EPISODE 913

[INTRODUCTION]

[00:00:00] JM: A software company needs to get many things right in order to be successful. Having a useful product with solid engineering is only the beginning. ReadMe was started 5 years ago. The company solved a seemingly simple problem; documentation for software products.

If you've worked as a software engineering, you have a look at documentation. You know that there is a wide range of quality among the documentation that exists for different software products. ReadMe solved the problem of documentation as a service and it solved the problem better than any other company in the market. But after building a great business around documentation, the direction in which to take the business was unclear. How do you expand a documentation software business? Should you start building API management systems? Should you build lead generation tools? Should you try to sell products to developer evangelists?

Greg Koberger is the CEO of ReadMe and he joins the show to talk about the business, the strategy and the fundraising process for ReadMe. Greg was previously on the show very early on in Software Engineering Daily's history, and he stood out as a guest that was remarkably friendly and willing to talk about a wide range of topics.

In the previous show with Greg, he was about a year into his business. Today it has evolved and it's been five years since ReadMe was started. We catch up on how his business has changed. ReadMe had become a profitable business very early on in its life. In Silicon Valley, a quickly profitable business is the exception. So rather than structuring the finances of ReadMe with the expectation of successive financing rounds every 18 to 24 months, Greg took a slower approach. His company grew at a rate that was affordable while maintaining that profitability and without raising money from more venture capitalists.

While ReadMe maintained profitability from its core product of documentation software, Greg patiently thought about what product to build next within the company. A software company will

typically advance into adjacent markets or it will offer products that the current users of a software product are willing to pay additional money for.

Eventually, ReadMe found its second product; developer metrics. Greg and his team figured out that gathering metrics around how APIs are being consumed has synergies with the core ReadMe product. With a second product, ReadMe was now able to forecast significant growth and market expansion. The company also gained an incentive to raise money. With additional money, ReadMe would be able to deepen the developer metrics product and hire a bigger team and think much bigger than they would have been able to with a single product company and a smaller total addressable market.

ReadMe raised a series A from Excel, which is one of the most respected venture capital firms in the business. ReadMe is now entering a new period in its development. ReadMe is a fascinating case study in balancing ambition with financial discipline at a software company, and Greg is awesome. So I hope you enjoy this episode as much as I did.

[SPONSOR MESSAGE]

[00:03:43] JM: This podcast is brought to you by PagerDuty. You've probably heard of PagerDuty. Teams trust PagerDuty to help them deliver high-quality digital experiences to their customers. With PagerDuty, teams spend less time reacting to incidents and more time building software. Over 12,000 businesses rely on PagerDuty to identify issues and opportunities in real-time and bring together the right people to fix problems faster and prevent those problems from happening again.

PagerDuty helps your company's digital operations run more smoothly. PagerDuty helps you intelligently pinpoint issues like outages as well as capitalize on opportunities, empowering teams to take the right real-time action. To see how companies like GE, Vodafone, Box and American Eagle rely on pager duty to continuously improve their digital operations, visit pagerduty.com.

I'm really happy to have PagerDuty as a sponsor. I first heard about them on a podcast, probably more than five years ago. So it's quite satisfying to have them on Software

Engineering Daily as a sponsor. I've been hearing about their product for many years, and I hope you check it out at pagerduty.com.

[INTERVIEW]

[00:05:11] JM: Gregory Koberger, welcome back to Software Engineering Daily.

[00:05:14] GK: Thank you very much. It's been – I look stuff today. It was four years ago, October, that I did it for the first time.

[00:05:18] JM: Yes. So you started ReadMe five years ago and that show was four years ago, and today there's a lot of good news. You raised a large funding round. You are profitable. You've been profitable for a while. So there're lots of cause for celebration, and I definitely I want to talk about the good news. Much of the conversations have centered around the difficulties of managing your own business. So I'd like to actually start off with something a little bit gloomier. So when you think back over the last five years, what was the hardest period of time in building the business?

[00:05:58] GK: Ooh! There're not a lot of really tough times. I think though that I would say the hardest to the point where it was a weekend, it was Labor Day actually. So exactly a year ago from two days ago or whatever, I was just done. I couldn't do it anymore. I just wanted to quit. I didn't know who to quit to, because it was my company. I think bad had happened. There's nothing specific. I don't remember what kind of like caused me. I was like, "I can't do this anymore." I just hated it.

It wasn't that I hated the people. Obviously, it wasn't I hated the product. I still think it was phenomenal. It was just after doing it for a bunch of years it just kind of like caught up with me and I was just a little burnt out and I was like, "I can't do this anymore." Then I kind of – Luckily it was over a weekend. I had some time to think and everything and I was like, "You know what? I'm going to kind of flip it and instead of being like I'm done, I'm going to try to raise money and fix the problems with money."

The good thing about being profitable is that you kind of control your own destiny and all that. If you can run it and be profitable, that's awesome. We did it. But the problem is you got a lot of things out, things that you desperately need. You ask yourself and your team to sacrifice a lot. Whether it'd be like just being like, "Oh! We'll get a real designer soon," or "Oh! We'll hire an extra engineer to help out with this, as extra support person." You're asking a lot of these people to go above and beyond.

After a while, like myself included, it goes – After a while, it just sucks where you're like, "Oh! We just can't keep asking people to like kind of run on empty and go above and beyond." Our numbers were fine, but like there's just some stuff. I guess we needed a good story. We needed to kind of like get some numbers in line, things like that, like get our financials ready. So it took me a few months to actually like actually start raising money. But I started a few months after that, I decided – That's when I decided to actually raise money and gave it a few months to kind of like get ready for it and internally kind of promote some people, be ready for an influx of being able to hire after that. Then we started raising about February or so of this past year.

But I think that my lowest point. The second lowest point was actually raising money. That was a miserable experience that I'm happy to talk about. It went really well. So I actually can't complain, but it was so incredibly hard as well. So I think those were the two hardest for me.

[00:07:55] JM: What you touched on there is one of the reasons I think I enjoy talking to you. My business is also fairly – I mean it's not a business where I haven't raised money, and I think you raised a very small seed round early on. You were profitable pretty quickly. The advantage of having a business that's profitable is you have some freedom.

The disadvantage is you are not incentivized to give up that freedom. You are not incentivized to go down the route of raising money. If you don't get on the fundraising train, there are a lot of ways in which your business tempo falls out of line with the norms of Silicon Valley. What are the ways in which that's caused kind of like a disjunction? Because I think the cadence of your business is very different than the average Silicon Valley startup.

[00:08:55] GK: Yeah, I think that was part of the frustration for me a year ago when I was like really burnt out. It was that I felt like I was putting a ton of effort into the company. To a certain

extent it was starting to eat away at me that the vision for the company and what we're doing day-to-day and the cadence, like you said, they just diverged.

If I had 15, 20 years, we could eventually build what we wanted to build. But we're just not able to get to it because we're very reactive when you go profitably. There's a reason why people raise money. There are different reasons, but at the end of the day the reason you do it is because it's basically a loan and you needed the money to kind of get ahead and like leapfrog a little bit and all that and then hopefully catch back up and keep it going.

But, yeah, like you said. There were so many things we wanted to do and I just hated that I'm going to the office and I'd be like, "I want to do all things. I have to fix this bug." I wasn't actually fixing bugs personally, but like we weren't able to hit the roadmap and every month go by and we would do a ton of work, but like it didn't feel like we're doing a ton of work. It didn't feel like we're like releasing things at the cadence that we used to when we're smaller and had no customers or that we could if we had a little more money and a little more time and a little more like just resources. Not just resources as in like, for example, just being able to pay for software and things like that. But being able to hire new people, specialize people and things like that.

Yeah, I mean there's – Like you said, like there's something kind of nice about things being about to fail, because it just causes you to like kick things in a high gear. When things are just going really well, not amazingly well, but just like really well. You never have the incentive to make that jump. So we wanted to do a series A for a while. Not desperately. It was always just one of those like, "Yeah, we're going to do it someday type things." But it was desperate. We never like really need it. We never saw like, "Oh, we desperately need this." So we never like really went for it because it was always like, "Oh, we'll do it next month." For something like that huge, it takes so much time and effort to raise money. Next month became next month, which became next month and it just never happened.

Whereas the same time, we had this like idea for the product I wanted to go down. Like at first I just liked the idea. Then it started to like eat away. I mean, that we couldn't build it because it was such a phenomenal, awesome thing that I knew we wanted to build. We just couldn't get to it, because we just didn't have the ability to. We're just playing catch up all the time.

So yeah, it's tough. It's really nice problem to have that we're able to pay all our bills and we're growing steadily and we're hiring new people and all that. So it was a great problem to have. But yeah, things had changed since last time we saw, because I kind of like raised the money and hopefully put it to good use so far.

[00:11:20] JM: Okay. So the reason that you're in that nice problem to have situation is because your core product is documentation, and documentation as a service is something that there's a huge need for. There's a huge audience of people that need documentation. Almost every software company needs documentation of some kind. Some companies need externally-facing documentation. So this software has extremely higher retention. It has good revenue properties. Good reoccurring revenue properties. But the way that I saw it from the outside looking in, it's not clear what the obvious expansion is. There're a lot of things that you can sort of squint and say, "Yeah, a documentation company could expand into this market, or that market." What were the different markets that you considered expanding into or tried to expand into? How did you think about the growth when you were in that profitable stage?

[00:12:25] GK: Yeah. So documentation is not my passion even remotely, or at least paragraphs and texts are not my kind of – Not my passion. I don't get up and be like I'm really excited to give people editing tools. If that was my passion, the maybe I want to go more towards where like notion is going or something like that. There're just really cool companies out there.

What I always want to do from day one was make APIs dead simple to use, and documentation happens to be the UI or the UX that people look at. So if you like close your eyes and think about like your favorite API and kind of imagine it. I know it's kind of a nerdy thing to ask people to do. But like it was probably Stripe or Twilio, honestly. You probably imagine like the documentation. It's not just paragraphs or texts that's great about it. They have such a cohesive user experience. That's what I've always wanted to build. That was always kind of the next level that I always wanted to get to, which was documentation that's just static that you and I see the same thing is not good documentation, or at least it's not where we should or could be.

So the big thing we always wanted to do is let people send API logs to us and then start customizing the docs based on what we know about you. So if it's your first time going to the

documentation, you should see an onboarding flow. You should see some marketing pages. I mean, you should see a sign up link.

Then you go through this onboarding flow and you get next steps and all that. But if it's your hundredth time to the documentation and we know that your API is down because you're getting a ton of five hundred errors. First of all, ReadMe should be the one that alerts you of it. We should send – We'll be like, Jeff, just so you know, you're using Lyft's API and 80% of your stuff has failed over the past week. You should check that out." Then we just help you fix it and the documentation should know exactly what's going on and help you debug.

So I never wanted to be the best documentation company out there, unless you kind of look at documentation the way I do, which is like an umbrella word for anything that's like usability-related for APIs. What I wanted to do is kind of build like intercom for APIs is how we talked about it, which is a bunch of awesome tools for developers, product people, support people, sales people that's backed by a ton of data and is different for each person and like helps support people debug and lets you send targeted messages to people, like segmenting your thousand users down to a hundred users who are using version one and kind of telling them, "Hey, we've created the API."

Some companies can do all these already, like big companies have the infrastructure for this. Most companies don't have it, and we'd like documentation. I like documentation not because of documentation, but because that's just kind of the real estate when it comes with APIs. Something people touch and feel. You can add tons of new features to it without having to like build a whole new product. Start removing – That's what I loved about intercom was like I used to have a suite of 20 tools that I would use, and now I have one.

I think we can do that for APIs. We can do API monitoring. We can do emailing your API customers. We can do support forms or mailing lists, things like that, very seamlessly and no one really notices. So that's why I like documentations. Those are long rumbly answer. I hopefully kind of answered your question. But like that was always where we wanted to go. It wasn't like we kind of sat down and we're like, "Okay. We've kind of conquered this. What's next?"

This is always what we wanted to do, and that's what started to eat away at me was that it'd been four years and I was like, "I had been talking about this for years. I've been giving toxic conferences for years," and I haven't actually being able to really build what I've been talking about.

[00:15:26] JM: You made some efforts at building products around APIs throughout the years, right? Can you tell me about the efforts that you made? I think it's interesting, because the last time we spoke, you mentioned that the failures kind of led to success, or you learned more about what you should be building through building something that was not the best product.

[00:15:51] GK: Yeah. I want to start with the best. Normally, I'm pretty sane and then the problems but –

[00:15:56] JM: You know what? We're going to get to more positive information eventually. It was funny, because right before the conversation you were like, "Oh, now we're going to have a more positive conversation." I was like, "Oh! That's not how my questions begin."

[00:16:08] GK: Oh, that's okay. It's the more fun or the more interesting stuff to talk about too. Yeah, about like two years ago maybe, a little more than that, about two years ago, same thing happened. I just got – This is maybe every six or eight month cycle where I'm like really frustrated. I was frustrated, we're like, again, I gave a conference – I don't talk at conferences often, but I like them because they kind of like help me clarify my thoughts on APIs and things like that.

So I'll go to like maybe an API conference or write a blog post or something and like I'll talk about how APIs are – Like just random things I'll throw up and be like, "APIs are meant to be simple. The only goal of your API is to be used. If no one can figure out how to use your API, it's not a good API. They should be simple. They should be simple."

Then I get back to the office and it's like, "Oh, this big customer wants us to add a nested array inside a blank in this weird edge case." I'm like, "This is like not what I wanted. This is not why I'm doing this. I'm doing it to help people like build the most complex," because APIs are literally

– Sorry. Companies are just basically really easy early on and they spend all your time just working on edge cases and all that. A company just at some point –

[00:17:09] JM: You're talking about building the documentation software.

[00:17:11] GK: Oh! No. Sorry. Like when you're building a company, like when you're building a SaaS app. You start with a really clean idea and then you just spend a ton of time building out like edge cases for customers, and that's what we're doing.

[00:17:21] JM: Edge cases for the documentation product.

[00:17:22] GK: Exactly. I was just getting frustrated where I was like, "We should be spending 80% of our time making APIs easy to use, not adding more complexity." So I get really frustrated and I started what was a little side project. This was around the time when lambdas were getting really big, like serverless stuff is getting cool. I was like, "That's what APIs should be like. It should be like a function call." Some people might call that RPC. Others might call it – There are a lot of different ways to look at it. But at the end of the day I was like, "I like calling a function. I don't like calling APIs."

So we started to explore this thing, or I had this thought. It was like if we hosted the documentation and the API, we would know literally everything. So the documentation should be amazing thing in the world, because I know that you've never used the API, or you're heavy user of it, or you used to use it or you're getting five hundred errors.

Our documentation now shows like all the possible errors, because we just have to do that. Whereas as like if we knew what error you're having, we could hide all the errors away unless you're getting a 500 or 503 error and then we could show that and be like, "Here are the docs you're looking for."

[00:18:22] JM: This is basically, if I'm an internal user at a company or if I'm an external user, like if I'm a consumer of a Twilio style API, you could imagine function hosting service on ReadMe where if I'm consuming this API, I make a function call. If something goes wrong, I

could get in my terminal or debugging or whatever, I could get information fed to me from the ReadMe documentation so that my error messages tells me exactly what I need to do.

[00:18:54] GK: Yeah, like in this like product we built. The lambda was actually not important. We kind of abstracted all that away. We used lambda. But, yeah, the nice thing was like you'd get an error and there'd be a link and you click the link and it will take you to the website. In this website you could see like the actual log. You can see, "Oh, here's how to fix it." You can see comments from other people and how they fixed it.

If that still is not enough, you could like open a support ticket and include it right in there so that like the person who created the API could like help you out. We'd take care of billing so we could tell you like, "Oh, the reason this is not working is because your credit card failed. Here's how to fix it." There are so many cool things that we thought we could do if we build a documentation that wasn't like statically generated on like Jekyll or something like that if it was like you knew exactly what's going on with the API.

It's kind of like if you had a Facebook, you expect everything to kind of now what's going on. It's really bizarre if you think about to go to an API documentation and it has no clue what's going on with you. Everyone else in the internet, it was just kind of like a random idea I had. It started being really cool. We also are really liking it. The more features we build, the cooler it got. But it was like unlike ReadMe, the main product. We call this ReadMe build. No one really wanted to use it. People wanted – It just was too simplistic. It was too made up in someone's like fantasy world of like wouldn't it be great if. Like at the end of the day everyone's like, "Well, cool. It's really great if you wanted to build like a 10 second API, but like where I can use it."

There are people using it but like no one wanted to pay for it. There was no obvious way to charge for it. I went into it not because I did it. I went to it because it was like basically like kind of like a product like fantasy, like wouldn't it be amazing if type thing. So we stopped working on that and like went back to the main product. Like one of the things we really realized in doing that was like we don't have to choose one or the other. We can take all the cool stuff from that and bring it into ReadMe. So that's kind of where we've been going over the past year.

Rather than like actually completely integrating the API and telling people how to write their API and all that and the docs, we get 80% of the benefits just from having companies send us the API logs now. So we can still customize it. We can still do everything I just mentioned. But people could still host it on their own infrastructure.

So it's kind of – Since then, it's been a lot of just taking the ideas and the amazing stuff we came from that. Some of the stuff that didn't work, we've let fall away. But like some of the amazing stuff, like kind of taking that and pulling it into ReadMe itself. Something like random cool things that I really like about it is we built a SDK that hit away all the URLs and all that and like that made it really nice sort of bringing that into ReadMe. We spent some money and bough the API gem and package and like in Ruby and Python and Node and all that. So we own API everywhere. So we're bringing that back. W

We could make API simpler at the SDK level. So that's an example of like is an SDK documentation? Not even remotely, but like if you consider documentation just anything that makes the API easy to use, like an SDK can smooth over a lot of the rough parts. If I tell you to use my API and like I give you an API key, like you don't know how to use an API key, and should it be an header, or should it be a query prem? How do you send an API key in? An SDK can just make that really easy. It can take care of behind-the-scenes, like URL and coding things properly and all that junk. We've kind of taken that. So we're going to like repurpose the API package for all that kind of stuff.

We show logs. We're in the documentation. So when you go to the docs, you can actually see all your logs and like click it. If something's wrong, you can open support request with the log. Like you and I are looking at the exact same thing, if you're the support and all that. So that's kind of a nice thing. It was bad that we spent a lot of time on this thing, but like it was great to kind of like – It was really nice to be able to step to the thought experience, experiment and just like, "Okay, if we could just throw everything away and like build the best API documentation possible, what would it look like?" We did that, and now we can bring it back to the main product.

[00:22:37] JM: What I love about that story is that this is one of the reasons that I really fell in love with software and software businesses, is because you have these stories like, "Okay, you

basically maybe woke up at the middle of the night one day, one evening, and you thought, “Oh! Let’s do a lambda-based product.” That’s it.”

Then the next day you’re like, “Yeah, that still sounds a good idea.” You come in and you tell the team. The team started to like believe it. They’re like, “Yeah, that makes sense. Let’s do it. Let’s jump into it.” Overtime, you start to interact with users. They don’t want it. I mean, I’ve built things like – I’ve built too many things like this.

[00:23:20] GK: As have I.

[00:23:21] JM: I know so many people who have built things like this. You build something. You may go down a one to two-year journey of building something and then you just have to wake up one day and say to yourself, “This is –” I mean, this is why YC’s motto is build something people want, because a lot of software engineers end up building things that nobody wants. That’s not a fun experience. Maybe fun for the first year.

[00:23:45] GK: Oh, it’s great at first. Yeah.

[00:23:47] JM: But then it’s like, “Oh!” It’s like, “I’ve wasted all my money. I’ve wasted all my time and I have nothing to show for it.” But the beautiful thing is that sometimes the learnings or the artifacts from that process are exactly what you need. My understanding is this was like the product that came out of this, the developer metrics product, or the API metrics product, was crucial to kind of getting the company to that next level where you actually were able to raise money and kick-start the company to the next level that you want it to be at.

[00:24:23] GK: Yeah, definitely. We’re really lucky that we’re looking at this in hindsight, because there are some ups and downs and all of that. Yeah, exactly. I mean, I kind of think of it as –

[00:24:32] JM: Am I painting rows of your picture than it was actually was?

[00:24:35] GK: No. I don’t think it was ever that bad. We never spent that much time on it. It was just this like fun side project. It kind of let us experiment. I think starting projects doesn’t

always have to be – You start a lot of projects, and I have too. I've started so many projects. I don't think there's anything wrong in not finishing them or not having them be a huge success, because it's kind of like sports.

A lot of times with sports, you spend a lot of time practicing and all that. Not everything has to be the world series. Sometimes it's just a practice. You learn from it and get something out of it even if all you get out of it is realizing that that was a bad idea and you can kind of move on from the idea.

Yeah, I don't think – Thus far, I haven't done anything too much to screw things up and all that. 80% of the time, 90% of the time, 95% of the time, the company is very even keel and not just waking up every morning and being like, "I have this new idea." Do you know of Hiten Shah? He is from Kissmetrics.

[00:25:31] JM: He was on the show a while ago.

[00:25:31] GK: Oh, really? Okay. So he talked about this thing, maybe he talked about on the show too. I didn't listen to the episode. But they used to call them Hiten bums. Are you familiar with that? Basically, he would like – He wrote about it himself. But he's like he didn't realize that people would call it that, where he would come into the office and like have this brilliant idea that is going to change the company. At first that's great, but then like people just started rolling their eyes, because they'd be working on something and then it would divert the company completely and then they would work on that. Then like two days later you have the next brilliant idea.

He talked about it himself like how looking back he was like, "Oh, it must have been horrible." So you have to be careful not to be like that. We're not like that. We've been pretty even keel luckily. But that doesn't mean you can't once in a while as a company like try to shake things up a bit. Had we not gone down this ultimately failed route, which didn't take that much or that much money or it didn't hurt us that much, but had we not gone down that route just a little bit, I think I'd been harder to convince ourselves that like, "Okay. Fine. People didn't want the actual product, but they loved some of the ideas we're putting out there, and they responded incredibly well to those."

We already have 3,000 customers. Let's just bring those features to the existing customers. It's not like we were like off building something that wasn't anything to do. We're building both, where just at the end of the day, an API documentation platform. Yeah, it was really great to kind of like swerve and un-swerve back.

[00:26:45] JM: What have you learned about management in your five years running this company?

[00:26:49] GK: Ooh! So many things. One thing that I'm really lucky that I had before I started ReadMe was that I had worked for really great managers who I'm still very close to today, and very horrible managers. The nice thing about working with horrible managers is that it sucks at the time, but you realize that being a great manager is not easy. It takes a lot of work. I think a lot of people probably just assume that management is an easy thing, and it is not even remotely.

So the things I learned about management since I actually started the company though. I was always bad at delegating, think you're kind of in the same position where like you've been working alone – Sorry, not because you're bad at delegating. Maybe you are. I don't know.

[00:27:27] JM: I'm very bad. Very bad.

[00:27:29] GK: I think it's because like that's the nice thing about founders, like people start something, is that you do everything. You do everything. That's how you start a company, is that no one cares about you and no one's going to help you out. That's okay.

[00:27:43] JM: You start to assume that's the only way to do it.

[00:27:44] GK: Exactly. You build the website, and then you actually do the episodes yourself, and then you like figure out marketing, because you've never done marketing before. I'm kind of like projecting on you. But like I assume that I'm not far-off, where like you just spend a ton of time. You're like, "Well, we have to do sales now. I guess I'm doing sales now." You build up this thing where you just do everything. It's really tough to delegate.

I was never against delegating, and I think that's a scary thing. I think I always felt that I was the kind of person I would delegate. I'm not against delegating. I don't think I'm better than other people. I don't think I'm better at things than other people. It's just you're just in this mode of like doing stuff, and like management is tough. First of all, you can tell people to do stuff, because you have to like – It has to be a collaborative process even though you're technically their boss.

It's very asynchronous and you just kind of like, "Okay. I did nothing today." Even though you have the busiest day and you spent like 18 hours – Not 18 hours, but like 12 hours in meetings, or 10 hours in meetings, or even just four meetings. The hardest thing for me has been my metric for what a day is, a good day's work, has been very different.

I used to know that I did a lot of work that day because I had made a lot of stuff. I could come and show you what I did today. I mean, like open my laptop and be like, "Look, I designed," or "I programmed that," or "I sold that." Whereas now I don't get that immediate feedback, and hopefully my hourly, like when I put an hour in, the energy I put into something like comes back way more, because someone else is out there focusing on it and doing it. But like it doesn't feel that way sometimes, and management can get a little depressing, where you don't actually do anything sometimes.

So one thing that I force all of our people managers to do at the company is we do work from Wednesdays, and I make everyone who is a people manager like realize that that's their day to be an IC, an independent contributor. If they're a program, program. If you're a salesperson, sell. Whatever you are, do it. Because we don't have any just managers at the company who are just like career managers. Everyone's been promoted from an IC position.

I think people – I think all of my managers enjoy being managers and really like it, but I also think that I know how hard it is to like go from like writing code all day to like just being a manager, and it's a little weird. So I try to push everyone to kind of like take one or two days where like they get to do what they like to do, and they like to manage as well. Everyone's that doing it really likes that. But like if nothing else, what they're comfortable doing. Even if it's not the best use of their time, I think it is the best use of their time to reduce burnout, to reduce that kind of – The issues that come with it.

[SPONSOR MESSAGE]

[00:30:25] JM: Looking for a job is painful, and if you were in software and you have the skillset needed to get a job in technology, it can sometimes seem very strange that it takes so long to find a job that is a good fit for you.

Vettery is an online hiring marketplace to connects highly-qualified workers with top companies. Vettery keeps the quality of workers and companies on the platform high, because Vettery vets both workers and companies. Access is exclusive, and you can apply to find a job through Vttery by going to vettery.com/sedaily. That's V-E-T-T-E-R-Y.com/sedaily.

Once you're accepted to Vettery, you have access to a modern hiring process. You can set preferences for location, experience level, salary requirements and other parameters so that you only get job opportunities that appeal to you. No more of those recruiters sending you blind messages that say they are looking for a Java rock star with 35 years of experience who's willing to relocate to Antarctica. We all know that there is a better way to find a job.

So check out vettery.com/sedaily and get a $300 sign-up bonus if you accept a job through Vettery. Vettery is changing the way people get hired and the way that people hire. So check out vettery.com/sedaily and get a $300 sign-up bonus if you accept a job through Vettery. That's V-E-T-T-E-R-Y.com/sedaily. Thank you to Vettery for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:32:14] JM: Have you done something to differentiate the hiring process or the hiring culture because you are in such a different position than the startups that have raised tons and tons of money and they're on that fund – I guess you're on kind of the fundraising treadmill. But it seems like to a lesser degree, because you're still – I don't know if you're profitable.

[00:32:43] GK: Well, we're a little out of profitability at this point.

[00:32:46] JM: You don't want to be profitable at this point.

[00:32:46] GK: Yeah, it would be a bad thing. Otherwise we're just kind of collecting interests and no one likes that. Yeah, I want to get out of profitability. I want to get not too far out. I don't have much money.

Yeah, to answer your question, let's say that you start a company and people love the idea and they give you a ton of money. You have a ton of money. You have to just kick-start and go quick. So you have a bunch of money and you start doing things like you hire awesome people from awesome companies that are proving track record, because they're very expensive, but you know they're going to work and show up. That creates a bunch of like culture issues and all that as well. Sometimes it's really great. A lot of times it's really great. A lot of times it's bad. Because we grew a little more profitable and all that, we had an opportunity to kind of find people that were special in their own unique ways. I'll talk about the interview process in a second, but I think that contributed to it.

We had little times like let these people grow and then build out teams and promote people and not just bring in a bunch of like middle managers and all that. Everyone, like I said, that sort of the company started as an IC and has moved up. I couldn't think of better people to do it. Hopefully, we'll see. Ask me again a few years when I'm back on the podcast. But I'm really hoping that it helps us build an awesome culture that's more sustainable.

So we do things like for technical interviews. We don't do kind of technical jargony-like, anything on the whiteboard or any of the cliché things. For the technical stuff, we have people do one thing. It's bring their own projects to work on in front of us. It's such a great way to find people that – So we can only do it because we're relatively small. If you're Google, you compare to people.

But it's such a great way to like – There are a few great things about it. First is that you see what people actually are like in an environment they're used to. If I'm interviewing you, we have an hour of interview. You might spend the first 25 minutes just trying to understand why I did certain things, and I'm going to ask questions like, "Well, why did you do that?" You're like, "I don't know. I just thought of this for the first time. I just didn't know this existed five minutes ago." You don't care about the problem you're solving.

Your answer is just kind of – I don't know. They're not great answers, because you don't know why you're making these technical decisions, because you thought about it for 13 seconds in someone else's codebase or someone else's problem. Whereas if you bring your own thing that you've been working on for a few weeks or a few weeks or even a brand new idea that you want to kickoff, I get to see what people are like in their actual environment, which is going to be true after three months of working at ReadMe. You'll be pretty okay with the environment.

Like you get to ask them questions and like the answers are so much more thoughtful. It's like, "Well, why did you use this?" The answer, "Oh! Well, I used this because – I used Mongo because we originally use Postgres and we switched off it from Mongo because –" It doesn't really matter what the answer is, but like you actually get real answers as supposed to like, "Why did you do it that way?" They're like, "It was the first thing I thought off."

We need a technical interview sometimes. You ask them to do something specific or solve a problem or debug or whatever. So we get to that, and something that like most companies don't get to do it. Because of that we found some amazing people that maybe we wouldn't have hired otherwise, but we just got to see how creative they were, or like some people really impressed me with their ambition with like what they wanted to build, or their excitement.

To [inaudible 00:35:45] a little bit, like the first question I ask everyone who interviews is – I ask two questions, and that's pretty much our only interview, is I ask what is the coolest thing you've done or built. I get so much out of that. It doesn't matter what it is. Some people say technical things. Some people are like, "Can I say something not technical?" I'm like, "Sure." They'll be like, "Well, I just built like this table." I'd be like, "Well, how did you do it?" They're like, "Well, I went up to like Berkley and I got wood and like I brought it down, I polished it and sand it."

I don't care what it is. Obviously, we do a technical interview like I just mentioned. So like we're not just letting people in because they like building wood tables, but like I just really like people who like get passionate and excited about stuff. If someone says like to me if they're like, "Oh, well, I built this like reporting system at my last company." Then if I try to dig in to it, I'm like, "Well, why did you choose to do that?" They're like, "Well, my boss told me to." I'm like, "Well, what did you use programming language-wise?" They'll be like, "I used Node." I'll be like, "Well,

did you liked Node? Why did you decided to use it?" They'll be like, "I don't know. My boss told me to."

I'm not saying that you have to do stuff outside of – You have to program outside of work. I think that's a lot to ask of people. But like I just like people who are thoughtful and excited. If they can get passionate about a wood table they're sanding down, then they can probably get excited about like ReadMe and all that.

Yeah, we've been kind of really lucky that we've been able to hire in a different way and find like just awesome, amazing diamonds in the rough type people or just like really junior people, or they're not junior. They just don't have the resume to back it up. We've been incredibly lucky with who we've been able to hire due to that.

[00:37:12] JM: Let's come back to the developer metrics product, the API metrics product. So you built this API hosting product. Initially, the idea was we're going to allow people to spin up APIs on this service and it will be tightly integrated with the documentation. Overtime, you sort of inverted it and said, "People are already hosting their documentation. What if we just allow them to insert or decorate their APIs with this kind of functionality?" What's the insertion point if I want to add – Well, I guess, first describe what the API metrics product does. What is your second product do?

[00:37:58] GK: Yes. I can give you a quick overview. It's great. There are two different ways to look at it. If you're an API creator or an API consumer. So if you're an API creator, or if you're consuming, you're just kind of along for the ride. You can only use it if the API you're using uses it. But if you're the API creator, you can do a lot of – First of all, you send API logs to us. That's not exciting. There're a lot of companies that do API logging, or logging in general.

But the cool thing about us is that we don't just show logs. We show the actual users, and you can see statistics. You can see like, "Oh! I have 2,000 people using my API." You start segmenting it. You can see only people who have gotten 500 errors on version 1 of the API in the past six months. Things like that. Then that lets you do things like make more informed decisions, "Can I deprecate version 1?" You're not only just looking at like just raw traffic like you would with Google Analytics maybe. You're actually seeing the actual company. So maybe

you're only getting like one user still using version 1, but that one user happens to be your biggest customer, and that's good to know, or a big customer, and that's really good to know.

When I tell people about, everyone's always like, "Oh! That exists already." I'm like, "Does it though?" No one's doing this except unless you're like a big company like Stripe or Twilio and have built it internally. It's so obvious, because – That's why back to intercom. Intercom so well against Zendesk is because, with Zendesk, you just – And they fixed this a little bit. But like with Zendesk you're just getting an email and you're responding to the email and like you have no context. But like you start using Intercom and it's like this person signed up six months ago. They've been using you. They stopped using you a week ago. You can actually see like the actual user and it made the support so much like better.

You can also do things like – This is not just an Intercom thing. This is in everything. A lot of services do this. But like being able to target certain people using the API and being able to say like – Either be have it triggered and be like, "Oh! You are about to hit your limit, or your API limit," or "Oh, you're getting much 500 errors. Here's how to fix it." Or "Hey, we're deprecating the version of the API you're on in about a month." Being able to segment users and contact them is something that is really difficult to do right now. That's something on the creator side, the API creator.

Then on the consumer side, APIs are black box. You send something to the API. You get something back and you have no clue what's going on. Most API sites are statically generated, so you can't – There's no like – Maybe you'll have an Intercom register or something, but there's no great way to ask them questions or talk about it. Most people go to like Stack Overflor or someplace else. You don't know what's going on. The docs don't know what's you're API, your API usage. So it's all just generic stuff you're just reading through a bunch of edge cases you don't need to know about or care about.

But if the API logger actually ran the docs, you can actually see the errors in real time and see your own statistics and be like, "Oh! I've had an uptick." Because maybe having 404 errors 25% of the time is totally normal. Other times it spike. Just being able to like debug stuff yourself is hugely helpful. Being able to do your customize support, where the person give you support can actually look at the logs.

We have a dumb video we made off at launch, and like one of the things like this woman comes out and it's like, "My API is not working," and the person who created, it's like, "It works for me." I think that's very typical –

[00:41:04] JM: I watched that video.

[00:41:05] GK: Oh, did you? It was fun to me, because – Actually, it turns up the actor in it, a guy named Kevin – He goes by Kevin Webpage. He's actually a ReadMe user at PagerDuty and he's also an actor on the side. I was like, "That's so weird that he's like –"

[00:41:17] JM: That's cool. Those weren't just people in your office.

[00:41:19] GK: No. Those are all like real actors. Oh! Were they that bad that you thought they were our – Or do you think that my employees are so charismatic and outgoing.

[00:41:25] JM: I thought your employees were that charismatic. I was like, "I want to go work at ReadMe." They're all juggling.

[00:41:30] GK: Okay. Edit that out then. Yes, that's what the office is like every single day. There's just people juggling and everyone looks like a movie star and all of that. No. I wanted to be in our office and I wanted to use actual employees. But I kind of got pushed towards the hiring production company and all of that.

[00:41:43] JM: You hired a production company. Okay.

[00:41:45] GK: They took care of everything. I did nothing other than pay them, and I showed up and all that and I wrote the script and all of that. But, yeah, that was fun. That'd be my next thing. If I couldn't do anything in tech, I think movies would be fun.

[00:41:56] JM: By the way, I think it's getting cheaper. I think you could have gone on Thumbtack. You probably could have hired individual actors. You probably could have hired a

videographer. You probably could have saved a lot of money. I mean, I'm with you on the delegating thing.

[00:42:08] GK: Delegating. We just went over this, Jeff. I got to better about this.

[00:42:11] JM: I know.

[00:42:12] GK: Six months later I'm like Quentin Tarantino editing. People were like, "Well, we haven't gotten paid in months." I'm almost on the final cut. Yeah. I thought about that too, especially when I saw the price tag for it. It wasn't that bad.

[00:42:26] JM: It's like a minute-long commercial.

[00:42:27] GK: I know. I mean, it was like it wasn't that much. It was like $15,000 or $20,000 or less than that, and it kept creeping up for like 10k.

[00:42:34] JM: But you saw how the sausage was made and you were like, "Come on!"

[00:42:36] GK: There was a lot of stuff going on. I would not want to do it. I was very impressed by the team that did it, and they're great. So, yeah. I don't know. I'm glad I did not do it. Even though I would have wanted to do it, because it seems fun. Yeah, so that's kind of – Yeah, that's the whole point of the product.

Then, to use, that was a hard time. So we wanted originally for it to be a proxy because we're like, "This is working at the most leverage."

[00:42:54] JM: So you got to integrate this thing somehow.

[00:42:56] GK: Yeah. So I was thinking proxy or gateways as they call them for APIs. So it's basically a proxy. But I was looking at like Kong and AWS Gateway and like, "These are gigantic products that people are like really buying into." I don't want to compete with them partially because I don't want to compete with them on a business level and partially because how do you make money against Amazon? They're just bottoming the barrel of prices and really good

product, and Kong is open source and free. How can we compete with that, first of all? Secondly, I don't want to compete with them because I was looking at the products, like they're just so nuanced and there's so much going on.

[00:43:28] JM: Oh, yeah. You can't expand into an API proxy. But can you insert at the proxy?

[00:43:33] GK: Yeah. What we can do is just kind of API logs, and that gave us like 90% of what we wanted as far as features go. So there are two ways to do it. You can either use a SDK right in the code. So like there's a node library or like you're using Ruby, something like that. Kind of like you would throw in like Bugsnag or something like that. It's just like one or two lines of code and it takes care of it, or most of the API gateways have a plugging architecture. So you can like, for Cloudflare for example, you can like add the ReadMe worker to your Cloudflare thing and it will just silently run.

Ideally it would have been nice if we could just like tail server logs, but the nice thing about what ReadMe does compared to other logging stuff is we actually log thee. We know who the user is. It's not just like a random server log. We know who the user is. We know the data they've sent, data they've got back. It's all very whitelisted or blacklisted depending on how you do it. So it's not like we had sensitive data, but like at least you know the shape of the data and you know the SDK they've been used.

We didn't get enough information out of just server logs to do it that way. But we did realize that just getting the logs was all we really needed. So you can either like put it right in your codebase or you can put it in your gateway's plugin.

Approximately, it would have been better in a perfect world, but like it really wasn't much that we wanted to do that we couldn't get from logs, and logs are a lot easier to get people to deal with then a full proxy.

[00:44:48] JM: So you get logs streamed to you for the customers who are buying the developer API product and you – What are you doing on the server side to process them and –

[00:45:04] GK: Yeah. So we just throw into a big database and then, step one, we're just building like some basic, just basic logging platform. We can like just see the logs. Step is putting like metrics on top of it so you can go into a user and like see when they start using it and like their usage overtime and graphs like that. Step three, we're just adding some nice UI around it. It's really cool if you could do like a replay of the API log and stuff like that.

So we have all the information. We know it was sent. We know it was returned. Let's say that you want to debug it, like not just saying here's the log, but being like, "Here's the curl command, like run this exact API log, or API call again." Then like debug it and tweak it. Little cool features like that.

The logging stuff is not exciting to me. If we didn't – I wouldn't want do to that if we didn't have to do it. But like the stuff that you can do once you know actual user data is awesome, like actually what's going on and like it just makes it so much easier for customers to debug their own stuff.

[00:45:56] JM: What kind of stuff can you learn from aggregating the API requests? I mean, it's the users, right? The users as well as the developers who are – I mean, the developers who are writing the APIs or integrating the APIs for whatever kind of API company, the Lyft API, for example. Lyft is a customer of yours, right?

So the Lyft API, their requests. I don't know if their request come to the developer API product. But theoretically it could. You hoover in those logs. You analyze it. You can give access to the developers who are building with the Lyft API to the logs of their customers calling the APIs essentially.

[00:46:38] GK: Yup. Exactly. It's so hard to talk about, because there's like so many different like our customers, and their customers, and all that. For example, like –

[00:46:46] JM: Very subtle product, by the way. I mean, probably seems obvious to you and I at this – I mean, when you told me about it, it was like, "That's a great product." But it's probably subtle to a lot of people listening.

[00:46:56] GK: Yeah. I mean, first of all, I want to be subtle. I don't want you to use an API like ReadMe and think like, "Well, that was really flashing showy." I want you to think, "That's a really nice API."

There's this guy that I'm friends with. He used to work at Twilio as one of their first developer evangelists. His name is John Sheehan. I don't know you've had him on the podcast or anything. He built the company called – He worked at Twilio early on and then built the company called Runscope, which is an API company. But he gave a talk at Heavybit, which you do know, and you worked from there you said. It's called Hero Maker, and I really enjoyed the concept.

The concept of Hero Making for him is you want – So someone in the company takes a chance on you. They decide to use Twilio or they decide to ReadMe, and they are accountable to their peers. What you want is you want their peers to come to them and say, "Hey, awesome job with the API documentation, or awesome job with blank." Then you know deep down that like couldn't have done it without Twilio or couldn't have done it without ReadMe. But like the whole point is you want to make heroes inside of companies who kind of get like a pat in the back, because that makes them like you more and that makes –

For us, maybe it's not even peers, but like I want someone to pick ReadMe and then I want them to get feedback that like, "Oh! Your API was really easy to use." It's very subtle. What makes good API documentation? What makes bad API documentation is – Documentation is very subtle and there's not like why is Stripe's documentation is so much better than other ones? They all kind of look the same, because they've all copied Stripe's design. But like Stripe still is the most beloved. Like it's the reasons they do little things like when you make an API call and you're looking at the docs, the exact response is an actual response with your data in it. When you copy and paste an API code snippet, it has your API key in it.

I can go on about like the other little details they do. But the whole point is that none of it is supposed to be showy. If it such becomes showy, then that's bad. So it's very nuanced and subtle and like little things. It's kind of little things that you're never going to notice and you don't have to think about. Things like it just worked the first time, because it's just a struggle with like figuring out the URL and coding or figuring out, "Oh! This MIME type is wrong. It should have been like this instead of this. I posted the data as a form instead of an encoded whatever." Just

this like all these crap that APIs have that most API creators and consumers – Creators don't think much about it, because that thing is obvious, consumers like struggle with. So I want it to be very nuanced and all that.

Just knowing data means that we can kind of like skip a step here or there. Just little dumb things like we now show people a JavaScript code snippet if we know that they're using JavaScript, or I'll show them a Python code snippet and we know they're using Python. We don't make a big deal of it. We don't popup and be like, "Hey, we know you're using Python because we saw your logs." It's like, "Ha! Ha! We've clever." It's more just a like, "Oh, this is exactly what I needed. I don't have to click a link. Things like that."

So I think subtlety – You mentioned subtlety, and I think like I can talk about the grander parts of this. But like I think the subtle stuff is the stuff that really excites me about all of these and the stuff that we can do with like little subtleties that like you don't ever think about. You just internally just could be like, "That was a really nice experience. That was a really nice experience." Eventually – And so it goes.

[00:49:59] JM: Tell me how the creation of the second product kick-started the company and how you – Well, kick restarted, whatever, kick-boosted. We could talk a little bit about the fundraising process, because I think you have a lot of thoughts there. Just take me from the creation of the second product to when you felt capable of raising the series A.

[00:50:26] GK: Yeah. On like build, I didn't want to do it without having money backing it up. There's nothing more interesting going out and pitching VCs or pitching customers or pitching anyone if you're not making money on it. But I don't think the money part is important sense at like, "Oh! They look at our revenue and we're like, "That's a lot of money." But more so that it just kind of like money is a really good proxy for like, "Oh, someone actually cares." Obviously, it's more than just a proxy. But early on, if you're making $1,000 or $2,000 of something, like for example to rewind way back to when we do Y Combinator. I applied twice with the same idea. The first time we got rejected. The second time we got in. The big difference was we're making money the second time.

It's such a small amount of money compared to what we're making now. It's nothing. Right now what we're making hopefully is a small amount compared to where we're going to be in five years. But like the difference between zero and like – I think we're making like $3,000 a month or something like that. When we got into YC, like just changes the conversation. Before first time in the interview, the partners were like, "Why would anyone want this?" They said something like, "I don't think this." I'd have to say, "Well, I think."

Then it's like you both have opinions and like you have tenants to convince the Y Combinator partners that you're right. Whereas when you have customers and they're like, "I don't this," and you can say, "Oh, well. These 10 companies are willing to pay for it." That's just kind of ends the conversation a good way. The YC partners and VCs in general don't bring ego into stuff. They have theories and they have thesis. They have opinions and all that of course. But like most VCs aren't dogmatic. They're not dogmatic about like approach to stuff. They're dogmatic about like, "Oh, people really like this. I like this now." It's a good thing. There's no point in having – Because when you're too dogmatic and you're too idealistic, that's how you end up with ReadMe build. A product that I built that no one wanted.

There's something – At the end of the day you need users. You need people to pay for it or to contribute in some way. Yeah. So that was a big thing. I wanted to make sure that they actually have people using it and that people liked it and wanted, and we had that. We're making good amount of revenue off of it, which is really nice. It was almost third of our revenue I think at the time we raised and growing. So we were really smaller subset of customers, like the number of customers was really small compared to our actual product.

Yeah, it showed and it helped craft the story. We crafted the story around it. Sometimes I kind of forget what was true and what was – Not that anything was a lie, but like you kind of have to like craft a story and a narrative around it. So not that anything is ever a lie or anything, but like sometimes I forget like how many things we tried that didn't work and like we had to like move things around to different plans and all that.

When I say like disingenuous, it wasn't a lie. But it was just kind of like it seems right now in hindsight it was so obvious. Like of course people want their API documentation to be tied to the API. Of course, API documentation is better when it's personal. Like I can talk about this really –

Hopefully, I guess the listeners can decide. But like relatively coherently right now. Whereas like if you ask me a year ago, I probably like, "I don't know." I just think that like if we have like a bunch of API logs, I think we could be – For example, one thing that I thought early on was that we could pitch people, have them send their API logs to us, because make the experience better for the end user. We had a really hard time selling that to people. They just didn't care, I guess, or I guess – It wasn't the most important thing. They cared. They wanted bugs fixed. They wanted a better editor. They wanted the little things that are bothering them day-to-day to be fixed.

So that's kind of why we built this whole like analytics platform as well, so you can like understand your users, because that was something that they actually saw and use. So our buyer was starting at benefits as supposed to like just having to believe us that their users will get benefit. Does that make any sense? It's kind of nuanced with all these different people.

But rather than just putting the logs in the documentation. We also built stuff for the API creator as well. It was built on top of the logs. So that they got instant value out of it as supposed to just having to believe that –

[00:54:31] JM: Oh. Okay. So meaning there are two product owners here. So let's say I'm consuming the Twilio API. Let's say Twilio is hosted on ReadMe. Both the developer who is integrating the API and using it in their product and Twilio would want to know this information about the API theoretically, in your mind. But you said you ended up not doing that or –

[00:54:57] GK: No. We did that.

[00:54:58] JM: You did that. Okay.

[00:54:59] GK: So originally we're just focusing on documentation and we're just like, "Send us logs, and we'll show them to your end users." So we'd be like, "Hey, Twilio. Send us logs." Again, Twilio is not – This is hypothetical, but like we'd be like, "Hey, Twilio. Send us the API logs. Go through security clearance, pay more money. Send us the API logs. Then we're going to show it to your customers." They were like, "Well, I don't see anything. It doesn't make my life easier." No one have said. Twilio definitely didn't say that. No one said that.

But like where people got really excited –

[00:55:25] JM: They don't really have an incentive for that.

[00:55:27] GK: Exactly. Where the incentive they gave was like we pivoted our conversation from like our sales pitch from like the best documentation ever to like, "Oh!" We'd ask question and be like, "Well, how many people use your API?" Most people don't actually know that answer. They'd be able to like guestimate or something and they'll be like, "Oh! How do you email version one of your API and tell them that you're going to change?" They'll be like, "Oh, we don't really have a way to do that. Maybe we can email all of our customers or all that." There's no way to segment.

Like with that approach people started to be like, "Oh! I can't believe kind of these things that I take for granted on a website I need for an API." If I told you on a website like, "Okay. Can you see how many users you have?" "Yeah, I know exactly how many." "What's the average bounce? How long have they been on the website?" They're like, "Oh, the average time on the website is like 4 minutes or whatever." "Can you send an email to all your customers?" You'll be like, "Yeah, of course. I use customer.io or I use Outreach or something like that."

With APIs, you don't have it. That was kind of like the – That was the thing that we started our customer interview started to go from like, "Eh," to like, "Oh! Can I start using it today?" So yeah, that's kind of what I mean when I say it feels like very obvious and very smooth sailing to get here. But there's a lot of things where like I was really frustrated, because it's like, "I know we're on to something," and people would be like, "Eh! Not into it." It took a lot of time to like tweak the pitch and tweak the exact way. So it feels like a very smooth sailing, but it wasn't even remotely that.

[00:56:49] JM: So were you actually selling the product? You actually had customers when you started to raise the series A?

[00:56:56] GK: Oh, yeah. By the time we raised series A, we had a good numbers of customers on it. People were paying us for it. Yeah, it was going well.

[00:57:03] JM: Okay. So I think at least the narrative that I read in TechCrunch – I don't know to what extent this is actually true. But basically the idea was, "Okay. If we're going to be ingesting developer API logs, we need to have proper security built around this. Therefore we're raising a series A." I mean, that's certainly one reason to raise a series A. There are a multitude of other reasons to raise a series A. What was the reasoning for starting to pursue that when you had been in profitability for so long and then you had a new revenue stream? Why not just amp up the profitability?

[00:57:37] GK: Yeah, we still were just behind and all of a sudden we're making more money, but we also increased the service area incredibly. So when I say thing like security, I don't mean like things are insecure. But rather there's a lot of things that enterprises want in order for it to feel safe. Things like SOC 2 compliance.

You could argue whether or not back in forth, whether or not that makes something more secure or not. But like it kind of does and it definitely makes you feel like it's more secure. Things like just having so much data, like things were going really well and then they started going a little bit worse. Because like the data store we're using just couldn't handle the load and we had like start changing stuff. It's much like anything. When you build it, at first it goes really well, and then things start to break hopefully not devastatingly so, but just break in all certain types of ways. Like, "Okay, so we had Node SDK, but we needed to build an SDK for a bunch of other languages. That costs money."

Edge cases that people came up with, like bugs, or a company will come to us and use it, but like we didn't do this thing if they wanted or thought we should or whatever. Version one of the product is always the easiest version to make. So that was kind of easy and we kind of raised money. At the same time, we like launched version one so that we can kind of like have that, the money kind of help us like expand the product a little bit.

[00:58:53] JM: Okay. I want to begin to draw to a close. Do you want to talk about the series A process a little bit maybe? What was hard about raising money? What was hard about raising money in that series A process?

[00:59:05] GK: I'm happy to talk about it. I just feel like sometimes it's little bit like – Because it's in the past, it feels very much like a first-world problem, because it went really well. But the thing that was really hard for me personally was that I'm a single founder. I'm by myself. I have an awesome team that I love and they are great. But at the same time, when you run a company, your energy kind of – I hate saying this because it feels like a little egotistical to say, but like kind of seeps through the company. It was such a tough juxtaposition for me to like be in a pitch meeting and talk about ReadMe, and VCs poke at stuff. There's nothing wrong with that.

They don't want to – I mean, the good ones try to – And I said something about YC recently where they were like, "It's really easy to poke holes on something. It's really hard to like have your first gut reaction to be like, "How can this be a billion dollar company?" and then try to like, best case scenario – When you're a founder, like you and I, we don't think about problems. We think about like how can we make this – Of course, your podcasting empire and my documentation company are going to be trillion dollar companies –

[01:00:09] JM: Of course.

[01:00:10] GK: And you need is to get like a billion users listening, whatever. Like I'm being a little fictitious, but like you and I are optimists. VCs, it's easier to be a pessimist, and actually it's easier to be pessimist about like other people's projects.

It wasn't like anyone was like really criticizing or like that was going bad, but like you go through a pitch and like you have to keep [inaudible 01:00:29] in the pitch and then like they just want to talk about all the bad things and you just get through it. After it was just like, "Oh, that sucked," even if they eventually invest in you. They still want to poke holes and like all that. It was just really hard to like just go back to the office and be like, "It went well." Because if it went horrible, then like everything around you just starts to spiral and then your next pitch, everything is already spiraling, just kind of like spirals on top of each other. So it was kind of tough to like go to just pitch and go to meetings and all that and have like a lot of actual rejections, but a lot of like just like micro-rejections or like, "Oh, I don't really know," type stuff. The go back to the office and like then put in a full – Because my job didn't stop. I still had people's managers and I still had to like work on product stuff and make decisions. Make sure bills are paid and all that stuff. My job didn't stop.

I think that was the hard part was just like it's hard too, because I would never go up to you and be like, "Jeff, your baby is ugly." But like VCs have no problem doing that around a company and like, "I know a lot of company, like at the end of the day, it's just a bunch of code."

[01:01:29] JM: They're nagging you.

[01:01:30] GK: Exactly. Oh, they're definitely doing that, because they've very negotiators and they're very good at – Every VC is very good at what they do. But like I don't want people to say negative things to my company, because first of all, I probably already know them. Second of all, it just eats away at mea already. On top of that like, "I built it, and you're just insulting what I just spent the last five years in my life on." I don't think of it that way. I'm very – I didn't like go home and like cry and be like, "Oh my God! They don't like what I built." I don't think about that way. But like – Deep down, you're like, "Screw you. I spent so much time and effort into building this," and you have cried over it and all of that. So that was where it got really tough I think.

[SPONSOR MESSAGE]

[01:02:16] JM: GitLab Commit is coming to London. GitLab Commit is GitLab's community event. GitLab is changing how people think about tools and engineering best practices, and GitLab Commit is a place for people to learn about the newest practices in dev ops and how tools and processes come together to improve the software development life cycle.

GitLab Commit is the official conference for GitLab, and it's coming to London, October 9th, at the Brewery. If you can make it to London on October 9, mark your calendar for a GetLab Commit. Go to softwareengineeringdaily.com/commit and sign up with code COMMITSED to save 30% on conference passes.

If you're working in dev ops and you can make it to London, it's a great opportunity to take a day away from the office. Your company will probably pay for it. I'm going to go to a GetLab Commit at some point. It won't be this one in London unfortunately. But I will definitely go, because I'm excited about the GitLab ecosystem. It's quite an interesting ecosystem. Seeing it develop has been cool over the course Software Engineering Daily's. At GetLab Commit, there are speakers

from VMWare, Porsche and GitLab itself. So if you're in London on October 9th, you can check it out. I hope you do check it out. Thanks to GitLab for being a sponsor.

[INTERVIEW CONTINUED]

[01:03:51] JM: So I think is a good segue into maintaining especially as a solo founder. This is one reason I get a lot out of our conversation is I'm a podcaster, right? I'm not running a SaaS company. But I work a lot and I started this by myself, and it's – I mean, total first-world problems. Good problems to have. But it can be alienating. It can be isolating, and it's weird, because you're doing something that you love. You're bringing that you really believe in into the world. Takes a long time, it's fun, but it's psychologically, like it's treacherous.

You can really – It's treacherous in weird ways, because like the example of you waking up with your idea for the lambda-based product. That's a fairly innocent example compared to what I'm sure some people could tell you like, "Oh, yeah. I woke up with this terrible idea. I raised a bunch of money. Three years later we sold the company in a fire sale and I was broke." That kind of thing.

So just give me some reflections on keeping your head above water if you've made any adjustments or if you have not made any adjustments or looking to make some adjustments. I don't know. Tell me how you're staying sane.

[01:05:25] GK: Yeah. I mean run – Not always successfully, but I try to run this company with a mindset of if it all goes to zero a year from now, that I will have zero regrets, because I made a salary. The company pays me. So I made a salary. I got to work with absolutely amazing people on something I really wanted to work on. Just like any job. If you leave a job after two, three, four, five years, like hopefully it's like, "That was a great experience.

Companies are just supposed to exit really well, and obviously I want that. I want it not for the money, but because that means I actually like made a difference or did something. But I think that's kind of been helpful for not being burnt out is that I've kind of treated it as like focusing on what makes me happy today as supposed to like draining my bank account and like all these just for this like I want to build a billion dollar company someday type thing. That's not to say that

that's not my goal. My goal is to build something big, but I make decisions day-to-day that looking back I hopefully won't feel like I wasted five years of my life on it or 10 years or whatever.

As far as keeping sane, I told you before, I make my managers or I try to become managers like be ICs from time to time. I try to do that too both inside and outside of the company. I'm lucky, so lucky that I've got a good amount of – I like doing weird, fun, artsy type things. I am lucky that I have a gigantic bank account in the form of like a company that I can't just spend recklessly, but like if I can justify that it's like going to help us, then that's awesome.

So far I haven't spent any money on anything that hasn't like paid back. But like I ran a conference, because it's just – I wanted – It seems like a fun of to do and that was a lot of work. We hired a rap group at the end called Rhyme Combinator who like did a rap opera musical about starting a company at the end of the conference. I got awesome speakers, like people that I like idolize and looked up to and knew about were speaking at my conference and I got to know and stuff like that. Like similar to your podcast where you get to talk to people that five, six years ago you're just like, "I know that name," and now you're like talking to them.

I think like I've been able to do things like that that I'm like I just – It's so cool that I get to do it. It's a conference, so it's a work thing. Work paid for all of it, but like it made its money back in sales and all of that in marketing and all that. That's been really nice where I still like – I don't say I'm an IC per se, but I still get to have like fun side projects at work that I can spearhead and that helped the company and all that. Then I do stuff outside of work as well to kind of like get away from it a bit. I started an escape room a few years ago, because I was really into escape rooms.

[01:08:10] JM: You started an escape room.

[01:08:12] GK: Yeah. Oh, you haven't done it yet? Oh! We got to fix this. So I used to take my team to escape rooms every time we did offsites.

[01:08:18] JM: I did it once, and it was awesome.

[01:08:19] GK: Okay. Do you know which one you did?

[01:08:20] JM: It was in Austin. I don't remember what it was called, but it was awesome.

[01:08:24] GK: So we'd do offsites in different places like Austin, and we did one in Austin as well. We do a bunch of like escape rooms as a team bonding. It was just like an hour of things that didn't involve drinking and didn't involve just sitting there.

[01:08:36] JM: It's a wonderful bonding.

[01:08:37] GK: Yeah. Actually, you get to like do stuff and work together and hang out and it's fun. I love them. Two years ago, I had the idea. My girlfriend at the time and I were just talking about like how to build an escape room. I got in my head if I'm doing it for team bonding, wouldn't it be great if we did a startup themed escape room? I was like, "Oh! Okay." So what I can do is like just all the clues can be you have one million dollars, one hour, to launch your startup or escape.

There are five tracks. There's programming, design, IT, marketing and product, and you have to do all five in order to launch your startup. Like all the clues, like it's different tracks. So if you're doing the programming one, the clues – You don't have to program but like it's very logical and like it looks like you're on GitHub or it looks like you're on programming. For like the marketing one, like –

[01:09:23] JM: Oh, it's like on the computer?

[01:09:24] GK: Oh, no. It's a real physical thing. It's in Tenderloin. It's an art gallery that we rented out. It's still running. It's called Startup Escape.

[01:09:31] JM: You have like an employee there?

[01:09:32] GK: Yeah, I got four employees.

[01:09:34] JM: What?

[01:09:35] GK: This is like a side project. I have nothing to do with it now in the sense that like I still own it and like I still like it, but I have a GM that runs it, Luke, and he's been great.

[01:09:42] JM: Is it profitable or breakeven?

[01:09:44] GK: It was profitable for a while. It started to slow down ever since it was licensed. So it's profitable. I've easily made my money back.

[01:09:50] JM: Dude! Nice.

[01:09:51] GK: It's been great. So many startups go through it. That's been awesome. So many people I know at different companies –

[01:09:58] JM: What's it cost to run an art gallery in the Tenderloin?

[01:10:00] GK: It's surprisingly cheap, because it's in the Tenderloin. I spent $3,500 a month on the space. Called another 500 on internet, power, electricity, that fun stuff. So like $4,000 for the space. The highest cost on top of that is obviously people, pay people well. Pay people $50 an hour. So if you play a game which is like $200, half of that off the back goes to the person working there. Because we do $25 an hour with a $50 minimum.

So most of the money goes to people and the space. I was trying to actually make money off this. I do things a little bit differently. I would do a little marketing and all that. But I would spend more of my time at it. But, yeah, I just wanted to build something and I love – One thing that I hate about San Francisco is the gloominess and the – I moved here in 2008 and like there's a recession. There wasn't much money. Everyone who is here just – They loved building stuff. No one wanted to start a company because they wanted to be a billionaire, because there's no path to being a billionaire starting a company – Or a millionaire starting a company in 2008. It was just the .com boom and like the 2008 crash. There's no money.

If you were in Silicon Valley building a company in 2008, it's because you were just like really excited. I miss that. I think there's a lot of negative things, and I could spend 2 hours talking to

you about like the negative things about San Francisco. We talked about the rent right before this. There's homelessness problems. We can on about all that, but like I still love San Francisco so much and I loved this escape room, sort of escape, like kind of like a loving parity of Silicon Valley. Just because like Silicon Valley is fun and we get to do cool stuff all the time.

I want it to be like not a – There's one other escape room in San Francisco that's also start up themed. It's like the CEO went on a coke binge and is now blowing up the company. Like you to have like stop them. I'm like that's not what startups are like. I want it to feel like fun and exciting and colorful. Like I kind of think of startups. Aside from that, I also like started writing songs. I'm not a musician, but like writing like parity songs about tech. I don't know if you want to play in the podcast. Like I just got one done about, it's like the end of the world as we know it parity by R.E.M, and it's the end of web dev as we know it and just goes to like web dev for the past 20 years. Starting with like –

[01:12:16] JM: Definitely send it to me.

[01:12:16] GK: Geocities and all of that up until like now. It's like JQuery in the middle and like React and Blockchain now and just kind of goes through all that. I did like a love song, a love country ballad called 410 Gone, where it uses API status codes, like write a song and like – So all the status codes are like the numbers, like 200 okay and all of that, like 503 forbidden. I think that's what 503 is. So I would just use all those words in like in the song and like use a SaaS code number and then the words. So I wrote that.

I've just been able to like find really cool, fun, creative outlets that are not that far off from what I do. The Startup Escape, the escape room, that was my own money and that was like had nothing to do with the company. But everything else is give or take like part of the company and like. I've been able to kind of – There are lots of really – Which we started out with, like really crappy things about starting a company. It's really hard. I could go on for hours about the bad stuff. Not to complain, but just like starting a company really changes you in some negative ways.

[01:13:17] JM: Oh, for sure.

[01:13:18] GK: I just briefly talked about this before this, but like it just changes your interactions with other people. It changes like – You just have to be more confident and always on and always selling. There's a lot of bad things, but there's also so many good things where like – Not to project a new, but like you get to talk to people that are way out of like – You shouldn't be able – Not that you shouldn't be able to talk to, but like amazing people. I get to do that. I get to like, yeah, like write songs and build escape rooms and run conferences and be like on stage.

[01:13:48] JM: I'm pitching myself all the time.

[01:13:50] GK: Yeah. It's tiring. That's what I hate about back to the raising money, what I was tired by, is like, "Yeah, you don't get to just turn yourself off." You're Jeff the podcaster. You don't get to be – Well, you probably don't call yourself that. But you don't get to like be tired. You don't get to like – You're actually worse off than me in a way where like I can go home for a month and like I'm still going to make money. Whereas like you can't do that. You actually have to constantly be pitching yourself and like you're only – At least I have some compounding stuff. Not to ruin your night. But you have like –

[01:14:23] JM: Oh no. You're not telling me anything I don't know.

[01:14:25] GK: Yeah, you're only as good as your last few podcasts. Yeah, it's rough. Isn't it? We talked about this before. I don't know if you're okay with me like kind of bringing it up. But like it's lonely when you're starting out. You have friends. It's not that you don't have the ability to talk to people. You actually talk to people for a living. But like you have this lack of camaraderie and all that. Then if you hire people, then you have these people around you.

I was reading this book, An Oral History of the Daily Show by like Jon Stewart who was interviewed a lot and other people.

[01:14:52] JM: No way! That's a book? That sounds awesome.

[01:14:55] GK: So it's actually made by Comic Central and like it has like everyone in it, Stephen Colbert, John Oliver, like everyone in it. But like they didn't shy away from like the

really bad stuff as well. It's fascinating. John Stewart has always been like, now that we're talking about politics. John Stewart has always been like one of my like inspirations. He's like just a manager. Look at like the sprawling landscape that's been created by his show. How many careers he's made, and they all love him too. I've always like looked at him as a –

[01:15:25] JM: He's like a one man S&L.

[01:15:27] GK: Exactly. Not just that. He's so good to his – According to like the book at least, but like – There are actually some negatives that are in the book too. They don't gloss over those. But my point to that was that it's worth reading, by the way. I really enjoyed it. It's like a magic book almost. But there's one part in 2008 which is when – I don't know the year, but it was the writer's strike. If you remember that. All the writers in all of Hollywood went on strike and they wanted residuals. It was something around DVDs and then streaming were happening and it wasn't on their under contracts and all that.

I was like my heart was a little – I mean, again, first-world problems. But like my heart was like all broken for John Stewart, because he was like talking about like I felt we're all friends. I thought we're all in this together. Then like they started striking against me and like I realized I was the man. I realized that we weren't friends. He like paid all their salaries, or he paid some people's salaries while they're striking. He gave other people like advances on their paychecks so that they could like survive for these like six months that they're striking.

My whole point to all of that is like I haven't this yet luckily, but like as much as you get along with the people that you work with, it's still your company and you still to be the bad guy sometimes. That gets really lonely sometimes. I can go to all my employees and I can be very, very honest about everything. I don't hide anything. I don't like about anything. I don't – Nothing like that. But like the one thing that you have to be a little hold your cards close to the chest done is just how you're feeling about stuff to a certain extent. I always kind of say like I don't lie about the facts, but sometimes I might misrepresent how I'm feeling about it, because you don't want things to spiral. You have to kind of be the leader, and that gets a little exhausting and tiring after a while. You just kind of want someone to like be your safety net as supposed to the other way around.

[01:17:11] JM: It's not your college friends.

[01:17:11] GK: No, exactly.

[01:17:15] JM: I remember, that's one of the weird things about moving to the city and getting really into it and having to be on all the time. It's not like your college friends.

[01:17:24] GK: No.

[01:17:25] JM: You're not hanging out and drinking beers and speaking with extreme candor.

[01:17:32] GK: Yeah. Because like let's say that you and I are hanging out in college and I'm like, "How is your math test?" You're like, "I fucking failed it." No one cares. I'm like, "Oh, that sucks man. Your teacher is a dick." That's okay. I'm not thinking anything less of you because you just cheated in your math test. But like if you and I are hanging out right now and you're like, "How is ReadMe going?" I'm like, "We just lost this gigantic deal."

[01:17:52] JM: It's a leak.

[01:17:53] GK: Yeah, and it's almost like – If I'm like, "Things aren't going well." Just in general, like I'm worried. Then like you're kind of thinking negative about me, which is kind of weird. You have this like bizarre, and I hate this culture of like everything had to be better and better and better. But like you just kind of like you can't to a certain extent be completely honest with your friends, because you still have to be on. You still have to sell to your friends, which is dumb to a certain extent.

[01:18:16] JM: I don't know if you know this about me, but I used to play poker. I used to play poker like pretty competitively. There are these forums called the 2+2 forums. Back when I was playing a ton of poker, it was kind of like my life now where I don't have like – Most of my friends are like business friends. They're business acquaintances, like you.

In the poker days, I use the forums as like it would be like – I would like treat the people like friends. It'd be like, "Hey! I had a really bad down swing today. I lost a bunch of money. I'm

feeling crappy. How are you guys doing? Everybody doing well? What are you reading lately? How is your life going?" It was near the end of my poker career that I realized that was the stupidest thing to do, because these are not – I mean, they're your acquaintances, but they're your competitors too.

In Silicon Valley, it's a little bit different. It's a little more positive-sum. We're not really competing with each other. I don't think. Unless you're getting into the podcast game, and even then, there's not enough podcasts. I hope you're starting a podcast. I'll listen to it. I'm not starting a documentation company nor a developer analytics company. But like – So there's a little less zero-sum. We're not as competitive. But yeah, nonetheless, you can't leak as much. That's one of the things. I love this city too, but I do enjoy going home for thanksgiving these days. Much more than I used to actually.

[01:19:42] GK: Yeah, that's fair.

[01:19:43] JM: I go home for thanksgiving and I'm like, "Wow! God! I just feel like there's a weight that's been lifted off my chest and I can feel more honest."

[01:19:49] GK: To a certain extent it's kind of like back to the poker thing. I don't know much about poker, but like I feel like my experience with poker is like the ones you do well and the ones who are famous. They're constantly posting in Instagram pictures of like what they're doing with all their winnings and all that. Everyone does that. That's just the world. We can't blame Silicon Valley or poker.

Everyone's doing that with Instagram, posting the best versions of themselves. But like it's almost got to the point – Actually, I don't even think it's a Silicon Valley thing or I don't think it's being a head of a company thing. I think it's just, at this point, that's just kind of unfortunately what – Maybe it's just you and I are both the oldest we've ever been, we're going to get older. Maybe it's just what adulthood is to a certain extent.

But like it just feels like as you get older or at least as more recently, it's harder to let your guard down and just be open and honest. I wonder like what's going to happen. I wonder if maybe we'll start seeing like more anonymous accounts online and things like that, or more like a social

network is more anonymous, where like you're not just showing after your friends. You're actually being more vulnerable. That was a bit of a tangent.

It is tough that we're just being on and that just gets exhausting. I think we've talked at for like at least an hour here, and hopefully I didn't come off as being too on or too pitchy or too promotional, because like at the end of the day I love my company. I love what I'm building. I love the people I work with. I'm so stupidly lucky.

This weekend I'm hanging out with like four of them or [inaudible 01:21:05] dome, people that I work with. I just love the people I work with. I'm not like – I don't see it as a job, but yeah, it still gets exhausting once in a while.

[01:21:13] JM: All right. Well, Greg, awesome conversation.

[01:21:15] GK: Oh God! You started it with a gloomy thing. We ended up with a gloomy thing.

[01:21:18] JM: No! I think we ended honestly. Very good. Thanks for coming on the show. Yeah, great getting to know you the last couple of years.

[01:21:23] GK: Thank you for having me.

[END OF INTERVIEW]

[01:21:33] JM: Software Engineering Daily reaches 30,000 engineers every week day, and 250,000 engineers every month. If you'd like to sponsor Software Engineering Daily, send us an email, sponsor@softwareengineeringdaily.com. Reaching developers and technical audiences is not easy, and we've spent the last four years developing a trusted relationship with our audience.

We don't accept every advertiser, because we work closely with our advertisers and we make sure that the product is something that can be useful to our listeners. Developers are always looking to save time and money, and developers are happy to purchase products that fulfill this goal.

You can send us an email at sponsor@softwareengineering.com even if you're just curious about sponsorships. You can feel free to send us an email with a variety of sponsorship packages and options.

Thanks for listening.

[END]