GitHub’s Agent Era: 14x Commits, 200M Developers, Copilot’s Next Act — Kyle Daigle

84 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=LEWLSYR0CXA
https://www.youtube.com/watch?v=LEwlSyR0cXA

SUMMARY

The discussion features Cargo, the CEO of GitHub, who reflects on his journey from developer to executive and the transformative impact of AI on software development. He emphasizes the importance of leveraging AI tools to enhance productivity, streamline workflows, and improve team collaboration, while also addressing the challenges GitHub faces with rapid growth and system scaling.

- *Cargo highlights the power of AI in enabling developers to automate tasks and improve efficiency, allowing them to focus on creative problem-solving.*
- *He discusses his dual role as COO and CMO at GitHub and Microsoft, emphasizing the need for authentic communication with developers and enterprise clients.*
- *The conversation touches on GitHub's growth, with over 200 million developers and a significant increase in commits and pull requests, leading to scaling challenges.*
- *Cargo explains the importance of breaking down monolithic systems into more manageable components to improve performance and reliability.*
- *He advocates for a shift towards "ambient AI," where tools can proactively assist developers by understanding their context and needs without constant input.*
- *The discussion includes insights into GitHub Copilot's evolution, focusing on fine-tuning models and expanding its capabilities beyond code completion.*
- *Cargo encourages feedback from the developer community to enhance GitHub's offerings and improve user experience.*
- *He emphasizes the need for better integration of AI tools within existing workflows to facilitate seamless collaboration and productivity in software development.*

I just find that the folks that came from a different career, went to school for something else, went off and did this random thing and then became a software dev or were a dev, did a random thing, came back, learning that extra set of information, learning those extra skills and now having the power of an AI where I can crank up 15 agents on Saturday, you know, while my kids are doing lacrosse. Uh that's like really powerful and I think it gets me back to that feeling of like creation and it's very hard to like replicate that in most other senses.

>> Before we get into today's episode, I just have a small message for listeners. Thank you. We would not be able to bring you the AI engineering, science, and entertainment content that you so clearly want if you didn't choose to also click in and tune into our content. We've been approached by sponsors on an almost daily basis. But fortunately, enough of you actually subscribe to us to keep all this sustainable without ads and we want to keep it that way. But I just have one favor to ask all of you.

The single most powerful, completely free thing you can do is to click that subscribe button. It's the only thing I'll ever ask of you. And it means absolutely everything to me and my team that works so hard to bring the Inspace to you each and every week. If you do it, I promise you we'll never stop working to make the show even better. Now, let's get into it. >> Okay, we're here with Cargo, CEO of GitHub. Welcome.

>> Yeah, thanks for having me. >> You're not just CEO of GitHub. People know you as that. >> Yeah, >> you have a new role. >> Yeah, so I have an expanded role now. I mean we've been working uh I've been working at GitHub for 13 years and doing uh you know all things developer joined as a developer myself and now uh I'm also responsible as the CMO of developer for Microsoft and so all the kind of learnings and passion for developers and how we work with them and how we communicate and uh you know how we bring our products to market we're also bringing that expertise you know to the broader Microsoft ecosystem and uh and helping uh every developer that uses a Microsoft, you know, product or would like to uh to have a sort of similar experience that they've had with, you know, GitHub uh over the years. So, it's a big different uh role in some ways, but it's also just building on the experience that, you know, I've had at GitHub of just sort of tell the truth, be authentic, show people how to use it, and then let the you know, products speak for themselves. Not just doing that with all of Microsoft.

>> Yeah. And uh we're we'll be releasing this in conjunction with Build. you got lots of stuff planned and we can sort of touch on that whenever it's appropriate. >> Uh I think one of the interesting things is I rarely meet a COO who's also a CMO. >> I think you're a very >> outward facing and you're very confident publicly. >> Um that's rare. Like do you actually view yourself as COO? Like what's >> Yeah, I mean >> what is your thing?

>> I think for me like it's been funny. The titles have always been uh like always felt a little strange to me. I mean, I joined GitHub as a developer, you know. I mean, I wrote so much of the >> Let's bring that up. >> Yeah. >> Yeah. You wrote the back end. >> Yeah. I was going through like uh I was going through uh some old photos uh when, you know, folks were talking about, you know, how things were being built and how others would build GitHub.

I built uh uh web hooks and worked with teams building the API, built the platform layer, anything that integrated with GitHub. uh up until really 2018 I uh was built or ran the engineering teams uh and that's kind of where my like the beginning of my passion always was was helping people build things deliver them to like their customers. Uh and so being a developer building for developers was always super unique and I think as my role expanded it became you know my ability to talk to not just developers but also enterprise customers or you know business leaders and have this like translation layer. Uh and then through all those years, GitHub has always operated pretty uniquely. Like post- pandemic, working remotely was not as novel as it was when, you know, GitHub uh uh you know, started in 2008.

But all that expertise of running remote teams, doing it well, became this sort of bigger role ultimately turning into the COO role of how do we operate GitHub in the way that GitHub's always operated after the Microsoft acquisition. Yeah. uh and kind of so on from there. So I mean like for me I think the I've I still code. I love coding but the problem has always been like people. It's a much harder problem to both support our own

employees harder problem to communicate to developers and enterprise buyers what we're building why it matters because those are two very different messages. And so getting to work in the mix of COO, CMO, also just being a dev, uh I think is what's kept me at GitHub for for so long.

>> Yeah. Apparently you have your commits have gone up. >> Yeah. >> Uh what's this? What's going on? >> Yeah. I mean called me out pretty uh pretty aggressively. So I mean you know I think I mean as you can imagine, right? Like you can see my like normal era of being a dev in the you know 2013 2014 era and then moving into management and then ultimately the COO role. I think what you see there is me like really getting back to coding thanks to AI. You know, uh I similar to like attaching problems between, you know, how to market and how to operate a business and how to code. I find like building agents and workflows that are connecting very disperate problems to be what's driving this. So it's like some of it's writing software, a lot of it is like connecting a ton of different data sources to like help me out. Uh uh but that is completely me you know uh really really diving in uh on the AI side um in trying out our tools trying out everyone's tools like um but building for me building for the like non-technical leader though I'm technical you know and how we're uh able to use these tools more than just the simple like call and response that I think you know a lot of the like non-technical your employers like you have to get you have to use AI and so everyone uses like chatpt or copilot or claude or whatever to really get into like how is this going to help me out it I find that it's not the I need to write a blog post I need to you know those simple examples helping people find the workflows of like okay I need you to go through all the PRs today I need you to go through everything that we've posted online I need you to go through what we've did the last you know three months go through all of my Obsidian notes for any mentions of this then go through my transcripts at work. We uh use uh teams.

So like using work IQ, go call that MCP server, grab all the transcripts, go through all of Slack, and then build me out the plan of like what this week's messaging actually was. That's something that was like impossible because for me, I find AI in like what most of this like launch here is is actually like less building forward. It's actually like a recursive loop backwards. I'm always looking at what had happened first. Like go back through the week and tell me what we did, what worked, what didn't work, you know, >> and then tell me in the next, you know, three or four days, >> what would you tweak based on, you know, this sort of like looking backwards and then looking ahead a little bit. I find that to be so much more valuable especially for like non-technical because that retrospection is actually very LM are very good at that you know like finding all the patterns pulling them out and then applying that retrospection to just a couple of days or just like a short period of time uh is all a bunch of apps that I've built and launched like a bunch of uh internal tools I use the new uh GitHub copilot app the desktop app with workflows every time I crack open my laptop it's running workflows for me Uh it's just a ton of different stuff and of course it all ends up on it all ends up on GitHub.

>> Uh of course that's where that's where uh stuff is hosted. Uh man there's so much to ask you. Uh I was going to leave the the how do you run a company with AI thing at the end. I have to ask one one double click one thing. You said like you're looking back at the week you're you're understanding what uh what happens when you say we >> that's 3,000 people. >> Yeah. >> Yeah. how >> I mean I think you know uh when we started rolling out AI internally beyond engineering right um one of the things that I was really really uh passionate about is like we have to do this in a way where no one has to change how they work I don't want to have to teach you a tool I don't want to have to teach you something new and so for us we tried out a few tools most of

them don't work because I got to get you on board you know I got to teach you how to use it what we've actually ended up doing is we've built like a set of uh uh you know skills internally. We have like we each have our set of skills and we've just been distributing even to the nontechnical folks the CLI and then effectively we're just giving it access to like read about everything that we're writing. So that's for us that's usually GitHub um teams email and slack. Um so teams for video chat generally speaking >> teams and slack. Yeah, I mean so we use teams for video communication like but we don't use it for chat. We GitHub for a long long history right we always talk about chat ops and like everything is built into Slack like every command every flow >> even though you've been acquired for like I don't know eight years now. Uh >> yeah, I mean we still we >> still use that.

>> Yeah, it's I mean it's a purpose-built tool for us. And I think the reality is that moving off of it would be so bluntly expensive, you know, simply because all the tooling is uh baked in with that paradigm and they both have their pros and cons like but they don't work the same way like at all. >> Yeah. I mean we still use a bunch of different tools because it's you know the purpose-built tools that uh we need and >> but the same doesn't go for the rest of Microsoft presumably. I mean like the like you know various teams like operate in Yeah. various ways you know. Um I think it just matters what you're trying to like what you're trying to do.

>> Yeah. Yeah. >> Um but we do you know we do work across kind of every tool that we use and then uh by giving everyone access to all of that context in like um uh the new uh like work IQ MCP server which is quite cool if you do live in the M365 like world. uh I can ask it all these backwards facing questions and it's incredibly important for our teams that are working remotely. You know, there's a lot of stuff you miss when you're not in an office and we are spread out all over the world. So most of that is looking back and then we post uh we post either auto automatically into like uh GitHub issues or discussions um these sorts of like findings or like our industry reports like what's happening this morning today yesterday a little automation gets run we'll use the app we might use GitHub actions like with uh our agentic workflows just to go do that run and then we push it into GitHub and we keep having a conversation so usually for us it's about that sort like looking back, looking forward uh uh on the non-technical side. Um and then of course for a lot of those folks it's also you know building an app, pushing it to GitHub pages or pushing it somewhere to host it uh etc. Um but it's just like enabling everyone with that power of it's going to take me a week to figure this out. Instead we're going okay like I built a skill. Let's put it into a repo. We'll all share that skill together and then we'll use the CLI or now the app just to run it.

>> All right. I I think I think we're going straight into like the the team management and productivity thing. Uh I think a lot of people are getting various levels of LM psychosis. >> Um how do you manage the bloat of skills like everyone has their thing and they're like trying to promote it to the rest of their peers in your org, right? And obviously whoever >> becomes a skilled influencer internally becomes like an AI leader, right? Of sorts.

>> Uh I assume you have those. >> Yeah. I mean like I think we have >> and I assume it's a mess like >> Yeah. I mean there's like I like I think the reality is there's two pieces. Like first is I think that we're ending the era of these like massive beautiful perfect skills that are just like not any of those things you know uh cuz for a while right like every every tweet every day is like go download the skills the perfectly managed thing to do this entire workflow. And I think that like what we found in what I was just with my team um uh this week and we were

talking about the skill side and we're really talking about these like incredibly micro skills that are just doing one thing for us very very well versus a skill that's going to do like I said that full report that doesn't really exist on our side anymore. you know, it's usually like uh how do like a single skill that's going to identify the most important marketing information given any MCP server like this is the most important thing less about stitch a bunch of tools together and have it produce this mega output because then weeks go by, months go by, things change and you want to tweak >> your mega skill and you're screwed. You know, you can't do that. Uh, and so now we're really just talking about um like the Legos we're using and just letting the instruction book, you know, be something we're all putting together, whereas I think a lot of AI skills for a while have been that mega, you know, instruction book style.

>> Yeah. Uh, I've uh thought a lot about Postel's law. I don't know if that's a term that means things to folks. It's the idea that you should be liberal in what you accept and strict in what you output, right? And I think that's like a good framing principle for skills. Uh, this is my skills obviously. on. Uh I feel like everyone should have like you know how like some repos in GitHub are special repos. >> I feel like we should sort of raify the the SL skills and everyone like give it as some kind of special presentation.

>> Anyway, so um yeah, this is one of those like download uh download anything, transcribe anything and then you can string together the atomic skills that do one thing well >> into like some kind of orchestration skill that calls other skills. >> I assume uh does that match? >> Uh yeah, I I think so. I think that the the >> summarize anything >> totally like I think the for me summarizing something for uh like you know I do communications and PR and analyst relations and marketing and customer activities and so my summarize everything is very different for each one of those like contexts. You know what I Because if I'm summarizing something for an analyst, that's a very different thing than I'm probably how I'm going to summarize something for like a customer meeting or an engagement. So that's I think like the difference when we're talking about the um like the tools I might use on Saturday, you know, or the skills I might use on a Saturday when it's just for Kyle. Yeah, those are kind of like they have an atomic actual tool underneath or maybe skill and then Kyle cares about X. But I think when we're talking about work and enabling the uh you know the marketers communicators there it's the atomic this is what good summarization is and then this is what I care about as for marketing for communications for whatever and that I think is like the interesting matrix problem when we go from like a developer set of concerns to all kinds of different professions is that what that word means to me is different than it means to you is different than it means to the, you know, analyst or the saleserson.

>> And that's where I think the matrix mess is that we're starting to like still starting to find. It's not these mega skills, but they're all just slight permutations, but those permutations are really important. It's the difference between someone reading this and going, >> did AI make this? You know what I mean? Or like this makes total sense, and I I would expect this when I'm giving a briefing to Gartner or like whatever else. >> Yeah. I think a beauty of it maybe is that you don't have to be that careful about what goes in there. It doesn't have to exactly fit uh as long as it like roughly is contained in there. I used to complain about plug-in hell basically like when you have a framework and then you have 100 things that you need to integrate everyone do does like the GitHub used to be bloated full of these things and now we don't need them anymore.

>> Yeah. >> Now you just use skills. >> Yeah. Oh, and like I think the most magical thing is that just that like I can just also crack it open like yeah, you know, like yes, I could go like, you know, change the how the plugin is coded or like I could go uh, you know, do that now with AI, but I think there's just something more magical about getting a response back and be like, that's not right. And then you just crack the skill open. You just type English words, you know, and it's different. Uh, >> that that that building block is just uh I think very unique. um once uh once I get everyone to kind of understand how to best uh you know how to best make those changes uh uh you know to get the most power out of them. Is there a you know you you have a your peer group of people like you >> is there a common framing for the something I'm feeling is which is true is that uh is this a golden age for former developers who are now in leadership >> right because you can wield the tools you would know the right words you're maybe not too close to the details doesn't matter >> but like you're more effective than someone who doesn't come from that background >> I think that like the secret has always been your ability to identify patterns and solve problems. And I think that, you know, for folks that like myself that don't code day-to-day anymore, that has made me successful as a developer, made me successful as COO, now CMO. And so now that I have access to Git and write code, I'm now applying that sort of like pattern finding and problem solving and I know enough still, you know, about how to then go and say, "Oh, I want to make an app and I don't want to, you know, break into jail or create something that's not going to be able to work or to be deployed scale or whatever." Uh, that ability to apply all that additional business knowledge, you know, and still code, I think, is what makes that so interesting to me.

slightly different than I think some of the other like technical leaders that became business leaders and now are going back to their apps in updating them. Good for them, you know. But I think the the more much more interesting thing is well now I have this whole new set of expertise over 10 plus years. Why not take that and use that as a developer with these AI tools? So I definitely think that makes me more powerful, but I think that's true for like every dev as well. you know, most of the dev friends I still have also have some other underlying skill and passion. You know, there's really talented very, you know, kind of linear computer science software devs.

Absolutely. I just find that the folks that came from a different career, went to school for something else, went off and did this random thing and then became a software dev or were a dev, did a random thing, came back, learning that extra set of information, learning those extra skills and now having the power of an AI where I can crank up 15 agents on Saturday, you know, while my kids are doing lacrosse. Uh that's like really powerful and I think it gets me back to that feeling of like creation and it's very hard to uh uh like replicate that in most other senses. You know that first time you build an app and you click it and you show someone like that's magical. And so being able to do that not just in code but across all kinds of different assets like that's that's huge. Uh we were doing we're doing our like um every year we do our revenue planning you know we talk about you know okay what is it going to look like for next year and of course as you imagine there's uh slideshows everywhere you know talking about what are we going to talk about what's the narrative etc and so as you said you know I'm like okay well I could probably just like build something to build this and then that way I don't have to go build the whole spreadsheet where I have to pass it to my team. So we we went through this process and I got all the information and used the skills I mentioned. I built like a little app just to make it so I could look at some of the information in a SQLite database um more easily. And I ultimately built this entire presentation without touching any of it. And I was like, okay, I'm just going to present this to our CRO, the CFO, their teams without

mentioning I built it with AI. Uh I like built a skill to make it look very much not AI driven. Uh just not pretty, not pretty, but just like very clearly not AI like kind of like don't do anything interesting. Just go exactly >> did the whole thing through. It used my notes from Obsidian. It used all the context I mentioned before the plans >> and >> never came up once it was AI generated.

>> Yeah. >> Never once. Exactly. It didn't matter. And so now I can take that tool >> and go look I don't want you to go build slideshows. Yeah. They're just helping us share information with each other. If this thing can do it with a little bit of crafting from you and then we can look at it together, >> awesome. There's no value in all that extra work. I think that the ability to like make it look humanly bad and you know like and build a little app to like manipulate the data I think is part of like uh that upside for devs that are now in leadership roles because like the thing that I feel like uh like I said before this that's all a people that's all a people problem. I know if you've used a co-work or not to build a slide deck unless you spent a bunch of time to to not >> Okay. Well, so like I think there's a certain charm to just being blatantly AI. So I think that you're like, well, you're just honest about like there may be mistakes here that I cannot vouch for.

>> Yep. >> Um, so you know, how much value is is there? But anyway, like I think actually the real question I want to ask is like there's you were a chief of staff to Thomas >> and in in the preai world that that job would have been a chief of staff job of like can you prep me these slides and all that and now you do it yourself? >> Yeah. I mean like I I still I still have a chief of staff because like the difference is like >> it's sort of the the the discussion every time we you know have some sort of technology uh you know evolution is it's not that the >> the jobs like the roles don't all go away they just change you know and so yeah I don't have someone spending all their time building out slides for me in presentations because I don't need that anymore but now I need that person that is able to go and find all the different connections between humans in those discussions to help me find out, okay, I should be meeting with this group and this team and they have an opportunity and I'm going to be in San Francisco today. I'm going to be in Seattle tomorrow. Those sorts of like um human connection uh aspects is still incredibly valuable and has always been a big part of that like chief of staff role. Um >> but now just like uh you know chiefs of staff are not opening up like letters to process, they're doing email, you know what I mean? It's the same thing. and now they're they're not building out as many of these presentations because they have that, you know, the ability to have uh AI take it on for them. Uh and share that with me and great like let's keep moving because it's allowing us to go faster and make better decisions more more quickly.

>> Yeah. Awesome. Um well, so we can dive into more sort of uh productivity insights as you go. I did want to do a little bit of a brief history of KitHub. >> Yeah, sure. >> Uh because like we we started here >> and then you also involved the MPM acquisition. And I did want to touch up on that. >> Uh and then more recently I just want to bring up to present day where we're having uptime issues which transparently you've already already addressed publicly but we'll we'll discuss in the pod.

>> Sure. >> Uh did I miss anything like what any other major highlights? >> Obviously it's it's a lot of years to cover. >> Yeah. No, I mean like uh the I think one of one highlight was uh right before that acquisition closed in 2018, I got to launch the first version of actions uh GitHub Universe. Uh so it was >> they're that young. >> Yeah, it was October of 2018 I think. Yeah. Yeah. >> Jesus.

>> Yeah. Yeah. I got to I was an engineering leader on that project and got to launch that. Uh and then yeah, we did acquisitions of you know MPM like you said uh SEML dependabot you know Panda like a whole bunch of things. That was a big >> panda, >> right? Uh uh Obby Abby is doing well did well on DX. Uh >> you know I and like that was a that was the big shift after the acquisition. I to join the sort of business side.

>> So I need to hit you on on some of these things because you were there right and how how often do I get to talk to someone? Um but actions is that the number one source of security issues on GitHub. >> Oh sh I mean I think that the number one source of uh security issues is probably like the the literal code in everyone's like underlying repositories. I would say back further than that is uh if you remember like I had like in this graph this is I didn't say this before this is ultimately web hooks.

>> Yes. >> Like circa whatever it was I forget. Yeah hook shots in there. And so like back then it says GitHub services. Do you see uh it says Hookshot Hookshot FE for front end and then it says GitHub services. GitHub services back in the old days right you like we had a repository that was Ruby code and you could write any Ruby code in there and then we would execute that >> on your behalf as a service and then that way you know if uh if you were trying to integrate with something it didn't you know we would run it for you.

And of course no containers because >> no because it was 2014 you know like uh and so there was some isolation obviously but it was mostly the separations on the server level. Uh that's like an example as long as the very old version of pages which ran on its own containerization infrastructure not on actions >> which like all time great product pages powers the internet like uh you know at this point to some degree uh those were places where like clearly there were no like you know uh like uh issues like to my knowledge but it was those things where I'm looking at and going okay well like we can't be running arbitrary Ruby code you know like on everyone's behalf then containerizing all of that up into uh into actions now where like yeah like the containerization like is really good the like pinning like most folks aren't pinning it the uh like to a particular shot etc you know like their workflows and so that's a big p that's a big place uh uh of um of you know pain for folks if they're just doing >> similar to any you know dependency management just v1 or you know newest or latest I think um but uh uh that journey from that day to like okay we're just going to run all this arbitrary code and like it'll basically be okay to now. No, I mean we have like really good containerization. We have a new um uh underlying uh a agent uh containerization uh uh service. It's like through we're using it under the hood. It's through Azure. They recently announced it um uh the Azure like dev compute, but it's like very fast uh very fast compute to be able to like spin up uh you know your own cloud agents uh or whatnot. Uh we're using it under the hood uh for some parts of the new uh >> box.

>> No, no, no. Uh dev compute. Yeah, >> not finding it just yet. >> Oh, it's it's in there somewhere. >> All right. Well, we'll cut that out. >> Sorry. Uh but but with with like dev compute, you can like run uh really really fast u spin up really uh small VMs really really quickly. So you're doing a tool call just do it like containerized. Exact. Exactly. Uh so we're using that. So definitely moving that direction to protect us from every you know every uh every piece of code that we're ultimately running.

>> Yeah. I mean like look like that grows into the the full SDLC you know like um code hosting was just the

start and you know then it's grown grown beyond that. Um let's talk about npm maybe because I think that's also like a very major >> point in the industry like I I do think like uh it was looking for a home. was like kind of struggling as a business, right? I don't know. I don't know how you characterize that that whole acquisition and you know how it >> Yeah. I mean like you know when we were talking uh you know to the team I think the big thing for the both of us was to find a way to keep npm which was basically powering the internet then and way more so now to some degree you know uh uh running you know like keep it going continue it to scale was having um scaling problems if I recall back at that time they were doing some rewrites uh >> I mean that's cute compared to now >> yeah well that's the thing is like you know, when I'm talking to folks now, like there's, you know, there's so many more underlying uses of MPM than there were, you know, back when we uh uh uh we had them join uh join in with GitHub.

But that was ultimately the goal. It was really like, okay, we used to have pages, we have uh uh like the world's code. Let's make sure that we can keep MPM running well uh uh uh you know, for the world. and we put a bunch of time and investment into fixing some of the underlying backend um uh changes some of which we talked about like some of the manifest work etc. and then now like really trying to bring the um you know the security posture of MPM up to speed.

But like it is a unique challenge in that every move that we make to make it more secure will break a lot of people. Um, and security is paramount and also like we take it very seriously where like the, you know, anytime that we have a problem with GitHub where we make a change that makes us more secure but hurts there's like a snow day for developers or a really bad fire that they have to go put out. And so we like have changed the 2FA policies, you know, we've changed the way the tokens work.

When we find tokens that have been exposed or potentially exposed, we invalidate them. I love that feature of GitHub >> that creates issues. But like the but that's the thing is we're trying to push the community uh forward without necessarily uh uh doing something that is going to break the contract that's been for 15 years you know or close to or you know some amount of years you know uh on uh uh on npm. Yeah, I think the so now we're talking about uh open source and and publishing >> and I think there's something here with what people are calling slot forks which uh I think Malta from Versel is doing.

>> Yeah. Uh and uh part of me thinks like well the way to get past any like uh vulnerability is we just let's just get rid of the concept of npm >> and we only publish source code >> and anytime you want to import it you you have your coding agent look at it and then adapt whatever subset you're going to use into your like vendor it but like the AI vendor it >> is that realistic I don't know is will that solve all our security issues I don't >> I mean I don't think it will sol like I So Mitchell was just talking Mitch Hashimoto was just talking about this today and I think that like >> in some ways it's all you know all things uh old or new again you know like yeah absolutely vendoring everything like you know I do I do remember 2013 2014 >> yeah we must return >> that's what I mean is like we were vendoring everything we were having actual discussions around like or at least I remember we were like should we take this full thing like why is this so big we only need this one file and so I do think there's something true there where having like either taking only what you need or the dependencies just getting incredibly small over time I think will help to some degree but it's not going to solve the fundamental

problem I don't think because the vulnerabilities like in an agent looking at them there's time and time again there's a million different ways in which we can convince an agent that this thing is like secure or not and pull it in or we can do you know uh static code analysis or you know a runtime time testing to say whether the code works or not. That is I think the step that needs to continue to be like invested in. The question is just on like how much scope should it be this enormous project that I'm pulling down or should it be this piece either way uh you know most companies are running some amount of you know security checking on the on the um uh the packages that they're bringing in or vendoring that I think won't change.

That's like what you know advanced security does to some degree. socket does some degree, you know, like everyone is doing a piece of that. Um, how we each do that like especially when we're talking to enterprise customers is just like very very different. No, like there's no one wants one single way to do it. Uh, and I think that's always been GitHub's uh unique position in the world. Like I talk a lot to maintainers. I talk a lot to folks about this.

We're we rarely start like a a process and a practice and like push it onto the community. We usually wait for the sort of like RFC process socially or literally everyone agreeing and then we'll cement something in because otherwise >> if it's your role in GitHub yeah we don't want to shape the whole thing. we want it to be figured out. But like how do you balance that uh that like sort of you know role in the industry to keep everything as secure as possible and make sure that you're uh you know you're not going to be compromised as a human because that's usually how it all happens. Uh and not uh you know uh not create a process or lock us into a flow that you know you're not going to like or like Mitchell's not going to like or other open source projects aren't going to like. That's always been a tricky balance for us. And I think that's something that we haven't talked about enough, you know, is we're not going to be able to fix everything for everyone in a way that everyone is going to like.

So tell help us tell us what is working. When Mitchell was talking about um uh the up for what it Yeah. Yeah. I mean like when he's talking to us, I was chatting with him and talking to him about this and I put it on Twitter and we talked to uh also over DM. like we're going to keep working like but I think the important thing is keep I do actually want to hear what isn't working for you uh and as be as specific and clear for your project as is possible uh and to every piece of credit over the many years that we've you know known each other through the industry he's always done that and I appreciate that because there are places that we need to fix up and we hear from him and we'll fix up just like we do >> all other kinds of maintainers um uh but that like that process between you making those types of improvements and being more secure and like creating I forget what he calls it. It's not the uh the the proof process, not the claims process. Do you know what I'm talking about? He has that like uh he his projects have a way for you to kind of like vouch. Thank you. Yeah. He has like the vouch system uh for, you know, saying, "Hey, you should accept my PRs."

That's >> I just built this into GitHub. I don't know. Well, see, but that's the thing is that you you say that and like he and his community really likes this and then I'll go talk to other maintainers and other maintainers uh uh globally and they're like, "No, this this doesn't work for me." >> Yeah. >> And that is the tension, but also the kind of beauty of GitHub, depending on which way you look at it, is we want to help maintainers. So, we create

all these tools to let you have more control over how much you take in, you know, from AI and PRs. But you can also use this, you know what I mean? you can go use this project and if it takes off and becomes the kind of >> mostly standard then yeah we probably wouldn't enforce it but we would add it in because that's the flow that we tend to do you know.

>> Yeah I a lot of people don't know the history of the pull request. >> Sure. >> And like you know like that's how that's something that GitHub standardized basically. >> Yeah. Yeah. It was a a very messy process you know like beforehand and now uh you know the we have the benefit of it being the process you know uh and now we have to go and figure out the next best process or what adaptations change or what does a pull request look like when 80% of your PRs are just coming from your agents and not from other devs you know.

>> Do you like the prompt request idea from Peter? I mean like I think that for each uh like each idea I think has its merits like I'm not I'm not avoiding saying anything good or bad but I feel like I've seen a version of you know uh we have that we have you know um uh uh in entire you know Thomas' you know uh uh startup take all the assets of what you've built and put that in I think that's got great ideas like there's all these various permutations of the PR flow but I think The reason why there's not a single answer is ultimately we're trying to codify trust. We're trying to say like okay if Shawn reviews this, I'm going to trust it because you're Sean or you're the senior dev or you're the whatever. And right now when we are working in a flow where an agent writes code and another agent reviews code and then Kyle goes and looks at it, the trust is kind of diffuse. Uh and most of the tools that we're talking about are talking more about verification flows.

We have more assets to look at. So, I can probably say whether this is a good PR or not, but that still doesn't solve, I think, the human problem of I'm looking at a PR and I want to know if I can trust it. And we're still we still tend to use human signals for that. You know, uh >> Mitchell approving it or Kyle approving it or whatever. >> And so, I think that's I think that's why most of these uh options haven't really solved it is because uh it's a social problem ultimately. it's a it's a human problem to review it uh uh and agree or you fully trust the tool and you're imbuing that tool with full trust which I think in some cases that absolutely exists and >> so so like you know in the same way that there will be a tipping point in society when we don't allow humans to drive anymore >> because machines are measurably better than humans I I'm looking for that tipping point right >> like mythos is ridiculously expensive someday we'll have mythos on a desktop I don't know Uh what what does that change the equation?

>> Uh I think it's more like uh uh I took a Whimo here uh and I was on my phone and not looking around uh at all. Uh like there are other uh self-driving uh vehicles that I would not trust while like staring at the road. And I think that that trust is something that is >> this a zoo stick like what? >> I think that is both. I think that is both uh you know like >> there's zuks in this robo taxi. That's it. That's >> Well, I mean, depending on what level of self-driving, you know, but my point is sort of that I think part of that is uh you know, I strongly believe that that's like a mixture of verifiable proof, you know, like how many accidents, how much data, and so on, and the human aspect of how I feel when I'm in this car, what it tells me, etc. And so that's why I think some of the like uh some of these uh uh some of our AI tools tend to um imbue me with more of that feeling of trust even if the data says this is 100% accurate.

You know like I feel like it takes more time for us to go should I trust this or not? And that's in the soft sense of like startups with high agency weekend projects and open source. And then there's enterprises and regulated industries and everything else. And that is an even harder problem to go solve because even when it is fully verified, not only do you have to have trust from the humans on the team, you probably have to have trust from multinational uh uh multi-governments around the world, you know, regulating agencies. And so that's where I feel like until we tip over to your point like on the sort of like human EQ side of it like I feel okay like this feels okay like I've been proven enough uh then the ball will start to roll a lot faster uh where we'll end up getting to the okay we can trust this and feel good about it in the most difficult of cases >> you know if human trust is the thing that matters uh I feel like GitHub as the developer social network could maybe do more there like voucher is one system but like we have star counts and then we have >> contributor rights and that's it.

>> And like I feel like there should be more in that space. I don't know if there's any other design decisions that >> I I mean I think that like one of the places that we don't really expose right now in this sort of way is um like uh some degree of like hard trust and support which would like for me is like sponsors is a good example of that. It like costs you something you know to prove that I I believe in your project and I like trust you to some degree or I want to support you at the very least.

>> Okay. self payments for open source. So, why not? >> I mean, like like I think that I I think that like as we keep moving forward, right, there's more and more projects where I I'm like adding more and more dollars into sponsors personally because I want to like support them, but I also like know of, you know, I've probably never met them in person, but like I know enough of their work that I want to support them. I think the thing that I don't love about stars or commit counts or anything else is like ultimately even with all of the various like abuse and despaming and dduplication work that we do or anti-abuse you know work that we do these are all like not active social signals they're passive ones that are ultimately gamifiable and you may trust me but another open source maintainer may not and on what heristic should you be uh trusting me that I think is kind of where some of our thinking is right now. What signal from me is most important to you. you if you can define that potentially like honestly like in an agentic workflow like that's what we see some of these open source projects do where you have you know GitHub actions and then you have like an agentic workflow that's calling AI and you're setting these rules like if Kyle has submitted and gotten accepted PRs across any given project and has a social handle tied to his account in GitHub and that social account's older than certain amount like really complex measures that matter to you because most open source projects have that heruristic built into their heads if not written down in the contributing guidelines. You could take that and then go apply that and then just say, "Oh, we're not going to accept this PR."

Building something that is, I think, malleable to everyone's needs, uh, uh, is a little bit better rather than going, "hm, this account's too young because what happens, the attackers just go and build go and create a multitude of accounts and they wait until it ages up. Needs to have certain amount of stars. That's how star inflation happens. Need to have certain amount of repos with PRs. they all just create repos and submit PRs to each other and then they you know come in and do something nefarious and so uh it's hard like it's hard to find the measure. So I think we're we're looking more at how can we provide you tools so you can kind of choose what's best for you and of course we'll give you some standards but the trust vector uh gets down to like I don't know some version of like human digital ID like everyone's been talking about like how do I prove that it's me on the internet? Give

me your eyeballs.

Exactly. Uh uh I I got to keep moving on on on topics, but obviously I can go all day on this because I mean I've been involved in GitHub and open source my entire, you know, professional career. Um stars, >> yeah, >> very superficial. Everyone knows it. >> But I think, you know, time to 100,000 stars is the fastest I've ever seen, right? Like people just reached that in I don't know, months. >> Uh and then like at the same time, like I don't trust it, right? Like how many of these are real or bought or like whatever? Um, I don't know how to ask this, but like what can we do about it?

Like, you know, is is stars broken? Is stars fine? >> I think that there's kind of two there's like two pieces. Obviously, like we're constantly like trying to find ways in which like your uh users are, you know, producing spam, which would I would include like be like only doing star gamification. When we find them, we pluck them out, you know, and we uh uh >> as like a whack-a-ole. >> It's 100% like a whack now like powered by AI to be helpful. But I I think more so what I'm seeing is uh a lot of the like fastest time to X, you know, tends to be because we're now inviting so many more people into like software development on GitHub that like the zeitgeist is just swarming. Yeah.

>> You know, >> it's not just developers >> and it's not you and I like, you know, like however you want to say like what a developer is, you know, it's not just folks that have been coding for a very long time. It's folks that have maybe started coding or only joined in since the AI era. And >> what's the latest Octo number? I I know 80 million was my last memory that like a number of developers on GitHub.

>> Oh, we're over 200 million now. >> Yeah. Okay. Well, you see >> Yeah. Yeah. Yeah. Yeah. Yeah. Like over 200 million developers now. >> Yeah. But but it's not developers, right? Like it's it's people with a GitHub account. >> So like so this is this is the biggest debate that like I would say like everyone loves to have at GitHub at this point. From my perspective, right? I think that there's there's clearly a difference between like professional enterprise developer, you know, and then developers. But I think that I think that the idea that, you know, uh we should be like, I don't know, splitting hairs or segmenting developers in the early era of software development is like not worth our not worth the time.

>> You get into gatekeeping like >> 100% like 100% because I mean I wasn't a developer when I started writing code, you know, I was going to I I made I like cloned the thing like seven years before I learned to code and then I and then I wrote about my learning to code journey and people called me a fraud cuz I had a GitHub account >> and I'm like well no I just use GitHub but I don't know I didn't know what >> I mean I I remember that like I remember those sets of posts and like that's that's So I fight very clearly on the line of like if you create code, if you have an idea and you create it into some way of like I'm I'm going to run it and use the app right now. You may still use AI in that moment. But that's okay. At some point you're going to do the next thing. You're going to create a you're have to learn about this database. You're going to fix a bug.

Whatever. Like we're all on some same journey. And those people are also hearing about the great new agent skill package or a new CLI tool or a new whatever. And those projects are going up because you want to be a

part of this moment just like I wanted to be a part of the Ruby community when Ruby was popping off when I started becoming a developer and now I can just click the star button. And so I think that yes, there's clearly some amount of like, you know, spamming and gam gamification that we're working against. But I really think we're just seeing this whole new cohort of folks that are moving from technology to technology because they're not working on a 20-year-old software application. They're working on a side app that they built on the weekend for their friends or for their new idea or whatever. Yeah. Uh and that's how you see these enormous charts going up and to the right with with stars.

>> I think something that's remarkable is the persistence or that like GitHub extends to those folks. Usually when I see platforms go into a new audience, they usually have to like have like a second platform with a different name that like wraps the main platform. >> Uh but somehow GitHub has been able to sort of persist and extend and it's friendly and whatever, you know. So it's it's nice. >> Yeah. I like uh uh that's partially why I think as we've tried to move into like um I don't know more like low cody things, you know, like we we you know started working on Spark as like a way to like build an app and run it.

>> I think that the reality is that >> we anytime we try to like kind of put even a veneer on top of it without like uh like when we put a veneer on top of something, we still always show you the code. That's kind of like a tenant. we're never gonna like hide the code from you ever. Um because what like yeah is the whole point, you know. Uh however, I think that what we learned with things like Spark is that really the value of Spark for most devs is like easy runtime. And you may have a runtime or a host that you're going to use for that or you just build something and run it, but like the package of making that like even more simple isn't really needed like for folks that are trying to build software uh and not just trying to build like an app which is like slightly uh slightly different uh a slightly different goal. So I want to get you in.

I want to get you comfortable. I think the best thing for me as like someone that did not like you know traditionally come into software dev way way way back. I want anyone to be able to like breach that chasm and not be in the, you know, I don't know. I feel like we're we're still in an era of like STEM, STEM, STEM. I've got a 12-year-old and an 8-year-old and it's like we got to get them into STEM, you know, over and over.

Uh and I uh I like I do I do the things that good parents do. I was like, "Oh, we want to do coding. Yes, I want to do coding, do coding classes." But now they're just not afraid of doing software. And that's I think the thing that's honestly kept me at GitHub for so long. anyone should be able to go and build a thing just like I can go change a light switch in my house. Like I'm not gonna go into the breaker box because I'll probably kill myself, you know, but like I can go change that light switch.

Everyone should be able to go and say this freaking app doesn't do what I want. Like I want it to work like this. Yeah. And that I think is what's kind of kept us all connected with GitHub through the years and some you know and like during the easiest of times during the hard times because of that opportunity of like we're the home for all developers and we want everyone to be able to have that feeling that we've had of I had an idea I created it and holy like you know here it is.

>> Here it is. Uh all right I'm going to try to do more spicy questions. Great. >> Is it an easy time now or a hard time? >> Oh at GitHub. Yes. I mean it's a hard time like I mean like it's a hard time and also like I was just with my team and I said this is also like the best and most exciting time that I think I can remember like at GitHub because >> best of times worst of times never >> because we've you know like we're talking about Octiverse reports and like usually we do an Octiverse report once a year and we look at the numbers and we say oh my goodness like I was at Universe in October saying this is the fastest year of growth that we've ever had right and now we're doing more in a month than we did in a year last year.

>> Uh you're talking about PRs, >> commits, >> PRs, kind of like you name it. By roughly every measure that we're looking at, there's some amount of sort of growth that is much much bigger. And that is breaking our system in new ways, not old ways. Like, you know, web hooks were always notoriously unreliable over the years. You know, >> whose fault is that? >> Like like not anymore mine, but for a period of time, I'm sure you could pull up a tweet that was like, "It was me.

I'm sorry. Uh, but like now like that got rewritten at a scale level that is still working and is not having problems today. Now what we're finding isn't just the like isn't the the the simple stuff that folks are on the you know sometimes on Twitter or on the internet are like hey like why is this like this? Sure there's absolutely you know silly problems that shouldn't exist. But now we're talking about like unique novel permission problems that happen only at a scale across all different objects or whatever that now we have to go rewrite this underlying system. And so it's uh uh there are problems that yeah caught us offguard which I think I said I mean like the growth is astronomical but also we're making such material progress in that um uh that I'm excited once we're you know once we've kind of like re uh reimagined the underlying foundation layer or pieces of it at least what's going to be possible when it's not just all of us and all the new people that are being developers and all of their agents in all the tools like working together um uh because that'll still happen in that you know uh in that GitHub uh you know uh tool that GitHub community but it's a hard it's a hard day anytime we can't give you what you're looking for we have the same problem internally I mean we operate through github.com of course we have backups when things go down and whatnot for our own operations you know but we feel it too you know if it's not working it's not working for us and that's kind of like the promise of dog fooding for GitHub. It's always been true. We're using the same tool you're using. We're not using a super secret version. Uh and so we also need it to be great for us, for our customers, of course, for open source. Uh and now, you know, uh uh an exponential growth of agents uh doing it too.

>> I wanted to load uh for for audio listeners who who maybe haven't seen your tweets, whatever. Uh so 1 billion commits in 2025. Now it's 275 million per week on pace for 14 billion this year. Uh if growth remains linear, is that still the pace? >> Yeah, it's I mean it's it's speeding it's still speeding up. Yeah. Yeah, exactly. This was in April. >> All right. So So basically you have 14x growth, right? Year on year. Uh and I think that's a scaling issue. I think uh I'm going to like try to really steal man this thing, right?

>> People have experienced 14x growth. They haven't had your downtime. And that's like can we go dig into that? Like why? Like what's the what broke? Um what are we doing to fix it? Like you know just anything for the community to reassure them. >> Yeah. Yeah. I mean so there's like I was saying there's a couple different

places that we've seen the uh the the growth issues. Some of the growth issues uh which is why we're I was talking about uh you know pushing hard on more CPUs is in actions in particular. Uh more tools more agents more PRs mean more builds.

more builds need more CPUs and so we are expanding through not just our data center but obviously we were talking about moving to Azure and moving to like adding an additional cloud compute because we simply need more CPUs uh not uh not as much GPUs like we definitely need GPUs too but now CPUs are becoming a factor you know uh underneath the hood when it comes to uh like some of the underlying services we've been breaking up over the years our database infrastructure so that way we have uh more cognitive separation between the various services. The place that we continue to have pain is in uh permissioning. And so right now many of our permissioning layers sit into a database that we like internally call my SQL 1. And old hovers will know what I'm talking about. And so like we've been pulling things out of my SQL one for many many years. uh because like and we use you know we use the test and we use other technologies to shale was born from this >> 100% Sam uh you know old harbor and friend I mean like and so finding these opportunities to like break this out and then do that globally the other thing that I think is interesting in um uh like both a unique opportunity and tricky is we also run everything I just talked about in a like blackbox container with GitHub enterprise server for people that work on on prem. So we take everything I just said and we also do it on prem and we also do all of that and we do it in a data resident setup uh for customers that need to have their data in a single location. Each of these has the unique characteristic around how we're sort of storing that data uh in my SQL or in a permissioning setup. That's where some of these outages have occurred where you're seeing it more like across the board rather than just like the one piece isn't quite working.

Exactly. Exactly. And so part of it is that I think there's been some other places where agents are much more or more projects appear to be moving towards monor repo versus we were going the other direction for many many years in the industry. Repos were smaller but there were more of them and now we're seeing the opposite. Repos are bigger and there's uh not fewer of them per se because there's new growth but like we're just seeing many more big repos.

big repos uh big monor repos have always had like a unique performance problem uh like because each one uh is slightly different if particularly if the underlying blobs are incredibly big inside the repos and so we've been a ton of work that you pro like most people haven't probably experienced uh unless you're in this case of the monor repo uh but that git uh uh infrastructure layer improvement does help the overall uh system because um many of the improvements that make monor repos work better, make all repo infrastructure work better. And so, like, I can kind of keep going like down the line where it's another thing where, you know, we're moving out of uh uh we're changing how we do um uh like I'll just say like job queuing for lack of a better like explanation, like changing the underlying technologies there.

>> I I spent two years being a job queuing guy. >> And so, like it's a kind of a little bit of like a little bit of piece by piece. And it's mostly because as we were as it was built, we built everything in a way that uh uh assumed I guess in some ways that the size of the pipe of work was going to remain the same. There's just going to be more people coming through each of those pipes. But instead now in places where a git push was generally a certain

size for example is now like no longer true.

>> Oh yeah. you know or uh like on the average online commits >> same thing with PRs you know like PR is like same thing uh uh and like we've like talked about optimizing that and making changes where like and there were like technology choices that did not work there you know and it got slow and it didn't it was not fast it did not do what users wanted and so we've been like reeling that all out you know going okay that's just not right let's stop putting you know uh good money after bad and do it the do it the right way or the right way. Now, so there's it's a it's a lot of things. Not quite like uh when I've experienced scale at GitHub historically, it's almost always two options that we've used. We go vertical scaling, particularly with databases, right? And we go horizontal scaling. Oh, we just have more people using the service. Great. We're going to add more servers and we rack them in our data center. We use it in a a cloud. And now like we're sort of in a like diagonal where like vertical doesn't really work anymore. horizontal isn't work either because like we're all we all have some CPU or GPU constraints in the world now and now we have to go and like crack open services that have been running for 10 or 15 years and go okay the rules of this service have like legitimately changed and now we have to rewrite them.

None of this is an excuse. This is like we're we have to do the work. We have to make it better. >> I mean actually as an infra guy I'm like that this is like one of the most fascinating scaling challenges I've ever seen. That's like that's that's the thing that that's the thing that it's hard for like when we weren't talking about it publicly and I was like I came out and I was like hey I just want to explain what's going on. Part of it comes from a very old GitHub like ethos which is it's our it's our uptime it's down. What like I know you're a developer so you're you're inclined to uh you know want to understand more what's going on but at the same time like us going hey this service didn't perform the way we expected and now we have to go change it. We weren't we're not trying to hide anything from you in that. that well that's our problem because you expect us to be up and I think that's like really baked into the core origins of GitHub and so now what we're trying to do as a team is do all that work and just tell talk about it more just share you more technical details write these blogs write the posts get the engineers who built it after they finished the work just tell you okay this is what we did I think that's the contract that we want to bring back to the community and say hey we're still very serious about what we're doing we haven't been telling you about each piece. So, let's do that. And we're going to keep, you know, building this and scaling it in a way to support the if it's not 14, then it's 30 or it's 50 or whatever the next, you know, uh, exponential growth is going to be.

>> Yeah. Um, first of all, fantastic answer. Um, I mean, I think >> and I apologize in advance if like any of that is like slightly incorrect just simply because I'm not, you know, uh, the I'm like still in the weeds with this, but it's not my, uh, day-to-day. But like that's the thing is we're all looking at it to that level, you know? uh you know and like obviously if people want to help they can join. Absolutely.

Um so like I think that is uh good. I I think people also just want to know like when are when are you through the thick of it right like is there have we identified all the issues? Is this just >> never ending like is git broken like do we have to change the git uh protocol like what like how much is breaking right like it's been a while. >> Yeah. Yeah. And so I think people do want to know what's the path back to the the reliability that everyone

expects out of GitHub.

>> Yeah. So I mean like our uh our availability in in rec like recent few weeks has been much better than the three weeks before that or the 3 weeks before that and so forth. So a lot of these improvements are still very much paying off for us. I think that um we're still working on that, you know, uh uh that database piece that I mentioned and that just is a little bit physics like a little bit of time to get it uh to get it fixed up. Um because we have to the way >> uh my my my uh the answer I had in my head was call YouTube.

>> So YouTube ultimately >> they also use Vest. >> They also use Vess but the uh uh >> like whoever was the guy the scaling guy YouTube you know what? >> Yeah. Yeah. that's that I believe went to planet scale and was a part of planet scale too but like >> oh you mean >> uh I think so yeah yeah yeah and so uh and so >> he's super based now >> the whole Postgress drama >> yeah yeah yeah totally so I mean like some of it's that I think the other piece of it is um uh our move to get additional compute will alleviate a fair amount of this particularly on the action side because a lot of the underlying um outages is actually related to uh >> tell you that actions this It's the root of all evil.

>> I mean, it's all it's it it has its pros in that it's the core it's the core compute layer for either CI side project. >> Actions. >> No, I don't know. >> I mean like actions >> I pay a lot for for comput, right? >> Yeah. Yeah. I mean like actions is like definitely a a a piece of the overall business but I would say that like >> we ultimately also give away so many like minutes you know as part of our entitlements as that that I was saying everyone's using it we we talk about it as CI/CD but the reality is people use it for CI/CD and various processing and automation exactly and so I mean like part of it is also that like compute piece that uh that is also alleviating some of our availability >> my abuse of actions I have been scraping for every day. Uh and just like I just >> Thank you for your service. Uh >> but this is also how I track actions on time.

>> Sure. >> You know. Yeah. So anyway, >> uh so I mean like some of it's going to be that I would say that like each month I expect you know in the next 3 months you're going to see like fewer and fewer moments where we have an availability problem where things are going to go down. And that's not just it stopped. It's that we're still experiencing faster growth than ever before. It's just that those underlying improvements that we've been hard at work on uh are finally paying off. It's just that there are improvements take it's less about like these incremental improvements where you make a small change and you get this big output. It's now material change that takes a bit of time and then you see a step change in our availability.

>> There's the thing we used to do at Amazon. I don't know if it's like a thing but like you know automated software verification or simulation of load testing and all that. Like I'm I'm just like at this point you have a whole map of GitHub >> and like well you can assume whatever growth rates on whatever dimensions that you care about and just run it through the system right like I feel like there's a way to >> I don't know have a systems model of GitHub and like see what breaks but obviously I'm I'm not that close to problems. Yeah. But I mean, yeah. So, yes, totally. And I would say like that's been the journey and work that's been happening since like I would say November to now because October, right, was the time where we even said like, oh, look at the growth and like and then you start to see the chart like really really pick up. It's like, oh, we tested it at, you

know, n amount of scale and now it's at like n cubed maybe, you know, like in some uh in some uh vectors and so now we have to go and build it, you know, that way and make sure that it can handle all of that scale. Uh, let talk co-pilot.

>> Yeah. >> So, how many original creators of Copilot are there? >> Oh, jeez. >> I count like 12. >> Yeah. I mean, like I forget like all joking aside, I forget the number of people that were on like the original like uh GitHub co-pilot team. Uh, but uh there was a bigger group. >> It's Alex. >> Alex worked on it. Ugo worked on it. Like there's a like a bunch of people >> and their entire management line. Okay.

Uh so so like uh you know enormously successful at its in in its in its day. I think the last number I think Mario came to my conference uh and talked about the hundred million mark. I think most recently 300. Uh I might be out of date as well there. >> I don't think we've shared the dollar amounts. Yeah. >> All right. Cool. Um just like what's the state of co-pilot? Uh it's it's obviously as a concept brought into more of Microsoft.

>> Yeah. >> Uh but just add GitHub. >> Yeah. Yeah. I mean so I think you know one of like one of the challenges is that we had with co-pilot right is that we came out the gate with code completion it was you know super great powerful etc and then what we initially worked on after that sort of like initial year year and a half was um going after fine-tuning because you know our customers the industry on the whole was really talking about okay well like how do we get more um uh you know more correct or performance out of this. And so we were working on a whole bunch of efforts to do fine-tuning on uh larger and larger code completions or like next edit suggestions with fine-tuning etc.

>> And let me clarify uh is this fine-tuning one model or per customer fine tune model? >> Per both but like but like fine-tuning one model for the overall like uh use and then fine-tuning per customer that wants this as like a service effectively. And around that time is when uh you know the next generation of models came and that's around the same time that you know all these other you know AI uh coding tools came to be because the models really really sped up and so everyone kind of like will ask like well what happened to GitHub copilot like there's all this time and I would say that we were on an era of going okay we want to improve everyone's results and so let's focus in on fine-tuning because that'll give us these better results. And then the models got better. And so then ever since we've been really on this kind of journey to go okay of course we have like this great code completion and we've done a ton of investment in the better underlying models that we have uh you know post-train better uh next set of suggestions of post train language specific models all the stuff that kind of like sits in the ether of GitHub copilot is code completion but also have now ha now have uh like a single underlying uh SDK and harness for our uh our uh coding agent, you know, co-pilot ultimately, uh, the new CLI, the new desktop app, cloud agents that use the same SDK. And so there was this moment of, you know, both, uh, really really trying to figure out what our customers want, models, uh, sherlocking us a little bit, then going and saying, okay, what does everyone ultimately need? And what we think is that it's not solely about the code generation. Uh it's really about having the ability to use these um you know coding agent brained uh uh harnesses or runtimes across not just the coding experience where I'm going to like send a bunch of tasks out or I'm going to use fleet to um uh to break up a single task or autopilot similar to goal, you know, all this stuff. Uh but also how do I do that for all of my security remediation? How do I do that for every GitHub issue that comes in? just stick

a coding agent on it just to say if it's possible. uh how do you know go through my repository and see all of my documentation and extract out like okay this doesn't actually match like that amount of sort of AI coding agent automation uh I think is a big part of what we see when we're looking at okay we're still kind of going through a similar but very different flow it's just all happening at the same time you know like there's not really the same like I'm gonna create an issue to track my idea of building this you're probably just gonna go like do it you're gonna say hey just build this Right. And uh uh there are still tons of uh open issues and projects etc that are using uh issues like Peter and OpenClaw you know to be able to sick all of his agent um uh on that that kind of infrastructure layer and a really really great coding experience that allows you to handle the sort of multipplexing uh aspect is what we've built are still building uh you know with GitHub Copilot. Um, and so for folks that, you know, haven't really used GitHub Copilot since the thing that got them excited about this, you know, which I like I get, I really encourage you to like look at especially the GitHub uh copilot app. Like that's my new daily driver. Uh, uh, I obviously like if you prefer the CLI, also the CLI, be able to use all the models, the bring your own key side of it. Like we're still improving our own models and using those too. Um and uh it's just like a very very different experience but I think that broader sense is uh of like software development and how coding agents can help throughout not just writing the code uh or even verifying it or deploying it you know uh is where we have this unique uh uh angle. The other side is the context piece like >> oh god >> I mean like we're still it's like one of those things where I think the you know the final thing that will let me ultimately uh uh feel complete at GitHub is like when we have this ability for GitHub to act like Kyle wants it to act or Shawn or whatever. Uh and we all codify that in rules and in memory and everything else >> and that's an open research problem right 100%. But like if we can even just do it where my team without me having to codify everything and as our methods shift on purpose, you know, to be able to have that full experience and all the understanding of what's happening in my dependencies or open source, uh that feels like a big place for us to be able to continue to provide something really unique and valuable uh with GitHub copilot.

>> Yeah. Is there a form factor that we haven't explored? You know, I think like uh you know, we did code completion. Yeah. Then we did kind of let's broadly call it agentic IDE >> which cursor uh famously popularized and then now it's now it's all about the sort of >> agent orchestration backon agent whatever whatever um and then there's the security review I feel like everyone has like just thrown agents at everything >> the entire SDLC has like covered with agents >> um >> are we like at the end of history here basically you know like you know is is is it just refinements from here on out >> I mean I think that We're all still in such this like hyper myopic era of AI where the reality is that for various like boring security and governance reasons at least for most people's work why is my coding agent even if it's all background agents background running not like losing all the context that's available to it across everything that I'm doing outside of coding.

>> Yeah. you know, like I I think the most interesting thing to me in AI is actual ambient AI, not insert, you know, assistant name thing or like I've tried just about every pin in tool and whatever and they don't work the way that I'm looking for them to work because they are just trying to capture and then they are trying to codify and then recall. And I think the thing that I'm looking for is back to the very beginning. I'm looking to be building out the next version of web hooks or like implementing a new feature and it for it to know every spec doc, every email, the conversations that I've had online, everything about how this could be implemented and be able to like use that as part of its decision-m and none of these tools are ultimately doing this. So I think that it's as if like software development were was a single lane task was like it only needs a developer once I once I

write the perfect code will be done here. But that's just never been true. It's all the context of the other team members what the business is doing what's popular right now. And I think that's this huge opportunity for us to go much broader than really really excellent coding agents, you know, and that is honestly why I think OpenClaw has been so interesting is that sure it's connecting to all the data uh sources that Kyle the human cares about. And now my question is like, okay, how can I take all that and use that every day as a software dev connected together, not just have a new way to kick off a coding agent?

>> And that's where we're at. we're saying, "Okay, I'm going to go use this CLI under the hood or this SDK." But that's not what I'm talking about. I'm talking about I'm having a conversation with you. It downloads the podcast and it realizes, "Oh, Kyle, sounds like Kyle needs this app or this thing or this >> that level." Exactly. That level of that level of connectivity I think is where we still have a ton of ways to go in software because then when we have that red thread we want to pull that idea it can not only use the perfect way to write that code but instead all of the sort of taste and judgment calls and uh you know expertise that I've earned or that we've earned as a group uh and use it as part of the actual implementation.

>> Yeah. Um, you know, the extreme of it is AI runs your life, right? Um, and I think there's a scary inversion of control in the way that I literally doing it in the way that developers mean it in terms of frameworks like you know the the Hollywood principle like don't call me, I'll call you. >> Yeah. >> Um, like there at some point there's an inversion of control where like you should you stop telling what the AI what to do. AI tells you what to do >> and like that's a little bit scary but also like maybe better. I mean, like, you know, uh, uh, Nat, uh, I think Nat Freeman shared this in like a Stripe event, you know, like talking about his Open Claw was like he connected Open Claw to his cameras and it was like watching >> directed his Uber.

>> Uh, and >> there's a degree of this where I was like, I actually would love OpenClaw to tell me to like drink water. I don't know that I want it to be uh changing where my car goes, but I do think that's kind of what I'm talking about, which is it needs to have so much more information at its disposal for it to be helpful to me. And I still don't think we're like anywhere near talking about AGI. I'm just talking about every time I have to tell you something I care about that I've ever kind of said or I've said a dozen times, it it should be able to know that, codify that or gain access to it. Um like the dreaming ideas like are an attempt to kind of do some version of this, but I think there's a much more proactive angle that uh will help software devs if if we can, you know, test that out a bit more.

>> Yeah. Yeah. Yeah. Well, the other thing about Open Claw that reminded me is Microsoft has a CVP. >> Yeah. >> Dedicated to OpenClaw. >> Yeah. >> Why? >> Because you don't think they should. >> I don't know. I mean, I I I think CVP is a high title. >> Yeah. Yeah. >> Uh what um why is this so important? Like, you know, Microsoft doesn't even own OpenClaw. >> Yeah. >> Like what's what's >> Yeah. I mean like so you know we're talking a lot more about this at um Microsoft build this year too. I think like the main thing is that what openclaw has done is it has made this connection for people to have access to the resources that you have access to and be able to do things for you in a way that previously people were trying to codify into their own agents. And so when you think about it like in the work context, wouldn't it be great to have a claw like object that I could actually run on my work device that or had access to my work assets made worked well on Windows like

what that would look like.

And so I think that openclaw has become a personification of like a valuable agent that understands me because it has access to all of my information and it can use a computer uh and so thus it can you know do a lot more than uh just a task oriented process or like a you know a chat tool etc. Uh and that's like a bunch of you know the the goal of build right like we're at build this year trying to take a very different approach of you know it's unapologetically you know aimed at developers we're trying to show the like bigger investment to not just say hey like you said why do you have a CVP of openclaw well because like one of the problems that we have right is that our agents if you install them not on a Mac mini or not on a hosted device you install them on a a personal device or a work device, we need better sandboxing at the OS level. I need to be able to use that claw and not like get fired. And so Microsoft is like, okay, great. Let's like do that, too. And then it's okay, well, where should I be able to talk to this agent? Should each of us just have a claw available to us at work?

>> Probably. And so there you go. You know, uh continuing to contribute a ton uh to the open source project, too. like Microsoft I think uh as I've gotten more and more uh uh uh information like there's so much investment into the open source uh projects themselves that for whatever reason just I think there's like this uh they don't want to come off like those teams don't want to come off as like taking any credit or getting any recognition but so many of these core contributors of teams are full-time just pushing into open source projects uh in like I think that's that kind of shows the difference between like well why are we looking so hard at something like claw. Why are we looking at sandboxing on Windows? Why are we looking at cloud versions of sandboxing? Why are we looking because ultimately like we need more platform components. We don't need everyone to be building the same exact like topline product. Uh and so if we're building for builders, that requires us to give you all these components and tell you what they are and how they work and why you should be interested versus only delivering that single vertical like over and over and over again. Yeah.

Um I I I think like my maybe one way of framing it Yeah. is that Microsoft is the original operating systems company >> and here's the new operating system for AI. >> Yeah. Yeah. Yeah. I mean like you know I I think that uh we are also in an era where we are like we need to help build that bridge you know like all joking aside like operating systems need to look different than they looked 5 years ago because it's not just you using them anymore.

Yeah. >> You know uh and that's changed the whole idea. It's not okay my claw is going to create a user account. doesn't work like that, you know, and so just like uh just like all of us, we all have to look much much more deeply in the stack all the way down to like the silicon layer in Azure to be like, okay, well, what do we need now? Because the workloads are different. It's not just okay, we need more inference. It's okay, well, what type of inference do we need? What type of compute do we need to run these agents or run these agentic flows? Uh it's a really interesting kind of like multi- uh multi-layer problem. Uh versus kind of uh I would say you know software in the last you know five or six years we're all going to our events and we're kind of saying a version of the same thing. SAS product has new SAS thing.

It's the best SAS thing ever. >> It was boring for a while >> you know and so now it's like oh my goodness like

we're at physics. >> Yeah. >> You know we're at physics problems. Uh and that's exciting. >> Yeah. I mean, we're now trying to make like semic uh room temperature superconductors. >> Uh still. >> Yep. Yep. >> Uh that's that's that's never going away. Uh no, I think like that that's a really good overview of like everything.

Um I I think have I have we left anything unsaid that you wanted to really get out there that we should cover? >> Yeah, I mean, you know, uh I I'm really excited by like for folks, you know, checking out um checking out the announcements that we have at Build. Uh uh like go, you know, you can go look at them online and take a look. I think that I'm hoping that it's driving like a degree of curiosity and interest because there's such this big shift that we're making at Microsoft uh for developers where if you're a daily driver of like um you know a Mac device or a Linux device and you're like okay I don't use Windows I mean like there's improvements that are being made that I think are going to surprise folks to just be like oh that's like the they really want to do that. Like not I'm talking for developers. I'm not talking for I play video games on the weekends on my Windows computer. I'm talking like my daily driver. Uh like all the way from that to okay well what is it like to build an agent or build an app and deploy it and run it at work in particular. I think that is a big piece of it where I talk all the time uh uh with the team how I build on the weekend should be how I build at work. But if you're working in a Fortune 100 or Fortune 500, you're probably not viating an app and then shipping it to some service. You got to go through security and compliance. How can we move just as fast at work? Uh, and that's I think something that uh uh we have a bunch of different offerings for to give you that same sort of agility and power but in the work context. And then I will tell you like I've mentioned it a couple times and uh it's very freaking cool. uh like if you are in the M365 land in any way, check out work IQ, check out Foundry IQ. These little like uh uh oversimplifying it context engines are wild good and like we've given them to our uh developers at GitHub. we've given to employees at GitHub as we've used these tools to be able to just ask questions around everything that you have in your work context and with Foundry IQ be able to just do the same exact thing across all your existing stores like what not not move to new tools just connect them in it's surprisingly powerful and you still your boss is still not going to get fired and it's not going to turn it off because it's leaking all this private information that is the trick that I think uh is sometimes getting lost when we're talking about all these uh all these great new platforms because I can use them. I'm like, "Oh, this is super powerful."

>> Oh, and I can't like I can't use it. Like, and it's not because I'm at work at GitHub. It's not because I'm not allowed because they can't do all the things that large complicated companies need. And so, uh whether it be like I said, just the kind of interesting daily driver curiosity all the way through to oh my gosh, like I can go use this at work tomorrow potentially. uh and have that context layer, have that intelligence. Uh it's a huge it's a huge shift. Um uh and so, you know, check it out. I'd love to hear I'm like I'm not shy on social. I'd love to hear feedback like what's working, what's not. Um uh but hopefully surprise folks a little bit.

>> What I'm hearing I mean so first of all, I think that's that's great pitch. What I'm hearing actually is that you should put the work IQ people next to the co-pilot people because like the the exact pro context problem that you named Yeah. they've solved enough for you to do your job which is nuts. >> So like the the the thing that we are lit like that's literally what has been happening the last several months.

>> I already forecast you >> is like like totally cuz like you're totally right the code the code and the code asset

problem is a little bit unique but otherwise yeah we're all working with each other now it's all just context. Exactly. >> Yeah. Yeah. Amazing. Uh great. Uh I I'm going to be there. I'm going to be doing a couple sessions there. I'm going to be interviewing Satia. I know >> when I um first started the pod though I had like Jeff Dino on Jeff like it's like hall of fame of like I want to meet someday Satia's on there so like what should I ask Satia?

I I mean I think I think that the best question to ask is what he thinks is true in like two or three years from now, you know, like it seems like such a throwaway question, but ultimately uh the way that the way that he is looking at this AI problem, in uh inference problem, token problem, and what we're how we're actually going to be working uh I think you can see some of the recent shifts that have been happening inside of Microsoft to kind of drive us to a place where it's not four, five, six, seven, eight different things. It's not a lack of context everywhere, but like why is this uh you know sort of approach in two years going to uh pay off? Because that I think >> wow that's a bold Okay, I'll ask it.

I'll say I'll say prompted by you but absolutely uh it's a bold question because um you know I think there's a lot of uh doubts to be honest like externally and and so like yes I I I want like a straight answer from from him on that I think would reassure a lot of people and honestly like give me a lot of food for writing. >> So uh thank you so much for spending your time. Thank you for doing what you do. I think like you you know as a co you don't need to be the external face but like because you are authoritative because you you have so much background with GitHub and it's so authentic like we on the outside feel it. So thank you for that.

>> Of course I appreciate it. Thank you so much Sean.