you're designing. So, the next category of software I made was super uh interesting. It's like modifying your own software interfaces to fit your own goals. So, what I mean by that is this is especially good for the web. So, you can make Chrome extensions for yourself that modify websites or apps that you frequently use to make them better for you. So, for example, I built a little Chrome extension called Tab Out. So, the problem I struggle with, as you can see now, I have a lot of tabs in my browser and I never close them. Like the only time I close my tabs is when my computer crashes and they are forced closed. I struggle with like finding the right tab when I need it and then it's just like too much to deal with. So, I want I just brainstorm with Claude like how what can we build to do this? And it it turns out that you can actually customize your new tab tab. So this is the new the new tab tab. So when you open a new tab, this landing page, you can actually v code this landing page. Um so I turn this landing page into kind of a mission control for all your open tabs. So you can see them at a glance. And then you can also um close close any tabs at one click. So, for example, I can see what are duplicate tabs and then just like close them with one tab. And then there's also a very satisfying like swish and sound and confetti when I close them. And then if I can close a lot of different tabs at one go for for example, if I want to close all my Gmail, I can just click that and then close all my homepages.

So, it it becomes almost like a game. It's very addictive. Um, I wanted to make closing tabs addictive because I think the way most browsers are designed, they only encourage you to open more tabs. They never encourage you to close them. So, I was like, I want to modify the the interface. So, that encourages different behavior. It groups them by domains, which makes it easy to see what you have. Um, so this extension is open source. Yeah, you can change the sound. You can change whatever you like.

Like this is the cool thing about open source is a lot of people have actually modified it to fit their own habits. So for example, this user just uh sent me this video the other day like they actually change it to like a to-do management system. So you could have your tasks uh you could have like your your schedule like your to like reading list whatever you want on this page. So I think the new tab tab is a very underutilized real estate on our computers because we just open it so frequently throughout a day. But usually it's just like a Google search bar or some random stuff. So I encourage everyone to make the full use of the new tab page and just customize it however you like. Like some people change it to dark mode, some people add their frequently visited web pages. So because it's open source, anyone can just like do whatever they want with it. So for most of my open source project, I actually want people to not just use it, but like remix it. Um, and so yeah, this is the code on GitHub.

So you can just uh download it and or get cloud code to install it for you. Um, and the other interesting thing which I'm still building, it's not finished, is like a tool that helps me learn from long YouTube videos. So >> right. So they >> so for example I have this like super long podcast that I want to watch but I may not have t time to consume the full length or the the content might be too dense for me. It's like a lot of technical stuff. So I could open this side panel which first first it grabs the full transcript of the podcast so I can quickly read the text. is also fully >> work audio and I can click right or trust >> part to like jump to that um part of the video. Uh if there's anything I don't understand, I can just select it and then >> uh >> and then the AI will actually exp.

So this is very useful for like any sort of technical uh videos or podcast for education. Um, and then if you're not sure if you if this is worth your time, you uh there's an overview tab where you know you get um highlight of the highle overview of what the video is about and worth watching if you are you know one of these things and then um it also divides it into chapters and then you can click on any chapter >> data efficient >> right >> to that part of the video. Um, it will also pull the top quotes from the video um, using LM and then you can click any part to download or copy it with one click. I also like this remixing idea cuz sometimes I prefer to read it. Um, but I don't want to read a bland summary cuz the summaries are usually really bad. Like they don't capture the best parts of the video. They're too brief and too generic. I like a few um, form factors for the remix. The first one is a magazine article. So the prompt is basically like transform the transcript into like a New Yorker style magazine that is like a long substantial read but also a lot more uh easy to consume than like maybe a long video. So this is an example of like a magazine article remixed from this specific video. And this is very suitable for things like a long video podcast like Lex Freeman type of podcast. You can also do like a business biography if it's like a person being interviewed or you can do like a briefing. Whatever format you like.

Notes is also really interesting. So as I'm watching the video, you can see there is a note button here. So what I can do is even without pausing the video, I can click it to take note. So let's say I'm watching >> entropic >> and then he says something I want to >> take note >> sun micros system >> and it's just gonna pop up this little panel that just take what he just said into you know a note and then uh put it in my notes panel. So, it's going to rewind maybe like a few seconds to capture what I just heard and then put it uh turn the transcript into a little note which I can like copy the text or time stamp like maybe I can turn into a tweet or you know put into my notebook or whatever. So, this is kind of just like a personal learning tool to make the most of uh long YouTube videos. How much time does it take to make a new project? It really depends. I think the making part is actually not that hard or timeconuming. The time consuming part is actually using it and feeling it and iterating it based on how you're using it actually and your own behavior because for example for tabouts I first made a more sophisticated tool actually I I had an LLM categorize all my tabs into like tasks and themes and then I realized I just wasn't using that feature. I ended up just like want to being able to close them. So, I turned closing tabs into the main feature after just feeling it for a while. So, I think even if building it only took like a day or a few hours, it maybe took me a week to get a hang of how I'm actually interacting with it and how I can shape it more around my actual behavior. The other category I I I thought was really fun, which was I think you can create your own personal ad network to shape your own behavior. So what I mean by that is so advertisers are constantly trying to capture our attention on the different platforms. But what if we can kind of manipulate our own attention to do things we actually want to do? I have this habit of hoarding X bookmarks. So I save a lot of posts on X as bookmarks, but I I just rarely open them. Sometimes I just forget about it or I don't have time. So I just end up with like hundreds of bookmarks. So I was like, how can I get myself to actually read the

bookmarks? I built a little Chrome extension that modify the X homepage. So I think the cool thing with browser extensions is they can actually like modify the web interface of products you use. So here I inserted like a bookmark from my bookmarks into like the top of my X homepage. So it's almost like an ad. So every time I open the X homepage, the first post I see is like something from my bookmark. So, it's a great reminder to, you know, okay, go read your bookmarks first before scrolling the rest of X because the X homepage is something I'm already frequently opening. It's like a great entry point for me to read the bookmarks. This is what I mean by like a personal ad network. So usually if you use any social media like this spot will be occupied by a lot of ads but you can actually use a chrome extension to shape it to to you know get the uh the behavior you want in yourself. Um and the same thing with like YouTube watch later etc. I think another real estate uh very underutilized is the phone wallpaper because we just open our phone like hundreds of times per day. When I was learning Japanese, um I tried turning my wallpapers into shuffling Japanese vocabulary images. So you can build these like wallpapers using uh clock code or any agent. So the the the images are just HTML. So you can just get the agent to make a HTML in like the phone dimension and then export as images and then you can put them in a album on your phone. And on iPhone, you can actually set shop um your wallpaper to be like a shuffle for a specific album. So this is how you get this like on key effect on the wallpaper and then every time you tap it, it shuffles to a different wallpaper. So uh this is kind of like how you can this like almost like a personal ad system if you want to study something. Um and this is also a cool lesson which is most people think you uh you can only make images with image gen models but actually code is very good with designing images like this. So if especially if it has to do with arranging text on screen or text and images actually code and front end is very very good for that. So if you want to make posters or like these cards or you know decks like code is actually a lot better than image generation. And the the the next category is content remixing. Now that we have agents, we actually have the opportunity to remix any sort of content into whatever format you like because we all consume content in different ways. So for example, I tried turning the acquired podcast into a physical book. Um I I like the acquired podcast. Uh it was it's just super super long and I don't have time to like go through all of them. So I was like if if it's a physical book, I would love to read it. So, um I actually just downloaded their transcript from their website and then use Claude uh use a super long prompt to turn each uh episode into a chapter of a book. U so each chapter will be like 20 pages. It's still a very long read, but I I actually really like reading it because the content is so dense and sometimes when I'm listening to the podcast, I find myself drifting off or like it's just hard to focus and whereas when I'm reading it's a lot easier for me to focus. So I and I actually got it printed as a physical book and I designed the cover and everything. So this is what it looks like. Um and then each chapter is like an episode of the show. Um and uh this is the prompt I use. It's pretty long, but basically I get it to uh it it's it's not a generic summary cuz I don't think a summary works. It's also not very readable. So, I got Claw to write it as a great business biography, kind of like, you know, Shoe Dog, The Everything Store, that kind of uh vibes. And also, it needs to preserve the best quotes and an anecdotes from the original show as much as possible so that you still get the best parts. Um, but then it maps it in a story that's very readable. And then um also if there are a lot of technical or business concepts that may not be understandable easily, I I also got it to explain it in layman language. This kind of prompt I build it with Claude.

So I have an idea of the types of things I like and then I just uh bounce ideas um with Claude together. So do I spend a lot of time on prompt building? I think I just spend a lot of time chatting with Claude. I think the specific words don't matter that much. I think when the models first came out, you sort of had to say like the magic words to get them to do certain things, but now they're smart enough that um you can just brain dump

your thoughts. I think brain dumping is very important. I know a lot of people use the voice input to do that. I think that's very good. So basically just like ramble, just give it a lot of thought.

It doesn't even have to be super structured, but I think just giving it your raw unfiltered thoughts and a lot of the context uh would be good. And then clock can help you turn it into a very structured prompt. The other type of remixing is uh turn videos into newsletter. Just like how I showed you, I can remix a podcast into like a magazine article. You can also turn that magazine article into a eub kind of like a ebook. And then you can have it emailed to your inbox on a regular basis. So you could get a weekly newsletter of your favorite podcasts and then read them in the ebook app almost like a book. This is also a super interesting project because I don't really know how to code still but I kind of want to understand how the code works under the hood. I also had an opinion around how we should learn computer science uh in this age of AI. I think traditionally in school we learned bottoms up. So you had to go through a lot of the foundational courses first before moving up to the more interesting stuff. But I feel like now we should just like build first. We should just turn our ideas into reality with the coding agents and then go and ask hey how did you build it? How does it work under the hood? So I made this project called codebase to course because I think the best CS course you can take is your own vibecoded project. like that's the best learning material. So, uh this is a demo of that. So, for example, I built this tool called long cut. I just gave the codebase to cloud code and like it's like turn into a course. So, this the course is just like HTML file that's interactive. So, it first breaks down what it does the user journey and then for the if there are technical terms you can hover over it to see a layman explanation. Um, a also had this really useful side by side of like here's the code and here's the layman like English version translation of that. It will also give you quizzes to help you understand the key concepts. And then uh this part is really interesting where so first of all you can trace a user journey through like these cool animations. So you can see for example how data flows in the system and then you can also simulate how the components interact with each other as a group chat. So it's like the different APIs or you know technologies in the codebase they're talking to each other just like people talking to group chat. So I found kind of simulating them this way really helps with understanding. So just a lot of different tricks to help you break down technical concepts into easy to understand components.

All of this is packaged into a skill that you can use on GitHub. So basically point it to any GitHub repo or local codebase and it will turn into like a HTML based CS course and then also turn Excel spreadsheets into interactive dashboard. So I think HTML is very good at visualizing data. So for example, here is a excel spreadsheet of my x analytics and then you can just uh give it to cloud code or any agent it will turn into like interactive chart uh that you can play with. Which tool I use for which skills? I mainly just use cloud code and codeex. The use cases I pick for them is like if has anything to do with design or brainstorming products, I always use cloud code. Uh, if it's just executing or like everyday day-to-day tasks, I use Codeex. I think Codeex is great for when you already know what you want or you just need to like fix a bug.

It's great at execution. It's also a little bit cheaper. But if you don't know what you want, you just want to like brainstorm and chat or you want product ideas, you want to be inspired, you want great design, then claw code is much better. The last part was about the projects I built. The next section will be about how I built them. So first of all how to get product ideas. Um the way I think about product ideas is I think is the intersection of users and technology. So a product brings user and technology together together. So uh when you think about product ideas there are two directions you can take it. One is to start with the user and the other is to start with

the technology. I think both can work really well. If you start with the user, I think I got a lot of ideas uh not just from my own pain points but also from talking to people who are very different from you because um I think products are all about meeting a lot of people's pain points and you need the pain points that's actually like generalizable and has a sizable audience which is why I think uh creating content has helped me a lot because when I put content out there I get a lot of feedback from people who are very different from me. So uh I get to know how people think and then I I also meet a lot of people for coffee and and stuff. So just hearing the pain points from a lot of different people really help uh inspire my ideas. Um, the other is uh feel your own point, brainstorm with clock code. And uh this is important like use it for a week and see if you'll still use it because I actually killed a lot of product ideas because I thought theoretically I would like use this product but then my own behavior just showed that I wasn't using it as much. So then I I just felt like it didn't work. So I think a good idea has to be proven empirically that you or someone would actually stick with it. So here I wanted to give an example of like tab out. I actually went back to cloud code and was like like hey go through our chat history like and then build me a HTML that kind of simulate how this idea came about. So it didn't start out with like hey build me a Chrome extension that does blah blah. It actually started with a question. The question was like my Chrome browsing history is stored locally right? Can you access it? because I just saw someone on Twitter build something with their Chrome history and I thought, "Oh, wow.

The AI can access my browser history." Like, it must there must be some room there for building cool stuff. But I I didn't know exactly what I wanted to build yet. So, I literally just asked Claw like like, "You have it, right?" And then it was like, "Yes, I have it." I was like, "Let's do something with it." And then I just told him my problem. My problem is I hoard too many tabs. I never close them. some of them are tasks um that I haven't finished like what can you do to help? I didn't really have a specific idea and [snorts] then Claw gave me a few ideas but then I said like how do you make sure I'll actually use it because a lot of ideas sound simple in theory like there's so many to-do lists or like task management apps. So I just end up like still not doing my to-dos cuz all of them require you to be very disciplined to like click certain buttons remember to open certain things. Then Claude said like maybe have you considered like making it your new tab page which I thought was a brilliant idea because on my own I didn't know the new tab page could be customized and like Claude told me that so I was like okay do that and then that's how that became the entry point for the product because I at first it it was it wanted to build me an extension that's you know where you have to click a button to pull out your tabs and I just told it I would never click the button. like I will never remember to click a button. So, can you put it somewhere where I don't have to click anything like I'm already naturally like opening it frequently.

Um, and that that's how it came up with the new tab page. Um, and then uh I got it to build some mockups and then uh just gave it a lot of specific feedbacks. Most of it is just from my own behavior. Um, and for example, can can you add like satisfying animation and sound effect? I just wanted to make it a lot more fun and addictive. So this is uh starting with a user. The other way is to start with a technology. So I'm always on X. I think I've tuned my al X algorithm such that it's like a very high density um information hub for like new technology and AI stuff. So I get a lot of new models, new products, new APIs, new cool technologies like use cases, designs. So I bookmark them and actually block out time to actually try them hands-on. So I think it's very important to try technology hands-on. If you see a cool new product or API just like send it to claude and like be like hey how do I how can we build something with it? So a lot of times it's not like you have a you start with a painoint a lot of times you just start with a technology like a cool API for example and then you can brainstorm

with claw like what can we build with it? Um I also I actually want to advocate for building something small because I think most of the internet will teach you to build something big but I actually think in the age of AI there's a lot of value in building small. Uh so first of all every big thing starts as a small thing.

I think in order to capture people's imagination you need a very sharp value proposition and positioning to stand out. And right now I'm hearing so many products that oh like allin-one AI agent that helps you with everything in your work and life and integrates with everything. And I just feel like that's so generic. Like if that's your positioning, I may as well just use Claude, you know, cuz Claude is that. So like if you want me to use your product, you have to offer something differentiated and special that has a opinion and soul behind it. And I think you you need to start with targeting a very specific use case and a very specific target audience and the problem and opinion. Yes, doing everything means it's doing nothing and speaking to everybody means you're speaking to nobody. So that's why I think we should start with building something small. And I also think we should build for fun and build to learn. So someone asked how did you monetize these projects? So I didn't actually build them for monetization.

These were mostly just like side hustles and hobby projects. also a great way for me to learn because I think the best way to learn product management is just to build products from zero and this wasn't available as an option if you were trying to become a PM say like three years ago because back then you had to rely on engineers to build out your ideas but now you can literally go with from idea conception to building it to distributing it all on your own so I'm kind of my own product manager my own designer my own engineer my own marketer all in one. So I think that's just a great learning experience. It's also really fun like almost like you can build a community of people who who are using your products and think alike. And I also think opinionated products survive. So like I said if you're just building a generic unopinionated AI tool like people may as well just use like their general like codeex clock or whatever. So the I think the reason people will use your tool is because the opinion behind the product resonates with them.

So here is the opinion behind my products for front end slides is like I don't like PowerPoint use HTML to replace PowerPoint like HTML is the next medium for storytelling and then for codebase to course uh the the tool that turns a codebase into like HTML based course opinion behind it is like in the past in traditional education we learn to build whereas now I think it should be reverse which is build to learn like we should build first and learn later um and for tab out is like we should make closing tabs addictive and then kind of change the default behavior around browsers. And I think the reason a lot of the products resonated with people is because the ideas and opinions behind the products resonated with people. So sometimes I almost think of my GitHub as like a substack. So because uh in the past when you had an opinion you wrote an article or you make a video or something. Now when you have an opinion you can put it into into a skill or a product because I think every product actually is a expression of how you think and the your unique way of perceiving the world. If you want to uh get started with building similar projects here are some practical tips to get started. But first of all is I think uh especially for beginners like using the best model can really save you a lot of time and even even save you money because I think Boris who's the founder of clock code once said uh if you use the best and most expensive model it can sometimes turn out to be cheaper because if you use a cheaper model sometimes it will just get stuck at solving certain problems or it just it just doesn't work and then you end up spending more time in token to get it to work. So I usually just use the best model by default. Uh but unless you have like wrote repetitive

work in that case you can use a cheaper one. And the way I talk talk to the model is I I don't treat it as my employee or assistant. I actually treat the model uh as my co-founder. So I do a lot of brainstorming with it. I describe problems not solutions. So for example for tab out I just said like all my browser history is here. What can we do with it? um without a specific idea in mind and Claw was the one who came up with the specific idea. So I think uh we don't have to have a very uh complete spec when we talk to Claw. We can just describe the mess we're in and see what we can do with it. Um and then the third one is build around your actual behavior and not what you think you do because human behavior is very messy. Humans are lazy. We're forgetful. We're like uh unpredictable. So uh if sometime for a lot of the products I think there's a lot of wishful thinking it's like they expect people to like remember to open a website and click on a button and do one two three like most people would just never do that or like they'll just forget. So I like to build products around that which is why I think after I build something it takes me like a week to feel it to see okay am I still using this or did I forget or you know did it did it actually is it actually shaped around my actual behavior and then tweak it from there. Um and then the next one is cut more than you add. So I think the models are very good at adding stuff and then what because adding things is so easy and the cost is so low these days we tend to become very greedy like more and more and more uh whereas I think now the human's role is actually subtract sub subtraction more than addition so we actually need to be very disciplined and always get the model to cut stuff uh so for example for tab out I first uh built in a um AI organization feature that categorize the tabs. Um, but then I realized just I just didn't end up finding it that useful. It also consume a lot of tokens. So, in the end, it was like just let as let's just get rid of that. Like I'll just categorize them by domain. So, this tool ended up having like no AI, but it also ended up being better cuz it was shaped around what I actually needed. So, I think before I ship anything, I always take the time to cut a lot of things. Um, and then, uh, I think putting the wraps is really important. So like anything in life, the more you do it, the better you get. So um I think I got a feel for it maybe after like my eighth project or so. So I just built a lot and then some of them fail, some of them more successful, but I think just building a lot um is uh putting the reps actually really helps.

And then building a public is really useful. So uh when I posted tab out on Twitter, it got almost half a million views. And then this um uh visitor like why does it need to run a noode.js server. So at that point it had a server and I actually didn't know why I needed a server. So I just asked cloud code like why do we need this? And then cloud was like oh we we don't actually need it. So I just deleted it in the end.

Well, I I wouldn't have known if like uh he had not commented on my post to tell me that cuz so I think uh bu showing your product in public uh just and then having people giving you feedback and comment especially from like engineers is really helpful because um then you can actually learn from the internet and then uh get better at what you build. Um and then the last one is just make it fun. So, I think it's important to be not so utilitarian. Um, like you don't have to like make a ton of money from it or, you know, get like 100 million users. Uh, even if it's just for you and your friends, uh, it can be really useful and um, educational. Uh, so I really like this quote by Peter Steinberger, who is the creator of Open Claw. He said, "It's hard to compete with somebody who's just there to have fun." And I think Open Claw was born out of, you know, like a fun side hustle. it wasn't he didn't set out to build like a billion dollar business. So I think a lot of good ideas can come out of these like non-utilitarian pursuits and then if we just try to have fun with the models try to push uh them to their limits and see what they can do we end up with a lot of good product ideas.

Um and then uh because a lot of people are trying to make skills I wanted to talk a little bit about skills specifically. So for example for front end slice the way I make it is uh first you just have an idea and then just try to get claw to do it. So for example like I want to make a slide HTML like just give it the content and do it without a lot of specific instructions and then it will inevitably make slop.

So the first iteration is probably really really bad. So you just give it a lot of feedback um like oh don't use purple gradients, don't do this, don't do that. And then this iteration will probably take like 10 or 20 or dozens of turns. Um, but this is very important like you actually need to spend a lot of time on this at this stage. But then the next part is really important. It's like tell the AI turn everything we just did and then package it into a skill. So kind of the workflows and iterations you went into that is the actual skill. So um the way you make the skill is not by starting with making a skill. is but by ending uh with a skill. So you do the thing first and then you say hey turn whatever we just did into a skill and because you give it so much specific feedback it's able to incorporate your specific taste and criteria into the skill. Um and then uh at the for the last step you can kind of package it into a more user friendly version and then just publish it on GitHub to share with others. Um so uh to close out I think uh I feel like coding is evolving from a profession into a skill. So what I mean by that is um it's kind of like writing.

So traditionally we think of writing more as a skill. So there are professional writers uh like journalists and uh book writers but mo we mostly think of writing as a skill. So no matter what you do you can benefit from being a good writer. I think coding is the same. So uh before AI coding, we mostly thought of coding as a profession. So only the engineers did it for a living. But now that um AI coding has made it accessible to everyone, I think it's evolving into a skill where anyone can just talk to the models and um you know turn the ideas into reality.

So and no matter what you do, if you're a business owner, you're a marketer, you're a product manager, designer, I think uh knowing how to work with the coding models will help you level up in whatever you do. Um and so I I think we need to differentiate engineering from coding. So engineering is a very serious like discipline. It's a career. It requires a lot of training. But coding is just a tool. It's just just like writing like a skill like if you have an idea you can uh you know turn it into reality. And um most people never need to build stuff in production like because you you you're not necessarily engineer working on like a production level product, but you can still build something that improves your own lives that you use with yourself, your family or your friends. So I kind of think of it as like we we should stop thinking about coding as something that's like super serious and like different distant from my life. Um so I think co vibe coding is a form of self-expression. So actually for most of my career I was a storyteller. So I started as a journalist. I worked as a marketer. So I I did a lot of writing.

So um but I think when I started v coding I realized um I didn't pivot from storyteller to coder. I just discovered a new medium for storytelling and that medium is code. So uh which goes back to my title which is code is a medium for storytelling. So I I kind of encourage everyone to tell your own story through code because previously you probably told it through you know uh writing or videos or content but now if you have an opinion idea or specific taste you can turn into a product a skill and then share on GitHub with others. Um so I that's all I wanted to talk about. So you can take this stack with you by going to this URL or scanning the code.

And all of the projects I mentioned are on my GitHub. You can see here. Yeah. Thank you.