**EPISODE 857**

[INTRODUCTION]

**[00:00:00] JM**: Software projects are organized and planned using project management software. Examples of project management software include Jira, Trello and Asana. There are hundreds of tools for managing a software project because there are infinite ways that a project could be managed. Google Docs change project management by allowing documents to be easier to share and collaborate on. Newer SaaS tools, such as Slack and Dropbox and Notion have taken the design lessons from social networking apps to make enterprise software more engaging.

As the tools improve, our project management strategies change and new software tools emerge to fit those new management strategies. Kurt Schrader is the CEO of Clubhouse, a project management tool for software engineers. Kurt joins the show to talk about the history and the future of project management tools. He also discusses the engineering challenges of improving performance on a complicated web app, which the company he is building, Clubhouse, is quite complicated. It's required a lot of performance optimizations. Project management tools often have to load hundreds of small objects on a page, and in the case of Clubhouse, this required performance optimizations in the Clubhouse frontend JavaScript library.

The new Software Daily app for iOS is out. It includes all 1,000 of our old episodes. You can find all of our old episodes and greatest hits and topics. You can comment on those episodes and communicate with other members of our community, and you can become a paid subscriber. You can listen to ad-free episodes by going to softwareengineeringdaily.com/subscribe.

Altology is the company who has been developing much of the software for our newest app, and if you're looking for a company to help you with your mobile and web development, I recommend checking out Altology. They've done quite a good job with the Software Daily apps.

[SPONSOR MESSAGE]

**[00:02:11] JM**: You probably do not enjoy searching for a job. Engineers don't like sacrificing their time to do phone screens, and we don't like doing whiteboard problems and working on tedious take home projects. Everyone knows the software hiring process is not perfect. But what's the alternative? Triplebyte is the alternative.

Triplebyte is a platform for finding a great software job faster. Triplebyte works with 400+ tech companies, including Dropbox, Adobe, Coursera and Cruise Automation. Triplebyte improves the hiring process by saving you time and fast-tracking you to final interviews. At triplebyte.com/sedaily, you can start your process by taking a quiz, and after the quiz you get interviewed by Triplebyte if you pass that quiz. If you pass that interview, you make it straight to multiple onsite interviews. If you take a job, you get an additional $1,000 signing bonus from Triplebyte because you use the link triplebyte.com/sedaily.

That $1,000 is nice, but you might be making much more since those multiple onsite interviews would put you in a great position to potentially get multiple offers, and then you could figure out what your salary actually should be. Triplebyte does not look at candidate's backgrounds, like resumes and where they've worked and where they went to school. Triplebyte only cares about whether someone can code. So I'm a huge fan of that aspect of their model. This means that they work with lots of people from nontraditional and unusual backgrounds.

To get started, just go to triplebyte.com/sedaily and take a quiz to get started. There's very little risk and you might find yourself in a great position getting multiple onsite interviews from just one quiz and a Triplebyte interview. Go to triplebyte.com/sedaily to try it out.

Thank you to Triplebyte.

[INTERVIEW]

**[00:04:30] JM**: Kurt Schrader, welcome to Software Engineering Daily.

**[00:04:33] KS**: Thanks. Good to be here.

**[00:04:34] JM**: I have a recurring set of nightmares about working in a large corporation, and in one of these nightmares, I am staring at enterprise project management tooling. It's specifically a tool that we used when I worked at eBay. It's not unique to eBay. It was just a tool that we happen to – Vendor tool that we happen to use. I think that was like 6 or 7 years ago, and I know that I am not the only one that has this nightmare about project management tools and processes and Kanban boards. Why do project management tools fill me with a sense of dread?

**[00:05:14] KS**: That's a good question. For a long time, they filled me with a sense of dread as well. We started Clubhouse to address this issue, right? I think the bottom line is that the people, not all the people, but when decisions are being made in a lot of the tools out there at big companies, at eBay, they're targeting people way up the chain. They want to make sure that the head of product management for the 5,000 people is happy and they forget about the people that actually have to use this stuff every day.

We think of that as completely the wrong way. We don't want to build the manager's tool. We want to build the developer's tool and the project manager's tool. I think that's what it is. I think people just don't think somehow we've lost track of the people that actually have to use this software all day and that their life should be enjoyable and that we can give them a tool that actually is helpful and give them visibility they need into what's going on without completely being just kind of a piece of junk. That's kind of where we are starting from, and I think that's kind of where that dread stems from.

**[00:06:23] JM**: There's also a nightmare I have about an enterprise project management tool that we used at Amazon, but that one was actually built within Amazon. So, there are some companies that you have the developers in the company that say, "I'm sick of this project management tool. I'm going to off and build my own," and then they build it and the internal project management tool ends up also being dreadful. Why is it that even when engineers within a company try to solve this problem of project management, it ends up often resembling these dreaded tools that the manager uses?

**[00:07:07] KS**: It's another good question, and I've been guilty of this as well. I've been building and running engineering teams for 20 years now and probably complaining about these tools for just as long and have sendoff sort of that team to build our own thing. We know what we need.

I think there's a lot of inherent complexity in this, and unless you are spending all your time on it and trying to be very sort of explicit and directional about where to go, it's tough. It's hard thing to build on your 10% time, because when you're doing that, you're thinking about your needs or the needs of the 5 or 6 people around you. I do believe you can build a great tool in-house for that. I think you can run a 5 or 6 person, engineering or product team. I think we all know that you can run it on a card stuck on a wall or you can run it on a Trello board or something like that.

But I think the complexity and the number of interactions and the guidance that you need to the team as the team grows is where this starts to get hard, and being very thoughtful about all those interactions and how you can sort of give people just enough guiderails to do better work without sort of feeling like you're in their way or you're really punching them in the gut to try to work through these interactions between people is the really hard part. I think it takes a lot of work to sort of scale things up.

I think one thing we did was we spent almost a year building the initial version of Clubhouse, building the data structures, building sort of what we thought we would need to scale up to larger organizations from day one while still sort of respecting the user at the lowest possible level. I don't think that's where most people start at. I think they start with, "All right. Well, let's figure out sort of a minimal case we can solve here and build from there," and it's just really hard to build that up overtime. I think just like anything, somebody else picks it up. They hack on their piece to it, but unless your team is dedicated full-time to it, like thinking through all those interactions, thinking sort of outside of, "This is my needs and these are the needs of everyone else in the organization." I think that makes it really hard, and that's kind of why those things sort of get grafted together overtime, but don't really actually work for anybody beyond the original sort of builders at some point.

**[00:09:34] JM**: Before we talk in detail about what you are building, I'd like to get your sense of the archeological dig of different project management tools, because there are so many of them. I think if we look back over the course of history, we can see these sections of time that are dominated by one project management tool or another. We've got the Jira era, the Trello era,

the Asana era. Take me through your perspective. Give me a brief history of the project management space as you see it and take us to the present.

**[00:10:19] KS**: Sure, yeah. I think from my side of things, like I said, I've been building and running engineering and software teams for 20 years now. So when I started, there wasn't even Jira. There were some open source things. There was ThoughtBuzz. There was Bugzilla. Just sort of like things you need to sort of get the job done and starting to bring people together. Then there were sort of a lot of small things, open source projects, Jira, all these things fighting it out, and Jira did a lot of things right. They were a lot of better than some of these tools in a lot of ways. When the way you installed these things and dealt with them was sort of like putting a CD into a server somewhere and putting software on it. It's much easier to deal with Jira and spinning that up and running it in a data center somewhere than a lot of these other tools. Then they made the decision to make it super cheap on the low-end. So they kind of sucked all the air out of all those other tools in a lot of ways and just kind of in the de facto standard – I don't know, call it 10 or 15 years ago.

I think that's, in many ways, we spent a lot of time looking at and engineering and product and focusing on that and kind of ignoring the rest of the market in a lot of ways right now. So, almost all those people, once you get to a certain size, use Jira or some of them will use Rally or something like that, but the stuff like that is so much more expensive that it's almost sort of taking itself out of the running for a lot of organizations.

So, I think for a longtime, the de facto choice is Jira, and Trello makes inroads. GitHub issues makes inroads. These things are sort of dive-bombing in and working, again, for small teams. The UI, the UX, the experience is a lot better in many ways on these things. But just haven't thought through kind of scale. So you kind of always go back to Jira in a lot of ways if you're an engineering team.

Yeah, I think that is really most what we focus on. I mean, there's a whole another threat to this that you kind of brought up there around Asana and Trello and these more general purpose project management tools. We try to stay very focused on the engineering side of things. I don't think we're smart enough to solve that bigger problem. Maybe somebody else is. But I think to me that side of the market seems even more – It feels like new competitors are constantly

showing up there, and they bounce in, they bounce out. Some of them get big, the Asanas and Trellos of the world. Some of them stay small. It's also difficult, because all these tools, both Jira and the other set of tools are free. So it's hard to sort of – Or have a free or very cheap version. So it's hard to kind of get started and get a foothold, use that money to build a team around it and build up from there.

So, it's interesting. I think we're getting closer and closer to toolsets thinking about the users, the end users more, and really trying to make people's days more enjoyable from the very early days when you're just trying to get work done to sort of the middle ages when tools focuses a lot on sort of management. What we can show people way up the chain while sort of ignoring the users at the bottom. I feel like we're finally entering a world, Trello, Asana, us, where we're really thinking about users, what they need to get done and hopefully we'll sort of swing in that direction further.

[SPONSOR MESSAGE]

**[00:14:17] JM**: Codacy helps development teams of all sizes to automate their code quality by identifying issues through static code analysis both in the cloud and on-premise. The Codacy product notifies users about security issues, code coverage, code duplication and code complexity in every commit and poll request directly from their current workflow.

Codacy has been designed by developers to be easy to set up and use and it's completely free for small teams up to four developers, and it's also completely free for open source projects. You can find out more and try out Codacy by going to softwareengineeringdaily.com/codacy. That's C-O-D-A-C-Y.

Codacy is a tool for static code analysis and issue identification. It will help you find security issues and code duplication, all these other issues that you can find through static code analysis. Check out softwareengineeringdaily.com/codacy, C-O-D-A-C-Y.

Thanks to Codacy for being a sponsor.

[INTERVIEW CONTINUED]

**[00:15:36] JM**: We all know the importance of GitHub, especially Microsoft. GitHub is such an interesting case, because it's this product with almost unbounded opportunity. It's like kind of what we've seen with LinkedIn, where LinkedIn started as this resume thing and it's like, "Okay, I'm posting my resume here and then I'm linking up with people," and then overtime you realize, "Whoa! Okay. They've got so many opportunities. They can go in a million different directions. What is linked did this? What if LinkedIn did that?" and now they're finally capitalizing on it after 10 or 12 or 15 years and we'll slowly see the increase in dominance of LinkedIn. I think it's clear the same will happen with GitHub at this point. We know how valuable this platform is and we can see the opportunities they're going to expand into.

But at the same time, when a product gets that big, you start to realize, "Okay, they're not going to be able to tackle everything." GitHub is not going to be the project manager de jure for software engineers. I mean to some extent, it is. It's an irremovable component of a software developer's workflow, but it is not the only component. It is just one component of the development pipeline. I know you're integrated tightly with that or you have deep integrations that people can use with GitHub.

Although, a more recent player is GitLab, which they have this – I think they're kind of going after the same spaces as GitHub, although with a slightly different spin, and they're earlier. So you could see them being hungrier perhaps and maybe going after the project management space a little bit more aggressively. How do you evaluate GitLab as a project management tool?

**[00:17:38] KS**: Yeah. This is something we talk a lot about internally, and I think we sort of have a mental model, which is if you were starting a new software project or software team and you could hit a big red button and say, "Spin up all the software I need to build a new project. Let's call it 10 people." You got 10 people. You've got a project manager. You've got a designer or two. You've got maybe a QA person and 6 or 7 or 8 engineers. So, one squad or pizza team or whatever you would like to call it. What is that big red button do?

One option, I think the GitLab option, is hit this button and GitLab spins up, and GitLab gives you your chat interface and GitLab gives you obviously your source control interface and it gives you your build of the structure and it gives you your packet repository and your project

management tool, right? I think it's an interesting idea. There are a lot of pieces there that you need to do really well.

I think another option is hit this big red button and we're going to spin up a GitHub instance and we're going to spin up a Slack instance, and hopefully we're going to spin up a Clubhouse instance. We're going to spin up a Wiki instance, but there are not a lot great options there right now, but let's find something to slot in there. What are all the best of breed tools?

I think for us –

**[00:19:15] JM**: By the way, readme.io is pretty good.

**[00:19:17] KS**: Yeah. I think it is. We could have a different conversation there.

**[00:19:22] JM**: Okay. All right. Continue. Continue with your point.

**[00:19:24] KS**: Yeah. I think there is – Those are kind of the two models. One is sort of like as a team, I'd like all sort of the best of breed tools, slot and readme.io, or whatever. You could select from the 2 or 3 that are good out there. You could pull all these together. You hit your big red button. Everything's there and you can start developing.

The other option is, yeah, you get GitLab. I think there are different tradeoffs there, right? I think just by definition, the tools in GitLab are probably not going to be at the same levels of other tools. There're other tradeoffs. They might be better integrated. It's simpler to do that, right?

So, that's kind of I think structurally how we think about it. GitHub has their project management tools, but I think you'd hard pressed to scale them to a very big size. GitLab, I think to your point, is attacking this a little bit harder, and we'll see. I don't think anybody's choosing GitLab for their project management tools at least right now. They're saying, "This is a good source control tool. We can install this on our servers locally," and we're a SaaS tool. If you need a tool that can be installed in-house, then we're not the right choice for you anyway, right?

So, I think we look at it and we watch what they're doing and we understand sort of that model, but I think there is a big market and I feel like it will continue to be a big market of people who want to say, "I'm going to take this piece off-the-shelf. I'm going to take this piece off-the-shelf, because they're all better than the GitLab model or that GitLab offers." Then I think it's up to us to build really great integrations between those. So you sort of get the best of both worlds. You get things that are very well-integrated, work really well together. Your data flows through them appropriately and lands in the places you want it to land. But each tool you're using throughout the day is sort of a better option than what would get out of GitLab.

**[00:21:24] JM**: I did an interview with Sid from GitLab, the CEO, a while ago. What was fun about it was the guy, he's got this total poker face, and I was interviewing him in-person. I was like, "So you really can go after all of these things. You're really going to do this." He's like, "Yeah, breadth, not death. We're going to go after all these different things. It's going to open source. We don't expect it to all be the best, at least initially, but we want to provide you with a simple default for all of these different things, and we'll improve them overtime," and just like total poker face. I'm like, "Really? Okay. I respect that." I have massive respect, like, "Why not?" Who can tell him, "That's not going to work?" It might work. They might be able to improve overtime.

You look at Amazon, so many random services at Amazon. They just look jenky at first. Then overtime they just iterate on them. It's like they ship them out there in the public really quickly. The public gives them feedback. They're able to iterate on them quickly. I'm not sure Amazon or AWS is uniformly best of breed, but there are certainly some solutions where they're best of breed. Then they just have this horizontal expansion, this rapid horizontal expansion and they overtime expand those horizontal expanders vertically and improve on quality overtime. But I think your focus is in the product management space. So it's existential for you to make project management work.

**[00:23:07] KS**: Yeah. I mean, if GitLab gets to the point where they've built a better tool than what we're building, then that's kind of on us, right? This is all we do, and I want to build the best thing out there for teams to build software together on. If GitLab is doing 50 things or whatever, they want to do 50 things. But if they're doing a dozen things, say, and they do all of them better than everyone else out there, then I think they deserve to win. That's pretty impressive.

**[00:23:35] JM**: That's right. Yeah.

**[00:23:36] KS**: I don't think that's going to happen. If they do, then kudos. Kudos to them. But it just feels like – I think you keyed into it a little bit there with the way you described it. It's a pretty high bar, or like each of these is very difficult on their own. I know people at GitHub and sort of said, "Hey, are you going to build this? Is this something that you're working on?" Obviously, things could have changed. This was a few months ago, but they're kind of like, "There's so much you have to do here to build a good tool. We got all these other stuff we want to build. This is a whole another X-hundred people just to build a really good tool here, and you've been at this for 5 years, and this is where you are, and you have a long ways to go."

I mean, again, if they can do it, great. But I feel like by focusing in and really trying to knock it out of the park, we'll be able to stay if not one, or if not 10 steps ahead of them, then at least a number of steps ahead of them.

**[00:24:37] JM**: Yeah. I mean, GitHub is in a place where they're drawing up the 10-year roadmap, or the 20-year roadmap and they're just like – Because that's the way that they should be thinking at this point under Microsoft, and you're just like, "Look, I want to solve project management for today's engineers or for the engineers of the next five years or whatever." I'm sure you can think very long-term as well, but GitHub is in a different position resource-wise and what they need to accomplish in order to kind of move the numbers in a meaningful way for their recent acquire.

Slack has totally changed how we do work, or I guess you could talk about the Slack-like projects as well, like Microsoft Teams or Mattermost. How has Slack changed engineering and how does that plug into what you're building with Clubhouse?

**[00:25:33] KS**: So, Slack kind of came out of the gate not too long before we started to draw what eventually would become Clubhouse. In a lot of ways it was an inspiration, right? We all had IRC channels. We all had Campfire, like HipChat. Slack, day one, was not that much different from those things, but Slack came out of the door and said, "Hey, here's a tool that everyone in the organization can understand. Here's a tool that's fun, like click this button and

something fun happens. Here's a tool that looks modern and is colorful and has a UI that people like," and it just blew everyone away. Their success speaks for themselves.

From my perspective, my engineering team at the time at my last company had probably 30 or 40 people on my team. The last company I was part of. There was – I think it was like one day where I said, "Hey, let's try Slack." The management, some of my senior people kind of pushed back and said, "We already have Campfire," of whatever it was. Literally, I said, "Okay, let's just try it for an hour." Within like an hour everyone was like, "Okay, we're done. This is where we're going to move." I had never seen that before. I've done in this business for a longtime, and that's a rare sort of thing.

I mean, I think we could have a very long discussion about whether or not Slack has effectively increased communication – Or not communication, productivity overtime. I think there are a lot of questions that are starting to crop up in organizations around where information should land. Should it land in a tool like Clubhouse? Should it land in Google Docs? Should it land in Slack? Because I think Slack, a lot of information can get lost in there overtime. I think people are learning to sort of not be on it all the time, but there was a period of time where it was 24 hours a day, people are reacting to Slack. So, from my perspective, there was this –

**[00:27:44] JM**: It was fun. You have the party parent.

**[00:27:48] KS**: Yeah, "Hey, at your own emojis," which I love actually. We let people do that too.

**[00:27:53] JM**: It's like the /giffy. That was a thing for a while. /Giffy, random GIF please. We started to see this thing. It's like the same thing happened with Facebook, where you look back at like your first two or three years of Facebook usage and you're like, "What the hell was I doing?" It's like the cryptocurrency madness. It's like, "Yeah, we're all doing ICOs. Look, this is going to happen. This is a real thing." Then like the time passed and you're like, "What the hell? What was I doing? What happened to me?"

**[00:28:25] KS**: Yeah. I mean, I think there's a lot to take from that, right? Yeah, there was this big initial burst of excitement. There was a period of like, yeah, GIFs and whatever GIFs from Marvel movies and things like that. Now, it feels like people are sort of saying, "This is good for

the following 30 things, and it makes my life better in these ways, and if we put a little bit of guardrails on things, like this is how things go – Can make life better."

I think on our side, when we look at that and how that's changed work, it's sort of like, "How can we keep the best pieces of that going?" We use Slack. Every company – If I had to go through – We have 2,000 customers that use Clubhouse at this point, a little over 2,000, and I would say if not 100%, then close to it, use either Slack or Teams or Mattermost. I mean, there's clearly a need for this. There's clearly a – People don't know how to operate without it at this point.

But I think as an industry, we're getting to the point where we're starting to have this conversation around like, "What's appropriate with this? What's not appropriate?" Things that should be preserved longer term, where do they go? Back to the GitLab thing, maybe they'll have some opportunities here, because everything is so tightly threaded together, when all these conversation in Mattermost needs to flow into whatever the GitLab, Wiki is. Then that can flow through to GitLab tickets or something like that.

Figuring how that works and, for us, sort of emulating that or making that work really well is important. Yeah, I mean, Slack is just a beast and it's been super interesting for my perspective to watch them blow up in the way that they did and just be so pervasive across industry at this point.

**[00:30:25] JM**: Now, when you're talking to customers and they're talking about how they manage their projects. I'm sure kind of interview-ish sort of sessions with your customers and your power users. Do they get prescriptive with how they want people to be working with these different software tools? Because what has worked – I mean, I have a two-person organization basically, and there are some contractors and some other people that work with us. But, generally, it's like a two-person organization. What we do is we kind of have an organic development for how tools kind of find their place, like what we use Google Docs for, and maybe we'll try the trendier tools, like the Airtables or the Notions or whatever eventually. But it's kind of organic thing. We're not super prescriptive. At large organizations, do they get prescriptive about, like you said, this issue of where data lands?

**[00:31:25] KS**: It totally varies. One thing that I thought I had a handle around when I started this company. I used to do consulting work. So, I worked in banks. I worked in big companies, small companies, startups, all over the place. My mental model was sort of there are a few ways that engineering teams work, and this is where the best practices, how things are going to go.

I think probably – We've got 2,000 customers. There's probably 2,000 different ways that people do this thing. So, we see everything from – On one end of the spectrum, thou shall use Jira. Everything has to go into the Jira. This is how bugs are going to be filed through the Jira. Here's a 20-step process to file a bug.

At the other end, organizations that say, "Every team gets to choose their own tool. This is do things however you want." We'll assume sometimes it's from the selections. Sometimes it's from like just whatever people want to use and everything in between. People that can use a tool like Clubhouse next to a tool like Jira. So it's kind of all over the place.

I think the biggest problem here and something that I think is worth somebody diving deeper into is discoverability at some level, right? When you have a large, really big organization, or even a 100-person organization, understanding what's happening within the organization and also sort of where to go to find out what's happening gets more and more difficult.

We have a statement or sort of a shorthand we use internally called the Trello problem, which is Trello lets you build unlimited boards. We've seen companies with almost 800 Trello boards for a couple of hundred users. We're like, "That's –" I always have the same conversation with the CTO, the CEO, whoever is at the top and say, "How do you know what's going on?" They're like, "We have no idea what's going on. That's why we need a tool like Clubhouse."

I mean, there's good reasons at some point to I think put some guiderails on how things go or at least start to think through how your information is going to come together overtime so you can say, "What is this? Why do we build this? Who builds it?" Having to look through 700 Trello boards to figure that out is not a scalable solution.

**[00:33:46] JM**: I think this is actually the benchmark we should use for how we break up tech companies, is once you have 700 Trello boards, you are forced to be broken up into smaller

organizations, “Sorry. We’re just going to put this into the U.S. constitution. You need to be broken up.”

**[00:34:04] KS**: Just a federal rule, right? Breaks everything –

**[00:34:06] JM**: Just a federal rule. Once you hit 700 project management. So, the problem of finding tools within an organization, I had a conversation just earlier this week about that same thing. Somebody told me, “Actually, we’re using Okta now to find –” Okta, like the enterprise authentication solution, “because all the tools – We use Okta for the single sign in across our organization. Basically, Okta is the one place where we know we can find all of the tools that we’re allowed to use.” It’s like, “That’s how people are discovering what tools are available in an organization is Okta.

**[00:34:47] KS**: Yeah. I mean, we see a similar effect. I mean, there’s just so many tools. When we go into large organization, like the step one, people will spin up a Clubhouse instance. Really like it. Put it on their own credit card just to round-round some of these stuff. But if we want to go bigger than that, it’s like you need to go through the Okta or the Duo or any of these things, approval process. So you can be on the approved apps list. So when people log in, they can see you there and click and accept and it’s already approved.

I mean, I think there are two, maybe three companies I know of that are using Clubhouse to build tools that are just tools to tell you how many SaaS tools you have at your company.

**[00:35:31] JM**: Of course.

**[00:35:32] KS**: That’s just a whole – Just in the last couple of years, that’s become a whole category of company on its own, right? So, yeah. I mean, there’s just a million things out there. So, that’s tough. Cutting through the noise is definitely hard.

**[00:35:45] JM**: What I will say is great about this time is – So the last company I worked at was Amazon, and when I was on – Actually, when I left Amazon, I was still onboarding, basically, 8 months into the company. Because when you’re onboarding in one of these big companies, it’s like, “Okay. Day one, here is a link to 400 pages of Wiki stuff to read about how you get your

tools and stuff set up and how you get your IDE set up. Here's a link to the internal tool that does this and the internal tool that does that," and just like, "This thing is deprecated and don't use that thing." It's just a complete mess.

What you're describing is kind of a mess as well, but at least it's a mess where every tool has a company behind it and they're existentially incentivized, unless you go with kind of a bundled situation. I mean, the even the GitLab situation I would say is arguably better. If you buy into everything in GitLab, at least you have a company behind it supporting it rather than this environment where every company says, "Hey, we're going to build every tool for our use cases."

I feel like we are finally moving beyond that, like the territorial nature. You see this attitude in these gigantic companies where they just want to solve everything with their own internal tool, and it's like, "I think we can safely say that is not the most productive approach now."

**[00:37:18] KS**: Yeah. I think companies are realizing that the – It's an engineering team. Especially on the engineering side, people are, "I'll solve this problem. I'll just build something to solve this problem. Why would I pay you –" I'll use us as an example, because this the joke we had sort of early on. "Why would I pay you, Clubhouse, $10 a month when I could build something that is 20 times worth and only cost me 100 times as much to build?" It's like, "How can I argue with that logic?"

**[00:37:53] JM**: Yeah.

**[00:37:54] KS**: A lot of times it would sort of be like, "But it doesn't do the one thing that I needed to do." I think as we've continued to grow, we've kind of held the balance there on not implementing in the world, but a lot of those one things, like we now have a good solution to that. I think that's probably what it came down to a lot in the past though, where somebody said, "Oh, we installed this thing. It doesn't do the three things that support that way we want our team to work. So let's just build our own thing."

Then, if that works really well for that team, a few other teams pick it up and they either put their stuff on their or get frustrated in different ways. I mean, there's got to be – Come to a point

where you say, “It’s not worth it for every single person that we bring into the organization to have to learn a full set of the Amazon internal tooling set or whatever it’s called.”

When they turn out, if you leave there and you go to Microsoft, you got to learn the Microsoft internal tooling set. Because like you said, 8 months in, you’re still onboarding, because you have to learn a whole new way to do things. I don’t think every engineering team is like a special flower that needs its own completely different way of doing things. I think we can start to standardize on some of this and then really build value by building this stuff that we should be building. That’s kind of what we’re trying to do and bring to the forefront. We can build you a great tool. You can build your own thing in-house, but it’s probably not a good use of your time.

**[00:39:25] JM**: Yeah. We will get to Clubhouse. One more product. Have you used Facebook for work at all?

**[00:39:33] KS**: I have not. I’ve used Facebook, but I have not used Facebook for work. So, you’ll have to explain it to me if you want –

**[00:39:41] JM**: No. I haven’t used it. I saw somebody on a plane using it and I was like, “Oh my gosh! I want to like crane my neck.” He was across the aisle and I just saw it at the corner of my eye and I wanted to just like peek across the aisle and see how thing this works. I don’t know how it works, but I want it. I don’t even know what it does, but I want it. Because when I think about like how I want my organization to work, having an internal Facebook would be awesome. That sounds perfect. Totally, just envelop me in my project management tool. Anyway, we’ll have to wait till they actually come out with that to discuss, I guess.

**[00:40:20] KS**: I mean, I think there’s a lot of – That is an area where there’s probably a lot of interesting things that could be done. I mean, everything back to your Bluebook, or terminals used to be able to tell you everything about everyone you work with, right? I think we’ve kind of lost that.

I was hoping at some point, Slack seems like a really natural place. If they had a nice like – A lot of information about a person and there’s an org chart and people could fill in their own thing

and put in the name of their dog and all the stuff you'd want to know about a person. That seems like a supernatural place to put that in.

But, yeah. I think there's something there. It'll be interesting to explore. But I haven't seen it, and I've never heard a company – We have thousands of customers. We have thousands of people that have tried Clubhouse beyond that, and no one's ever said, "Hey, do you have Facebook for work, or we use Facebook for work." So I'm not really clear on what the penetration is at this point.

**[00:41:14] JM**: Yeah. I don't think it's very existent.

**[00:41:17] KS**: I think, another point there, right? I think people are kind of scared of that. Do people really want to spin up Facebook at work?

**[00:41:23] JM**: No. No, they don't. They don't, unfortunately. They don't. They don't. It's a problem. It's a problem. Yeah. Trust is hard won and easily lost.

What you said about Slack there is pretty interesting. This whole motto of Slack where work happens, I would like Slack to be my work identity. I think that'll be great. I mean, right now you're working – I don't know how your company operates, but I'm assuming it's like the Google identity system. I don't mind it. It's okay. But I think I kind of want my Slack. I would prefer my identity, my work identity, to be Slack. This is not a fully formed thought, but I think that would be preferable to Google. Google is too much. They're doing too much. I don't really want my – I've already got my personal, my entire – Like I've already gone – You have no choice, right? If you have to go all-in on Google as a human being – I've already gone all-in as a human being. Do I really have to rest my entire business identity on Google as well? Is that really necessary?

**[00:42:32] KS**: Yeah. I mean, part of this is there's – Yeah, there's really not a great option. I guess there's the Microsoft Suite, but we could argue about whether that's better or worse, right? But I think to the analogy I made earlier about the big red button. You hit it and you spin up GitLab. You're spinning up at Google Labs again. You spin up everything else and you're still spinning at Google Labs account. So maybe there's something there around disrupting that at some point as well. Yeah, it's just the de facto choice for everyone, I think.

[SPONSOR MESSAGE]

**[00:43:10] JM**: DigitalOcean is a simple, developer friendly cloud platform. DigitalOcean is optimized to make managing and scaling applications easy with an intuitive API, multiple storage options, integrated firewalls, load balancers and more. With predictable pricing and flexible configurations and world-class customer support, you'll get access to all the infrastructure services you need to grow. DigitalOcean is simple. If you don't need the complexity of the complex cloud providers, try out DigitalOcean with their simple interface and their great customer support, plus they've got 2,000+ tutorials to help you stay up-to-date with the latest open source software and languages and frameworks. You can get started on DigitalOcean for free at do.co/sedaily.

One thing that makes DigitalOcean special is they're really interested in long-term developer productivity, and I remember one particular example of this when I found a tutorial in DigitalOcean about how to get started on a different cloud provider. I thought that really stood for a sense of confidence, and an attention to just getting developers off the ground faster, and they've continued to do that with DigitalOcean today. All their services are easy to use and have simple interfaces.

Try it out at do.co/sedaily. That's the D-O.C-O/sedaily. You will get started for free with some free credits. Thanks to DigitalOcean for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

**[00:45:12] JM**: Clubhouse is project management software that is built for software engineers. Tell me some deliberate design decisions that you made to optimize for engineers.

**[00:45:24] KS**: Yeah. One of the first things we did I think before – I mean, we had two companies using Clubhouse, was build a GitHub integration. We want to people where they do their work, and we use GitHub. They all use GitHub. So, it did make sense to us to have to go into the tool and update – One thing that always bothered me was, "Oh, it's the end of the day. I finished this thing. Let me go in to this other tool, the project management tool, and move this

ticket card over. Click a button to say it's done. So how much of that can we take off people's plate and make life easier for their perspective?"

I think a lot of the decisions we've made are just around making – I think there's no way around. These kind of tools are overhead in some way. They're tracking the work you do. There's always going to be some effort that you have to expend that in a perfect world you wouldn't expend.

So, how do you lower that bar as much as possible and give people the information they need, but not let them sort of interact with the tool as little as possible. Come and figure out what's going on. Focus in on the things they need to do and then go work where they want to work and really like let them do the work.

I think that feeds into a lot of things, right? There's a lot of expectations you should have of your tools at this point. If someone else updates the thing you're looking at, it should update. You shouldn't have to refresh a page. If you want to load up the thing that you're working on, the ticket or the story you're working on, it should load fast. It should be lighting fast, as fast as possible. Everything else behind it, you don't really care about, right? That kind of load in the background after you get you work up on screen as fast as possible.

One change I think that's driven me not slightly is when you load Google or – Excuse me, Gmail now. You get that little M that pops up and then fills up red and then it eventually opens, right? We want to take this in the other direction.

**[00:47:39] JM**: Same thing with LinkedIn. Same thing with LinkedIn. When you go to linkedin.com, it's like loading bar. What?

**[00:47:43] KS**: What am I waiting for?

**[00:47:44] JM**: Yeah, what am I waiting for?

**[00:47:47] KS**: I think we think a lot about this as we use a term called paper cuts, which I think I got from Matt Friedman over at GitHub. These are the little things that just drive you crazy all day long, right? The more we can reduce those and make your experiences enjoyable as

possible, as fast as possible, support the tools that you use so you can look at the ticket you're working on, look at the story you're working on. Have the link to the GitHub commits that you've done against it. Have a link to the poll request, if you had a poll request. See how builds are going. All those different pieces need to be woven in there. So you can get in, find the work you need to do as fast as possible. Figure out the context and then jump to wherever you need to actually do the work.

It's an ongoing process, but I think just thinking everything we do gets look at through the lens of how is this going to affect engineers. I think in a lot of ways it's stuff we don't do. One thing that's a constant sort of drumbeat request is sort of, "Hey, can I track how much work each of my individual engineers are doing?" That's a feature we're just not going to ever implement, because, one, I don't think it has a lot of value. Also, I think it's counterproductive to engineers actually liking us. I don't want you to feel like we're tracking, counting your lines of code against someone else's lines of code or whatever other metric you can come up with that should be easily gamed.

I think there's that piece as well, and other things. We've always been API first. So, engineers, the team could come in and build whatever they want on top of it. Just stuff like that to really like – I think it's about respecting the user. When you think about that, like, "What would I want to use? How can I make this the best tool for me as an engineer?" Then we'll secondarily think about sort of the management on top of it. It just leads you to a different place and a different way of building things.

**[00:49:59] JM**: I like the speed point, what you said about the paper cuts. Because coming back to your admiration of Slack. Slack does not have too many paper cuts. You send a message and it gets to a person seemingly, instantly. I'm sure there are some little UI tricks they're doing to make it look like it's kind of arriving in its destination faster than it has, things like that. But I know, I've done a few shows about engineering at Slack and they are so hardcore about performance. What are some hardcore performance optimizations you've made so that you can get that Slack-like buttery fast responsiveness?

**[00:50:43] KS**: Yeah. Originally, we built our own frontend framework so that we could be as fast as possible now. Now, our frontend is kind of a mix of that where we need a lot of speed

and we're mixing in a bunch of React for things that don't need to be as fast or aren't as important on that front. But that was sort of day one. We had a test suite where we said okay, and we rendered 10,000 story cards on the page and how long does that take. If that started to change, we hit the brakes and sort of moved in that direction. We hit the brakes to fix that.

Some other things on our backend, we've done a lot of optimization over the years to minimize the amount of data that needs to get to the frontend so we can render as quickly as possible. I think, originally, we did that through a bunch of different endpoints. Now we have at least for some of the work we're doing, we do use GraphQL for a lot of that to sort of only ask for the data that we need. Again, we can get things to you as fast as possible.

I think it's just constantly – Again, this isn't that hard if you keep this in mind all the time. I think it's having a set of principles that you use to do the work. When we go to work every day and everyone internally understands sort of what the goal is here, and that's how you sort of build that out.

I mean, another thing we did on the backend was things started to slow down as our database started to grow. So we actually built a custom caching layer, which right now is distributed across the couple of AWS availability zones. Basically, we spend the time to take database data, and then in the caching layer, make sure that for each organization within Clubhouse, it was all localized to one machine in the cache layer. So when you come in now – Yeah, it's great. It's part of a function we use sort of esoteric database on the backend. That sort of enabled –

**[00:52:53] JM**: Datomic?

**[00:52:54] KS**: Yeah, hey!

**[00:52:57] JM**: I saw your Hacker Noon interview. I was like, "Wow! Clojure and Datomic." You guys are like serious New York hipsters. I've been trying to get the Datomic guy on – It's Rich Hickey, right? That's who created it?

**[00:53:11] KS**: Yeah, Rich and Stu from Clojure. Yup.

**[00:53:15] JM**: I'm trying to get them on the show, because this is like my little brother when he was going through his most hipster development phase, was really a fan of Datomic. I was like, "Cool! Okay, stateless of something." I don't really know what's cool about it. But that's cool that you use it. You're the first – I think you're the first person I've interviewed who actually uses Datomic.

**[00:53:35] KS**: Yeah. I originally wrote the backend here when we're getting started long ago, and now whenever those bad code, I say that was probably my code. But, yeah. It's been a journey. It really enabled a lot of things for us that I don't think we could have, it would have taken us way too long to do or we wouldn't have been able to do otherwise. Then the flipside of that is, "Hey, this isn't scaling the way we want to. So we have to hire someone who helped build Datomic and then build out our own custom aching layer," which when we got it done, is great. But where it was a period of time where it's like, "Oh, boy! I hope we get this done, because things are slowing down and that's not good."

So, yeah. I mean, it's super cool though, and one thing that I've learned from this journey is there are a lot of people out there who would like to get paid money to write Clojure. I think we've just built an awesome team there around that. Yeah.

**[00:54:37] JM**: What a competitive advantage. That's actually a great competitive advantage. You can be the Clojure shop. That's great.

**[00:54:42] KS**: Yeah. No. It's great, because there are not very many Clojure shops, and we'll see how far we can scale this. But so far it's just been an awesome experience with hiring really smart people.

**[00:54:55] JM**: That's a measured confidence, "We'll see how far we can scale. Well, till we have all the Clojure developers, and then we shut down Clubhouse and just become a Clojure consultancy."

I bet there's a lot of – There's probably a lot of trading companies in New York who you could rent out your Clojure – So there's like trading companies. They've got hipster developers that they had and they left the trading company and you can just become a Clojure consultancy and

rent out your Clojure developers to high-frequency trading companies just in case you need to pivot.

**[00:55:28] KS**: Yeah. Find a one service that's revenue. I don't want that. I want to build a great tool and get people using it.

**[00:55:33] JM**: Really good services revenue. If you're just serving Clojure developers to high-frequency trading companies. It's cool that you built your own JavaScript framework, and you're the second company actually I've talked to recently that did that. Airtable also did that, which is interesting, because it seems like speaking of standards, we've centralized around React. But now I guess React is too generalized. We need to build more specific JavaScript frameworks.

**[00:55:59] KS**: Yeah, and I think we started before React was really a thing. I mean, it was around – It wasn't a thing it is today, right? The frameworks at the time were pretty overweight and we just went back to, "Let's take JQuery. Let's take a couple of other libraries and build this out, because it has to be fast." We wanted that level of control and level of speed. Like I said, we're plugging in some React. React is great, because you can sort of incrementally add it to your frontend. Yeah, I mean it just always came down to this has to be fast. I'm a developer. I don't want to wait for things. I'm inpatient.

Part of this just comes back me sort of stomping around the office and saying, "What is this? Why doesn't this go faster?" I think I'm reasonably well-known for us buying new SaaS tools and them not working the way that I think they should or something and then immediately escalating to the CEO or something like that. I just hold myself to that same standard and I try to hold Clubhouse to that same standard. I don't know if there's any other way around it especially 5 years ago when we started. We had to put together a thing that could go as fast as we wanted to and that is really the only way to doing at the time.

**[00:57:13] JM**: Cool. Well, we got to wrap up. Very interesting engineering stack, very interesting objective. It seems like you're doing great. A lot of stuff we didn't get to. Maybe we'll do another one at some point in the future. I really wanted to ask you about ad tech. We've done a bunch of shows about advertising fraud and I've really been trying to get an

understanding of how much fraud there is. In the entire ad tech business just in denial about the amount of fraud in this business, like the amount of ads that are viewed by bots. Is it –

**[00:57:49] KS**: Take me back, Jeff, to my last company. It depends how long. I don't know. I don't know how that – It's an interesting industry, and I think there's –

**[00:57:58] JM**: That's one adjective you could use to describe it.

**[00:58:01] KS**: Maybe it is. Maybe there is a lot of fraud. I think it's also built into the price model. I think the best way to –

**[00:58:07] JM**: Spoken like a true New Yorker. Wow! Built in to the pricing. Of course! Oh, yeah! Definitely. It's an efficient market. This is the new efficient market's dogma, is that bots are factored into the price of online advertising.

**[00:58:22] KS**: I wish I had a better answer for you. But if you had to there's CPAs, again, I'm sure – We've both seen there's a slide somewhere or sort of like the ad tech online ad tech industry, and it's got those 7,000 companies on it, right? I don't know.

**[00:58:39] JM**: The Lumascape.

**[00:58:40] KS**: Yeah. Yeah, the Lumascape, right? 10% of those are probably just click firms somewhere in the country where people are clicking on it. I have no idea. I try not to think about it too much. Install your ad blockers and use Firefox as much as possible and you'll be better than 99% of the people out there.

Yeah, I don't know. It's a rough industry. My last company was an ad tech, but we're very much vertical solution for travel ads and we're within Expedia orbits, that sort of thing. So, trying to deal with too much ad fraud was never a big part of my mandate on the team I ran over there, which was great, because it sounds just like a nightmare.

Yeah, I mean, it's got to be. It's a lot. I don't even understand the entire economic model for quick fraud, but people must be making a lot of money off of it. If people are making money off

of it, they're going to find a way to do it, right? So, I'm glad I'm not there anymore. It was quite a mess.

**[00:59:40] JM**: It is hilarious. Okay. Kurt, thanks for coming on the show. Really good talking to you, and we'll do this again sometime.

**[00:59:45] KS**: Yeah, for sure. This is great. I love the show, and I'd love to come back sometime. So just let me know.

[END OF INTERVIEW]

**[00:59:53] JM**: FindCollabs is a place to find collaborators and build projects. The internet is a great place for communicating with each other. Why aren't we building projects together? Why is it so hard to find other people that will work with you on a new project reliably? Why is it so hard to find cofounders to start a small business with?

We started FindCollabs to solve these problems. I love starting new projects, software platforms, musical songs, visual art, podcasts, and when I create these projects, I want to share them with people. I want people to be able to follow my progress, because as they see that I'm dedicated to consistently delivering quality work, maybe they'll be inspired to join me, and it doesn't have to be on a financial basis. It could be just to collaborate and to make artistic progress.

You can see my profile on findcollabs.com by searching for Jeff Mayerson. You can see the different projects I've made; games, podcasts, open source software. Some of these projects are incomplete. Some of them are well-developed, and this is how innovation and invention happens. Projects start out in an incomplete state, and if you can prove to the world that you work on projects with dedication and consistency, you will get other people to join you. Whether you want to hire them or if you just want to collaborate casually.

You can see my profile and many other people's profiles, projects that they're posting on FindCollabs, and I would love to see your projects there too. You can check it out by going to

findcollabs.com, F-I-N-D C-O-L-L-A-B-S.com. You can post a project that you're working on no matter how incomplete it is.

The best projects in the world start with an inspiring vision, whether or not they have code attached. If you like to compete, the FindCollabs Open is for you. The FindCollabs Open is a $2,500 hackathon with prizes for the best React JS project, the best machine learning project, the best game. There are lots of prizes, and you can check it out by going to findcollabs.com/open and post your project.

Thanks for being a listener to Software Engineering Daily and I hope you post your cool ideas on findcollabs.com.

[END]