EPISODE 928

[INTRODUCTION]

[00:00:00] JM: Render is a a cloud provider built on top of Amazon Web Services and Google Cloud. Render uses the compute abstractions provided by the major cloud providers to build a second layer cloud provider with the goal of providing a better user experience. With time, Render plans to move off of the cloud providers and on to its own data centers. Anurag Goel is the founder of Render and he returns to the show to discuss how Render works and why there's a need for a new cloud provider.

Everyone knows that the market for cloud providers is gigantic, so why are so few companies pursuing it? From Anurag's perspective, there is no good reason. AWS, Google, and Azure, and Digital Ocean, and Heroku, they've only explored a small percentage of the potential ways that a cloud provider could be built. Anurag shares his strategy for building Render and also talks through his belief around modern software engineering, including beliefs around how developers choose their tools, which is mostly based on what they read in popular websites rather than what solves their problems. And he refers to this phenomenon as "fashion-driven development". It was a great conversation with Anurag. I'm sure he will come back on in the future. This was a fun show and I hope you enjoy it.

We are hiring a Head of Growth at Software Engineering Daily. If you like Software Engineering Daily and consider yourself competent in sales, and marketing, and strategy and want to explore the burgeoning world of podcasts, send me an email: jeff@softwareengineeringdaily.com.

[SPONSOR MESSAGE]

[00:01:52] 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-

© 2019 Software Engineering Daily

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 are 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 PagerDuty to continuously improve their digital operations, visit pagerduty.com.

I'm really happy to have Pager Duty 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 pagerduty.com.

[INTERVIEW]

[00:03:19] JM: Anurag Goel, welcome back to Software Engineering Daily.

[00:03:21] AG: Thank you. Really great to be back here.

[00:03:25] JM: Many people probably heard the previous episode. You talked at length about Render, the cloud provider you're building. Explain in a little bit of detail for people who didn't hear that episode, what does Render do.

[00:03:37] AG: Render is a really easy to use cloud provider that automates everything that you would otherwise do as part of dev ops engineering. So all you need to do with Render is push your code to GitHub and Render automates everything from there. This includes installing new versions of your apps, making sure you get zero downtime deploys, creating load balancers and automatically scaling your app horizontally and vertically and making sure your SSL and custom domains are set up correctly. Then giving you a lot of other features on top, like logging, and monitoring, and alerting.

[00:04:19] JM: You're a cloud provider that is built on other cloud providers.

[00:04:24] AG: For now, yes. Because we're a startup, we don't want to buy a whole server farm at the stage. So we're using other public clouds at a moment. At a certain scale, it would make sense for us to buy our own servers, and we'll get there when we get there.

[00:04:45] JM: Interesting. Now, you don't have to handle the problems of monitoring a data center today, but there are plenty of engineering problems that come from building a layer 2 cloud provider, building a cloud provider on top of EC2 and Google Cloud. What are some of the hard engineering problems that most SaaS startups wouldn't have to deal with that you do have to deal with as a cloud provider built on big cloud providers?

[00:05:13] AG: That's a really good question, and I don't know how applicable it is to most people in your audience. But we've encountered some really interesting problems in the past and we continue to encounter them every week pretty much. As an example, one of the things that we absolutely need to do is keep our user sites up even if our database goes down. We're using Cloud SQL, which does not have full high-availability. They have these maintenance windows where they just go down to upgrade even if you create a high-availability cluster. So it's "high-availability," and there's no way to avoid that with Cloud SQL.

So we had to re-architect our whole application to make sure that it was resilient to our database going down so that if your users are visiting your website, they have no idea that Cloud SQL is performing maintenance.

[00:06:11] JM: Are there a million high-availability SQL databases these days?

[00:06:11] AG: There are, and Cloud SQL claims to be high-availability, except for maintenance periods when they bring it down.

[00:06:23] JM: You think that's the case with the other –

[00:06:26] AG: No. We know that RDS does not have that issue.

[00:06:30] JM: Are you thinking about moving to RDS?

[00:06:31] AG: No. We're thinking of managing our own PostgreS.

[00:06:35] JM: Okay. What would that constitute?

[00:06:37] AG: Oh, it's actually fairly simple. We already do it for a lot of other people. We started out on Cloud SQL back in the day, and then we didn't have databases for our users until later. That's why we're still on Cloud SQL. But now that we have experienced managing PostgreS databases at scale, we just want to transition to managing it ourselves even for our application databases.

[00:07:06] JM: Would that be some kind of circular dependency?

[00:07:08] AG: No, because it would run in an entirely separate cluster or environment and it wouldn't come into contact with any of the PostgreS instances that we run for our users. All our user infrastructure is completely isolated from Render's control plane, which is the one responsible for things like when you create a new service, or even when you create a new database, or when you look at everything in the dashboard. That's all the control plane, and that runs on a separate set of isolated nodes that have nothing to do with our user nodes.

[00:07:48] JM: So you said, Google, they take down their manage SQL service for maintenance?

[00:07:55] AG: Yes.

[00:07:56] JM: The entire service?

[00:07:57] AG: Yes.

[00:07:58] JM: Why would they need to do that? Can't they just roll it or something and like replicate your database while they're doing maintenance on your servers?

[00:08:10] AG: It does require using things like streaming write-ahead logs and making sure that you coordinate updates and switch overs that way. I guess they just didn't build it that way from the start.

[00:08:24] JM: Have you talked to anybody on that team?

[00:08:25] AG: Not in the Cloud SQL team, no, but I've seen a lot of people complaining about this online and we're definitely affected by it just like all the other people and there's an open issue.

[00:08:35] JM: When these rolling maintenance windows happen, do your customer sites literally go down?

[00:08:41] JM: They don't anymore. But when we first encountered it like a year ago, we were like, "What's going on?" So now we built everything to be resilient to that kind of failure. Part of that includes very aggressive caching and making sure that we can detect drift across the database instances. So when they do come back up or if our caches are not fresh by the time they come back up, then we automatically catch up in terms of the whole infrastructure that is surveying our user traffic.

[00:09:20] JM: Is it routing logic's that's in that database? What exactly is it that would make you go down?

[00:09:27] AG: Right now, we don't go down if the database was down, but there are all kinds of things that we stored in the database. For example, what custom domain maps to which service in the backend? Something as simple as that, and whether a website is suspended by the user or suspended by a vendor even though the website itself is alive. Yeah. We have a feature where users can say, "Hey, I don't want to incur costs for this website, because I'm not using it right now and it's just a project that I will spin up later when I'm ready to work on it again." We have a lot of people sort of doing these side projects.

As soon as they decide that they would rather not pay for the service while they're not using it, we make it really easy for them to suspend it by just clicking a button. As soon as they do that,

we have to display to their users that this website has been suspended by their owner, the website owner, and that's a database call. At least it is, but it's also cache now.

[00:10:39] JM: Your mostly on Google Cloud?

[00:10:41] AG: Mostly. We use AWS for a few things, but most of our compute is on Google Cloud right now, but that's going to change soon.

[00:10:49] JM: Why are you on Google Cloud mostly?

[00:10:51] AG: Because when we started Render, we wanted – When I started Render, I wanted to make sure that I did not have to manage Kubernetes myself, because I've done that using kops on AWS, and this was two years ago, and the landscape has changed completely since then. But GKE was the only real viable option for managing Kubernetes or for running Kubernetes without having to deal with the headache of managing your own masters.

Kubeadm did not exist back in the day, and now you can upgrade your clusters with kubeadm, which is built into Kubernetes or as an official Kubernetes product. You can just upgrade your clusters using a single command using kubeadm, but none of that existed back in the day. So we decided to use GKE, which was better than any of the other managed solutions for Kubernetes. I think it still is even though it's suboptimal in a variety of ways.

[00:11:49] JM: This is narrative around Google Cloud, and, man, this is one of the problems with I guess people trying to report on the business of software. But the narrative is like Google Cloud is in third place. Will it ever catch up? It seems like a fairly misguided narrative to me for many, many reasons. What's your perspective on the "competitive dynamics" between Google, Amazon, and Azure, and other, I guess, prominent cloud providers?

[00:12:23] AG: Sure. I think that Google obviously does less than AWS, because they have a few years of catching up to do still for the stuff that they do end up doing on the backend. So things like just basic instances, VM instances, and networking, and storage. I felt having worked with both AWS and Amazon, I felt that Google does things slightly better. I also think that their UX, even though it is kind of still in the bad category I'd say, it is not as bad as AWS.

[00:13:05] JM: It's at least looks deliberate, rather than like the Chinese food restaurant menu of AWS.

[00:13:11] AG: Yes. So when you login – I mean I hope that they are able to improve it, but when you login, you don't get bombarded by 500 services and you're able to easily pin the services that you – Something as simple as that is like so useful but it's just hard to do, and AWS doesn't seem to care about that.

Anyway, I think Google is good when it comes to the services that they do have, but then they have these issues with higher-level services like Cloud SQL where they just bring the database down for maintenance and you're paying for a high-availability instance, but it doesn't matter. Then there are also issues with things like Google Cloud registry. I think that the contain registry compared to ECR, which is Amazon's container registry. I think that the feature set we've found, that the feature set in Google's container registry is limited compared to a AWSs version permissioning is often limited in these versions. But networking and networking speeds and being able to easily manage those networks, we found that it's easier to do on Google versus AWS.

[00:14:34] JM: I was talking to somebody who builds a managed Kafka at a conference recently, and they were saying that there is – If you're building managed services, like if you're one of these companies that's building a managed service that you're delivering over Amazon or over Google, you have to be – If you're building a database that your – Let's say you're CockroachDB. I was not talking to CockroachDB, but they were saying there are these really lower level things about the cloud providers that you will only discover if you are trying to build a service that is offered on both Google and Amazon, such as they handle disk differently.

[00:15:19] AG: Oh, completely. Yeah. As an example, Google's persistent disks are much more expensive than Amazon's. So SSD disks on Google are almost 2X the cost of AWS SSD disks. Having said that, Google has this feature which is much better than AWSs, which is being able to connect the same disk to multiple VM instances in read-only mode. So you can create a disk, populate it with, let's say, a large dataset, and then you can connect that disk, mount that disk to

multiple machines in parallel who can all read the same dataset at once. So you don't have to create multiple copies of that data. You can't do that on AWS.

[00:16:07] JM: Is there anything that you can't get out of this, like all these programmable infrastructure? When you moved to physical server sometime in the distant future, is there things that you're looking forward to having the ability to modulate, or is it just about price?

[00:16:24] AG: No. It's actually much more about control also. For example, let's take GKE, and we've been working with GKE for a while now. The current default version of GKE is 1.13, and Kubernetes 1.13 was released 10 months ago. We're now on Kubernetes 1.16. So we're three versions and 10 months behind, which means we don't have access to any of the new features, and stability fixes, and bug fix, and security fixes that all the new versions have.

You also cannot, for example, run anything other than Docker containers on GKE. So if you wanted to try a new container runtime, you can't do that. Now granted very few people actually want to do that, but for someone like us who's running multitenant workloads across the cloud or across multiple clouds, we want to experiment with different container runtimes for performance, for security, and we just can't do that on any of the managed Kubernetes solutions. So we have to run our own Kubernetes, because Kubernetes natively supports these things, and GKE has chosen to not support them.

Another big limitation is being able to control the feature gates that the kubelet, which is actually an agent running on every Kubernetes VM. Being able to control the feature gates that that kubelet enables on those nodes. You can't do that with GKE. This means that you can't try out experimental features unless maybe they have liked some versions in alpha. Some Kubernetes versions that are marked alpha, and you can try them in a separate cluster, but those clusters disappear after 30 days. So you can actually test any non-stable features according to – The stable according to Google. You can test any of them in any of your permanent clusters.

[00:18:29] JM: That sounds appealing. That said, you read the story about Dropbox moving off the cloud, right?

[00:18:34] AG: Yeah. I know the CTO of Dropbox at the time who was responsible or – And I also know other people there who ended sort of making that migration happen.

[00:18:47] JM: That sounded really hard.

[00:18:48] AG: It was. It was a lot of work, but I think that Dropbox, if they had to do it all over again, they would still choose the same path. It's the same for us. We know it's a lot of work. Even, I mean, running a cloud provider on top of Google is a lot of work. But I think that it needs to be done for the user experience that we want to provide to our users.

Another example is being able to resize disks on-the-fly without having to bring an application down. The ability to do that is available in newer versions of Kubernetes. Google Cloud does not support that right now. But that is so critical to the UX that we want to offer to our users.

[00:19:35] JM: You said versioning what?

[00:19:37] AG: Being able to resize.

[00:19:39] JM: Oh, resize.

[00:19:39] AG: Yeah. So let's say you have a database that you initially allocated a hundred gigs of space for and you've grown faster than anticipated, and now you want to resize the disk for that database. What are you going to do if you want to make sure that resize happens without the database going down? You can't do that on GKE today. But if you are managing your own Kubernetes cluster, you'd be able to do that.

Some of the other issues are lack of visibility into Kubernetes masters. So GKE, one of the biggest advantages that they claim, they have – I think it is actually somewhat useful, is that they manage Kubernetes master nodes for you. This means that they don't even give you access to Kubernetes master nodes. You can't SSH into mem. You can't log into mem. You don't have any access to the logs that Kubernetes documentation says you should check if something goes wrong. All the logging that they do is through Stackdriver, but that's not comprehensive, and I personally think that Stackdriver has lot of UX improvements to make. I

would rather use the native built-in Kubernetes logging than the Stackdriver UX, which is just slow and hard to deal with.

[00:21:03] JM: It's good that you have to dog food all these different UI experiences, because it's going to teach you what the right way to build a cloud UI.

[00:21:10] AG: Oh, it's great. Yeah. I mean, this is how Render started. I had to build out applications over and over and over again and deploy them on all these cloud providers. It was painful. It was mind numbing. It was error-prone. It was repetitive, and it doesn't have to be that way. That is when I decided that, "Look, lets us do something about it," and that's how Render came about.

[SPONSOR MESSAGE]

[00:21:42] JM: LogDNA allows you to collect logs from your entire Kubernetes cluster in a minute with two kubectl commands. Whether you're running 100 or 100,000 containers, you can effortlessly aggregate, and parse, and search, and monitor your logs across all nodes and pods in a centralized log management tool.

Each log is tagged with the pod name, and a container name, and a container ID, and a namespace, and a node. LogDNA is logging that helps with your Kubernetes clusters. There are dozens of other integrations with major language libraries, and AWS, and Heroku, and Fluentd and more.

Logging on Kubernetes can be difficult, but LogDNA simplifies the logging process of Kubernetes clusters. Give it a try today with a 14-day trial. There's no commitment. There's no credit card required. You can go to softwareengineeringdaily.com/logdna to give it a shot and get a free t-shirt. That's softwareengineeringdaily.com/logdna.

Thank you to LogDNA for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:23:01] JM: You talked a lot about Kubernetes there. We had a conversation about Kubernetes and how people are trying to figure out what they should run. Should they run their own Kubernetes? Should they go to a managed cloud provider? Should they run their own Kubernetes on top of a cloud provider? Should be used Fargate? Should they use a service mesh? What are some of the misconceptions about running infrastructure that you can opine about?

[00:23:34] AG: Sure. I think the we live in an era of FDD, which is fashion-driven development, and it all has to do with what you see and hear in the press about serverless, and about Kubernetes, and about Fargate. It's very easy to fall into the trap of thinking that you need to use Kubernetes, because all the cool kids use it. But my personal opinion, and obviously this is extremely biased, because I run a cloud provider that does not require you to use Kubernetes. But my personal opinion is that you're just fine without Kubernetes for 80% of the companies out there. You can get by by using just whatever Render has to offer.

We've actually had users move over from GKE to Render, and a major presidential campaign just moved over from GKE to Render because they didn't want to deal with the complexities and the issues of GKE.

[00:24:50] JM: That's funny. People are like, "I don't want to manage Kubernetes. So I'll go to GKE." But the reality is you shouldn't be managing a managed Kubernetes.

[00:24:59] AG: You shouldn't, and I think managed Kubernetes is it's almost like Tesla calling their cars autopilot. It's not really autopilot. It's just sort of assisted driving, and I think that that's terrible marketing term when they say autopilot, because it kind of lulls people into a sense of, "Oh, I can just drive without keeping my eyes on the road," and we've seen people die because of that.

Now, obviously, managed Kubernetes is a lot less consequential in that sense. But I think that managed Kubernetes isn't truly managed. All they do, all GKE does is manage your masters, and you don't have to run or manage your masters yourself. They'll say, "Oh, we auto upgrade your nodes for you. We auto upgrade your masters for you." But this used to be a good thing and a valuable thing two years ago with Kubernetes and kubeadm.

A lot of other tooling that has evolved over the last two years, you actually don't need GKE anymore if you want to run Kubernetes. Kubeadm, for example, has the ability to run high-availability masters and it has a single command to update all the masters. This is something that is built into Kubernetes and a Kubernetes native tool. GKE, which has been claiming that they make this easier, that that advantage doesn't exist anymore.

Even with managed Kubernetes, you have to deal with bugs in managed Kubernetes, and then you have to deal with the lack of missing features from the latest releases that we just talked about. Then you have to figure out sort of how to debug problems with masters, and there are other issues. You cannot – On GKE, you are kind of forced into the stack driver logging and monitoring ecosystem. I mean, sure, you can run Datadog yourself, but then GKE isn't making that easier for you. You still have to run Datadog yourself. So you end of managing Kubernetes regardless of whether you are on a managed Kubernetes or running a Kubernetes cluster yourself.

[00:27:12] JM: A fashion-driven development thing certainly resonates with me, and I feel really victim to that early on in the podcast where I think I still fall victim to it all the time, because the marketing messages just barrage me. To some extent, I'm selling this stuff on the podcast. But also it's easy for us to take that perspective, because we – The way that software is built in Silicon Valley is fairly different.

I mean, at least the software that gets talked about the most in Silicon Valley is fairly different than the way that, frankly speaking, most software is built. Like software companies in the Midwest, or big banks, or these enterprises that have such a variety of old and new workloads.

With those kinds of companies, I can see it much more actually like being practical to manage your own Kubernetes, or to have an open shift cluster. How broadly applicable do you think what you just said is?

[00:28:14] AG: I'd say even in the valley, it is applicable and that people are realizing that managed Kubernetes is not a panacea, the serverless is not a one-size-fits-all, and there was

just an article on Hacker News yesterday which got 1,500 upvotes, and the article's title and the summary in thesis was Serverless: Slower and More Expensive.

[00:28:39] JM: I saw that.

[00:28:40] AG: Yeah. The fact that it got 1,500 uploads counts for something. I think a lot of people have faced that in their own experiments with serverless. But AWS would have you believe Lambda is a solution to everything, and they talk about it at Reinvent, and they about how it's changing software development everywhere. The reality is that it is not suitable for the majority of workloads. I think serverless had its place. I think serverless is great if you want to spend up a lot of parallels compute instances and run computations, large computations in parallel and have all the results be eventually concatenated and orchestrated by your application.

For data apps, actually it can be very helpful, because you don't have to spin up a ton of VM's. But on the other hand, for something like a simple web server, you should really not be using serverless or Kubernetes even. I mean, if you don't want to use Render, use Heroku. You won't get a bunch of features that that vendor has that Heroku is missing. But I would still recommend trying Heroku or Render before you move to GKE or run your own VM's or serverless.

[00:29:58] JM: I love that you say that, because I listened back to our episode recently. You spent so much time in that episode dunking on Heroku.

[00:30:04] AG: I mean, I'm happy to go into why I think that Heroku is no longer suitable. It was great for 2010.

[00:30:13] JM: We did plenty of that in the last episode. People can listen back to it. Trust me. I generally agree with you. I mean, speaking as somebody who has most of my workloads on Heroku. Bu you kind of took my question in different direction than I anticipated it. What I was trying to ask is like if I'm a bank in the Midwest, or not in the Midwest. I'm bank in whatever. I'm a gigantic bank. I should be thinking about Kubernetes. I should be thinking about open shift and managing my own stuff. A bank shouldn't be trying to migrate all their workloads to Render, or Heroku, right?

[00:30:47] AG: I'm not so sure about that, but obviously am biased. I think that it is possible for even banks that are very security conscious to run workloads in the cloud. Now, historically, they haven't, and there are all these sort of fear, uncertainty and doubt points that come up when they think about moving to the cloud. This is why I think the majority of enterprise workloads still running data centers. They still haven't migrated to the cloud. But that's changing, and we are seeing more and more workloads migrate to the cloud.

Now, banks have large teams that have teams of dev ops engineers, and I think that for those organizations that have dev ops expertise and huge gigantic teams have dev ops engineers already, it actually does make sense to use that dev ops expertise and figure out what the best solution is for you. Now, I don't think that it's a straightforward choice between moving to the cloud versus running everything on-prem.

I think that you have to sort of figure out what is it in your application that requires you to run either your own data centers or in the cloud. It doesn't have to be either or. It can be a hybrid cloud as we are seeing over and over again. There are multiple companies that are using hybrid clouds and they run some workloads in their data centers. Then AWS and Google and all these other cloud providers have ways to connect their data centers through a private network to their workloads in the cloud. For example, if Goldman Sachs needed access to Spanner, they're not going to run that in their data centers. They can't.

[00:32:30] JM: Right. Yeah. It'd be interesting to watch. I don't know how much of the lack of adaption is due to fear, uncertainty and doubt and how much of it is due to just slowness, or like inertia in the way things are done.

[00:32:47] AG: I would say a lot of it is because the latter as well, and it's because you shouldn't try to fix something that isn't broken. So if you have infrastructure that's already running on your data centers, you have to have a really good reason to move that to the cloud. Especially when you're a large bank, there are a lot of considerations that go into making that happen. So this is where you see giant sales teams from Google and AWS and implementation consultants and sort of entire hordes of people trying to work with these banks and these hospitals in large organizations and trying to move them to the cloud.

[00:33:33] JM: Now that we're on that topic, you haven't mentioned Azure. Azure seems like the company that's best equipped to deploy these armies of partnership consultants and integration specialists to the banks, because they have these – As I understand. I don't know much about this area of things, but they have really well-defined sales channels for Microsoft database agreements, and Microsoft Office agreements, and Microsoft operating system agreements. From what I understand, this is what makes Azure such a force to be reckoned with. It's not necessarily their technical chops, although Microsoft certainly has those and they'll catch up to AWS and Google on the technical front over time. But Microsoft has the sales channels. Is that right?

[00:34:25] AG: That's absolutely correct, and this is why Azure is the number two cloud and it's growing faster than – I believe, in the last quarterly report, their growth was very impressive and it might've been higher than Google's, but I have to look that up.

[00:34:42] JM: Do you have any context on like what are they selling to these banks and stuff? I guess it's just straightforward like VM's and stuff?

[00:34:51] AG: Yeah. I think that they – Well, it's a variety of things, because there's also this thing called Azure App Services, which is sort of like Heroku, which is sort of higher-level. It's like Elastic Beanstalk. Like AWSs Elastic Beanstalk, or Google App Engine. But I've heard from people who use it that it isn't quite suitable or it's too expensive for what it is. But I think that Azure can sell multiple things to the cloud and to these vendors. This includes everything from perhaps even bare-metal all the way to a fully- managed cloud, like Azure App Services. Now they own GitHub, and that has to count for something.

[00:35:39] JM: What are the synergies that you predict from that acquisition?

[00:35:43] AG: The first one, which I believe has kind of come to pass already is the CICD piece, which GitHub introduced recently. I'm not completely sure how much of an overlap in terms of functionality there is with GitHub CICD and Azure dev ops, but I would not be surprised if there were a lot there, and that GitHub CICD sort of became the default Azure dev ops solution. Longer-term, who knows?

I mean, you can always imagine GitHub integrating with Azure container registry, or integrating better with Azure app services. It remains to be seen. Now, it's hard, because it's going to be hard for GitHub to do this sort of unilaterally, because there are so many developers using it who will raise a big stink about it if GitHub tries to push Azure on people. They might have to launch their own rebranded versions of Azure services, which will be interesting.

[00:36:45] JM: What do you mean?

[00:36:47] AG: Azure dev ops might be rebranded as GitHub actions.

[00:36:50] JM: No!

[00:36:52] AG: I mean, I don't know if that's going to happen, but it could happen. It's in the realm of –

[00:36:58] JM: We were talking about fake products being sold on Amazon earlier today already.

[00:37:03] AG: Well, I mean it's not a fake product.

[00:37:06] JM: It's a rebranded product.

[00:37:08] AG: It's a rebranded.

[00:37:09] JM: Oh, yeah. That's what those Shenzhen vacuum cleaner salespeople told me.

[00:37:13] AG: I mean, we'll see. It's in the realm of possibility. I'm not saying it's going to happen, but I would not be surprised if it starts happening, because they bought GitHub for $7 billion, something like that.

[00:37:26] JM: I think it was 7.5. Yeah.

[00:37:26] AG: 7.5, yeah. They have to do something with that.

[00:37:30] JM: What do you think of GitLab?

[00:37:32] AG: I think GitLab is not in fashion in the valley. No one I know personally uses GitLab, but I know a lot of render users who do. I believe that GitLab did really well early on to focus on CICD, and that was really their differentiating factor. So they've become really popular with organizations outside the valley that want a one-size-fits-all solution that takes care of all of their source control CICD container registry and stuff like that, and that has helped them become the big company that they are. I think the most recent round valued them at $2 billion.

But my personal experience with GitLab has been somewhat suboptimal. I've had the UI field buggy in a few different places. It's slower in general than GitHub. But GitHub itself is also slow. If I want to click on a pull request, I have to wait for five seconds for the page to load. Imagine if GitHu, and I think I tweeted about this a couple of months ago, imagine if GitHub made every single page just twice as fast.

[00:38:57] JM: Dude, imagine if GitHub had a mobile app.

[00:39:00] AG: Yeah. But even on the desktop –

[00:39:02] JM: Well, we're talking about like journeys to Mars here.

[00:39:06] AG: Yeah. Yeah. Well, Matt Friedman, if you're listening to this, please do something about it.

[00:39:12] JM: Well, it's like what you said. They have so many constituents, and like it's such sensitive infrastructure. It's like very hard to change things.

[00:39:21] AG: Well, I don't think so. I don't think that it's particularly hard to make a page load faster, to make a pull request page load –

[00:39:30] JM: They need serverless.

[00:39:33] AG: Okay. Well, I think that –

[00:39:35] JM: Can you guys just put some Lambdas in front of your software?

[00:39:39] AG: Yeah. I also think that post-acquisition GitHub is a much better organization. Maybe not yet a much better product, but it's getting there. But I just think that having Microsoft acquire them and having Nat as the CEO and then bringing in other people like Erica, who is now the COO, who was previously a cofounder of Bitnami, which sold recently, I believe, to Cisco or VMware.

I think that having these experienced operators and having the sort of oversight of corporate Microsoft daddy is a huge benefit for GitHub users, because my understanding of GitHub pre-acquisition was, having spoken to multiple people, there was very little coordinated execution and people kind of just did whatever they wanted. It kind of didn't have –

[00:40:34] JM: Halacracy.

[00:40:34] AG: I mean, I don't know if I would call it that. For example, everyone had been complaining about these issues with GitHub for years, and it's only after Nat took over that they started addressing them. It was just the people at the top didn't really want to be there. The founding CEO left and some of the other funders left. In the end, it was only one of the founders who was left before the acquisition.

So I think there was a lot of lack of strong management there that eventually made GitHub kind of stagnate, but that is changing. I see those changes pretty much every day actually, and it's really good to see GitHub improve things again. I'm really looking forward to the next few months where I know that GetHub is going to be focusing hard on becoming innovative and fast and nimble again. We'll see. This is actually going to make GitLab's job harder. But it's great to have competition.

[00:41:41] JM: Isn't it amazing how the tropes in software industry get turn on their head, because you're like talking with glee about how GitHub got acquired by Microsoft and now it can finally move faster?

[00:41:54] AG: I know. Yes. Yeah, I think that was a very special case, where pre-acquisition, the management piece was kind of missing or not as good as it could have been. That doesn't always happen. I mean, most companies that get acquired – I think GitHub was succeeding despite itself. But most companies that are required for that high a price already had great management in place.

So Microsoft didn't acquire GitHub management. They acquired GetHub's users and developers and then they brought in someone who is already at Microsoft but who also was acquired into Microsoft through Xamarin. Nat is not a lifelong Microsoft person. He was the cofounder and CEO of Zamarin, which is a really great .NET-based platform for mobile apps. It's open source. Microsoft acquired them, I believe, two or three years ago, and Nat had been in Microsoft since then.

But he understands developers, and I think there are other people like GitHub who are also now sort of very focused on developers and making sure that GitHub addresses all of these complaints that have piled up over the years.

[00:43:04] JM: Speaking of tropes, one thing I'm trying to sort out – This is pretty unrelated to what you do, but the mobile app release process, the app store gates. Do you think the app stores, the layer of manual human review, does that add security to the mobile app ecosystem or does it subtract from it because it slows down the release process?

[00:43:32] AG: That's a really good question. I think that it does act as an effective deterrent to people who might try to sneak in malicious applications into at least the Apple App Store. We all know that Google's review process is a lot less stringent.

[00:43:49] JM: There is a lot a lot more dangerware on the Android store.

[00:43:53] AG: Exactly. There you go. I think that the review process helps in certain ways. I think Apple has improved things over time. For example, with React Native, Apple has changed their terms of service that makes things like React Native possible that make the ability to release new features at a higher cadence possible. So it can be frustrating if you're just trying to get your app released ASAP. But that's the space you play in. At least, that's a level playing field for everyone, except Apple, where everyone's apps get reviewed sort of at the same cadence.

I think that in the end, it ends up being good for the consumer that you can't just have a malicious app on your phone as easily as you would if you were to side load them more if you could just install anything from the web on your phone.

[00:44:47] JM: Yeah, I don't know. It's so strange, because we don't really have this problem on the web anymore.

[00:44:53] AG: But that's also because the browser is a sandboxed environment, and apps have much more permissions than the browser does. Apps actually have to ask for your permission to track your location, to use the camera, to use the microphone. It's the same thing the browsers now. But if something malicious, if a website that's malicious, if you visit it, unless you download something, you're not actually going to infect your machine. Browser makers are making sure that.

So the sandboxing in the browser helps significantly. If we didn't have any of that, if you could just – I mean, going back to the days of people downloading random exe's off of the Internet, which probably still happens, there so many desktop machines that get infected because someone downloaded something that they shouldn't have.

[SPONSOR MESSAGE]

[00:45:55] JM: If your product has dashboards and reports, you know the importance of making those analytic products beautiful. Logi Analytics gives you embedded analytics and rich visualizations. You don't need to be a designer to get great analytics in your product.

According to the Gartner Analyst Firm, the look and feel of embedded analytics has a direct impact on how end-users perceive your application. Go to logianalytics.com/sedaily to access 17 easy changes that will transform your dashboards. That's logianalytics.com/sedaily.

Logi Analytics is a leading development platform for embedded dashboards and reports, and Logi gives you complete control to create your own analytics experience. Logi Analytics has been a sponsor of Software Engineering Daily for a while, and we're very happy to have them. So thanks to Logi Analytics, and go to logianalytics.com/sedaily to find 17 easy changes that will transform your dashboards. You can get better dashboards and reports inside your product with embedded analytics from Logi Analytics.

[INTERVIEW CONTINUED]

[00:47:26] JM: Do you think these are two paradigms that are going to exist for a long time? Because we have this paradigm of the mobile app and then browser app. People have been talking about progressive web apps or some kind of unification of the two. I talked to somebody. I talked to somebody the gVisor team that was talking a lot about like his beliefs around – You're smiling. So you must have some opinions on gVisor. Yeah. I don't know. What do you think? Is it going away or do you think this is like a durable paradigm? Should we start thinking about what can you do in the browser and what can you do in the app?

[00:47:57] AG: The only thing I can say on that is it's anybody's guess. It's hard to predict these things five years from now. Who knows? No one could have predicted the way the app store sort of completely took over the world before it was launched. Even Steve Jobs didn't want an app store apparently. Now, that's most of the usage in most people's daily lives.

Will it change? Yes. How it changes? Really hard to say, because you also have a web assembly now and you also have decentralized compute on the blockchain. So what happens there?

[00:48:37] JM: Man! Fashion-driven development.

[00:48:39] AG: Fashion-driven development. Well, I think that there are benefits to WebAssembly for sure.

[00:48:44] JM: Oh, yeah. That's for sure.

[00:48:45] AG: The same thing applies to compute on the blockchain. Although I haven't seen an example of a great use case for compute on the blockchain. I think it's the thing that people are trying to do. It's a bit of a two-sided market-based problem. It's like building desktop software before you had Windows.

[00:48:45] JM: That's one way of looking at it. What I heard, a friend of mine said this to me. He's like, “Why are we even thinking about compute? We don't even have the money thing solved yet. People aren’t even using this for money.”

[00:49:20] AG: Yeah, I agree. But having said that, I almost think that those two things are orthogonal, because sure you can keep worrying about the money part, but that doesn't mean that you should not or cannot use the same technology for something else. So there are people who are trying to do this without dealing with the money part, which I think is pretty interesting. But that's all it is right now.

[00:49:44] JM: Do you follow the WebAssembly stuff closely?

[00:49:47] AG: Not super closely, but we have Render users who are compiling Rust code for their static websites, and they're basically running Rust in the browser.

[00:49:58] JM: That is so fashionable.

[00:50:00] AG: Yeah. Well, it's actually super helpful for them, because one of them is using – I think this is a really cool site on Render called emojicamera.com, and they basically take your web cam photo and make that whole photo composed of emojis. It’s like pixilated. Yeah, it’s fun.

[00:50:23] JM: That’s on Render?

[00:50:24] AG: It's on Render, yes.

[00:50:25] JM: All right! Brought to you by Render.

[00:50:27] AG: Brought to you by Render. For sure.

[00:50:27] JM: I'm glad mission-critical software is running on your cloud provider. I'm sure the banks are right around the corner.

[00:50:35] AG: Hey, I told you, a major presidential campaign is also using Render for everything, and it will become public in a few days, because they have to disclose their spending anyway in a public report. So Render is going to show up in one of those spending reports.

[00:50:50] JM: Do you add a presidential candidate to your homepage users or is that like too politicized? People might be like, "I'm not going to use this." It's like, "Don't let a Democrat use our host."

[00:51:02] AG: You're resuming it's a Democrat.

[00:51:04] JM: Oh, that's true.

[00:51:05] AG: Well, I'm not at liberty to say who it is obviously. But to your question, do we add them on the homepage? We struggle with that, for sure, because we think of ourselves as a utility, a neutral platform. Think of Render in many ways as PG&E. If someone uses –

[00:51:25] JM: Dude! You have to work on your marketing.

[00:51:27] AG: I know. We're not a utility-utility. We're not a government subsidized –

[00:51:30] JM: Neither is PG&E. They're a terrible utility.

[00:51:33] AG: Well, sure. But in terms of the things that we can allow and not allow, purely in terms of that, I think they were much more like PG&E than, say, or a Verizon, or some of these telecom operators, or Comcast. Basically, things that are used by all kinds of people, and some of those people are committing crimes using the electricity, and the Internet, and the wireless service that they have.

But at the end of the day, Render is a neutral platform, and we don't – Despite our political, or moral, or ethical beliefs, it is our duty to our users to make sure that we support them as long as they're not doing anything illegal on Render. If we find out that they are doing something illegal, then we're well within our rights to kick them off the platform. But if tomorrow, another presidential campaign from a different party, perhaps an incumbent, decided that they wanted to use Render.

Despite my own personal feelings about it, I think it would be wrong for us to turn them away both as a business decision, but also as a practical decision, because where do you draw the line once you start drawing lines?

[00:52:55] JM: What if The Daily Stormer comes on Render?

[00:52:58] AG: We'll have to evaluate – We'll have to evaluate if they're doing anything illegal. I mean, if they're not doing anything illegal.

[00:53:11] JM: For context, I mean, Cloudflare kicked Daily Stormer off, and they weren't doing anything illegal, right?

[00:53:15] AG: Yeah, and that's a decision that Cloudflare took. I believe that every business has to make their choice. I think that hate speech is one of those things that we already have in our terms of service, where – Yeah. In addition to the legality, hate speech is one of the things that we do not allow in our platform. There are other things. We don't love crypto mining on Render. So it's not just the legal part of it, but there are well-defined guidelines.

[00:53:44] JM: I wouldn't say hate speech is a well-defined term, but we'll save that for another podcast.

[00:53:49] AG: Yeah, that's different. That's more of a political thing. That is why I said for The Daily Stormer, I think we'd have to evaluate whether the crossed the hate speech line. If they do, then we won't be able to have them on Render.

[00:54:02] JM: I totally interrupted you and hijacked it, but why was it useful for the emoji cam thing to run Rust in the browser?

[00:54:12] AG: Because they're doing a ton of processing in real-time, and all those processing is happening. If you were to do it in JavaScript, it would take forever. So they're compiling Rust to WebAssembly and they're using that binary to do all the processing.

[00:54:28] JM: Okay. Yeah. Right. Okay. Let's talk a little bit about business. You did raise money. Was it seed or series A?

[00:54:35] AG: It was a seed.

[00:54:36] JM: Seed. Okay. What are the inefficiencies of the venture capital ecosystem?

[00:54:42] AG: I believe a lot of money is still raised based on who you know, and as supposed to pure merit. The people making the decisions do not have all the information, which is why venture capital is not like 10 out of 10 business. It's a 1 out of 10 business.

[00:55:06] JM: 1 out of 10. Oh! You mean frequency-wise?

[00:55:09] AG: The number of companies that they fund that end up really hitting it out of the park are becoming as successful as they thought they would become is a very, very small fraction of the actual number of companies funded. I think there's a study recently that said they would last 10 years. Most venture capital firms have not had positive returns.

[00:55:30] JM: That's such a meaning study.

[00:55:31] AG: I know, but I think it does speak to the fact that venture capital, even though it's essential, is not necessarily, like you said, efficient. Part of the reason for that is the uncertainty of any early-stage project, like you just have to believe something and take a risk or take a bet on something. More often than not, that bet is not going to play out the way you expected. But I believe right now at least, it is hard for people who are not well-connected or who don't fit the typical Indian nation, white male profile to – It is harder for them to raise money. That's changing. I think that more women are entering venture capital, and I think that's a really good thing.

But when I was raising money for my seed, I believe of all the funds I spoke to, I didn't really see a single female partner. I spoke a lot of firms. I did meet – Well, no. I take that back. I think I saw maybe two out of – I don't know, 50 partners, who were female. I hope that changes, and that in itself is also big in efficiency.

Then there are things like politics within firms, where you have to be politically powerful enough to get a deal through. I've seen that happen –

[00:57:11] JM: Oh! So it's like win over the associate, and they can't win over the senior leadership.

[00:57:17] AG: Yeah, exactly.

[00:57:17] JM: That crusty old readership didn't believe in a new cloud provider.

[00:57:21] AG: I'm not saying that, but that does happen. It's not just partners over associates. It's also junior partners versus senior partners, like people who've been there for 30 years versus someone who joined two years ago and is still a partner, but is a "junior partner".

[00:57:42] JM: I cannot believe people would've passed on you though for any kind of – I mean, speaking of fashion-driven development, I mean, employee number – What were you? Employee number five at Stripe?

[00:57:53] AG: Eight.

[00:57:55] JM: What? You fit the fashion bill like nothing else in venture capital. Why would people pass on you?

[00:58:03] AG: When I was raising, it was just me and a prototype, and it's very easy to pass on a seed stage company, because you have many more hits to go later in the game especially if you're a well-known firm, because the next round, there is a good chance that I will talk to those firms again. So you have more data and you can choose – So VCs loved to delay as long as possible. So this is another form of delaying things where they might say no initially, and then they end up investing when they think the risk is lower or that the business has been proven out more.

So now, Render's revenue has been growing on average over 50% month-over-month for the last six months. We're much shorter a bit now, and we're serving well over 100 million requests every week and it's really interesting to see the growth in our numbers without any sort of major marketing efforts on our part. We haven't spent any money on marketing specifically. I mean, I think that this interview and the one that we did before, I'm sure they helped, but we're not sort of running any kind of ads anywhere. All our growth has been organic in that sentence, and it's really great to see people moving over from Google Cloud, from DigitalOcean, from AWS, because they don't want to deal with all the crap that they had to deal on these other clouds.

So going back to your question of why someone would pass. People pass for a variety of reasons. People pass because the valuation that I asked for does not fit the model that they have.

[00:59:57] JM: At the round where price sensitivity matters so much.

[01:00:01] AG: Well, I think that if you are only a seed investor, then you have to have certain investment goals and ownership goals that you want to meet at the seed stage. It makes sense for you to be price-sensitive, because you won't get another chance to get more of the company later on.

[01:00:22] JM: Well, it's more like ownership-sensitive rather than price-sensitive.

[01:00:25] AG: They're the same thing more or less, because I only want to give away X-percent of the company and I want this price. If I say that I want – Hypothetically, if I say that I am only raising $10 million and I want a valuation of 100 or 50, or let's say 100 for the sake of simplicity, then you get a 10% ownership. But if I was raising 10 million at 50, you'd get 20.

I think you have VCs and most investors have to hit certain ownership targets in every round for it to make financial sense for them if the company does really well. Because if you have like 5% of something that does really well, especially at the seed stage, you'll get diluted significantly as an investor. For some people, it was just like they loved Render, but the price was too high.

[01:01:20] JM: The gender side of things, we have a little bit of a discussion about that, you and I, in regards to how it impacts your hiring philosophy.

[01:01:30] AG: Yeah.

[01:01:31] JM: Can you just tell me a little bit about that?

[01:01:33] AG: Sure. I believe really strongly in building out a gender-diverse company from the get go, you can't get to 20 engineers and suddenly realize,"Oh! We have to hire a woman, because now we have 20 guys." It doesn't work that way, because by that time your culture is kind of already set.

[01:01:56] JM: Culture compounds.

[01:01:58] AG: Yeah, exactly. It is so important to bring in all kinds of diversity, not just gender diversity, as early as possible into the company. So not just so you can have that culture from the very beginning, but also so you can attract people from diverse backgrounds, because they see that this company isn't just a bunch of white and Asian dudes. This is more of a long-term investment as well. But having said that, I think it's very hard to do. Especially when you're a startup trying to go as fast as possible, there is an argument to be made that at a very early stage you should not be thinking about diversity, because all you should be thinking about is trying to get the engineer so you can build the product and get users and grow.

I think that if you have the ability or the background and the money and the investors who can support you in trying to build a diverse team, then you should absolutely do that. So we got lucky with Render, because we didn't have any investor pressure in terms of, “Oh, you need to hire these many people by this date.” It was all like whatever you want to do. Steve Herrod, who is our lead investor, was the former of VMware.

[01:03:15] JM: Previous guest on the show.

[01:03:16] AG: Oh, he was?

[01:03:17] JM: Twice.

[01:03:18] AG: Oh, really? Yeah, he’s great. I love him. Has been super supportive of how we hire, and he’s savvy.

[01:03:26] JM: Good choice investor.

[01:03:28] AG: Yeah. I mean, I back channeled him quite a bit. Even when someone's company went south and they were working on Steve, I remember them saying, “Steve was great. Maybe not the fund as much, but Steve was great.” Also, I spoke to people who worked with him when he was at VMware. This goes back to saying, if you're raising money, make sure to back channel your VCs and make sure to talk to every company that failed that VC is working with, because when things are going well, nothing really matters. Everyone's happy and people don't really try to make you do things. But it's when things are not going well that the true nature of a VC founder relationship comes out.

[01:04:16] JM: The gender diversity stuff, this has always struck me as quite hard to figure how to get early at a company. But I've talked to multiple companies that I have a lot of respect for that have out and out froze hiring at 3 to 4 engineers, because they did not have a female yet. To me, that seems like that’s like a strong action and it's – The thing is, like talk is cheap. This is one thing you know.

[01:04:46] AG: Talk is cheap. Yeah, you can talk about it all day long.

[01:04:49] JM: Absolutely, and action says a lot.

[01:04:51] AG: Yeah. So we did the exact same thing. We froze hiring in terms of talking to male candidates, and we did this for – And we desperately needed people, because we'd launched and our users were looking for all these features that we needed to build yesterday. So was it a collective decision from our team. We were a team of four people back then, and we had just one woman on the team, but we wanted to make sure that it was more evenly balanced. I decided that we would freeze hiring until we could get more women on the team, and it was tough. It was tough as hell.

We interviewed tons of people. We advertised in a lot of different places where it was more likely for female engineers to see Render as a place to work. Then at the end of three months, we gave in, or I think at the end of four months where we said, "You know what? This hasn't worked for us. We'll have to open up the pipeline." Somewhat completely out of the blue and extremely fortunately for us, we were able to hire two engineers who were women right when we gave up. So we had opened up the pipeline to other candidates, but it just so happened that we were able to find two amazing engineers who happen to be female and who are now on the team. Render is now 50% female.

[01:06:35] JM: That's a coo. What's hiring like in Silicon Valley in 2019?

[01:06:41] AG: Huh! It's hard. Especially as a startup, I think that it's obviously really hard to compete with the crazy salaries that Google and Facebook and all these big corps offer you, not just Google and Facebook, but even companies like Stripe, and Airbnb, and Dropbox are offering, because they had to compete with the Googles and Facebooks as well and they have the cash. So they can afford to offer great salaries.

When you're a seed funded company, you focus much more on equity, and living in San Francisco as we all know is tough on a low salary. So I think it was hard for us in beginning, especially when we hadn't launched. It was hard, but then it became progressively easier. Because after we launched and then we got a ton of user love, and even now when you search

for us on Twitter, you'll see a lot of people sort of raving about Render. So there is a lot of social proof now that people can look to and they know that this is the real thing. Whereas before, before launch and before when we were just like two people,
It's like, "I don't know what's going to happen here." I remember joining Stripe before launch and I had no idea what was going to happen, but it seemed like a cool team. I was like, "Yeah, why not?"

[01:08:06] JM: Yeah. See, that's funny, because I guess you didn't see it from day one. You didn't realize that this is going to be like the next –

[01:08:12] AG: No way. No. I mean, I actually refused to talk to them the first time they asked me, because I was also – I don't know if I've mention this, but that I was also the first Stripe user to move away from Stripe to Braintree.

[01:08:29] JM: Oh! No way. That's hilarious.

[01:08:32] AG: I was, yeah. This was back when Stripe hadn't launched, and a few – This was like early 2011. Stripe's product was just a thin layer over authorize.net and they were trying to build out the real thing, but –

[01:08:47] JM: What a brilliant MVP, by the way. The fact that they – What are they like? They've basically found some payment processor that they could kind of – I guess they would be white-labeling the payment processer with a better UX. It was like API in front of this just service that was –

[01:09:03] AG: Fax machine. It was literally an API in front of –

[01:09:05] JM: Oh my God!

[01:09:06] AG: So people submitted their application form online, and then one of my friends, Dara Buckley, he went and faxed that, printed that out and faxed it to authorize.net.

[01:09:19] JM: Why did you move to Braintree? Just because it was like they were a fax machine API?

[01:09:24] AG: No. I didn't know that at the time, but they were missing some key features, and Braintree had been around for like 7 or 8 years by that point.

[01:09:35] JM: Man! The Braintree story is agonizing. That company was hard to build.

[01:09:40] AG: It was, but at the same time I think that they focused on a segment that in the end ended up not being as significant for them as developers. This is why focusing on developers is just so crucial.

[01:09:55] JM: Dude, I think they might've been too early if they would've focused on developers. The Stripe timing was –

[01:10:00] AG: Timing is always a part of this, but I think that Braintree could have focused on developers even sooner, because I remember asking Braintree for payments prelaunch before – This was a side project that I was building, and it involved payments, and I had PayPal but I also wanted Braintree to have credit cards right on my site.

They said to me in 2010 – So not long before Stripe launched, and Stripe had already been private beta at the time. They said to me, You either have to have VC funding or you have to have launched." I was like, "This seems to be a bit of a catch-22 situation," because I want to have these credit card payments on my site but they were like, "No. You're too high- risk. We can't accept you."

I mean, they knew that I wasn't doing anything illegal. It was a fairly straightforward thing, but they just didn't want to deal with me. After I launched, they were happy to take me as a user when I switched over from Stripe to Braintree. But then I stayed in touch with Stripe, because I was also having some problems with Braintree and I kind of continued to talk to Stripe about the problems that I was facing. Patrick, who was the CEO, very wisely offered to help me with my Braintree problems also. So we stayed in touch and then he was like, "Hey, do you want to join the company?" I said, "No. I don't."

Then a couple of months later, my wife was moving to Boston for grad school and I was working from home at the time, and we wanted to continue to live in California. So we decided that I would stay back in California, because she was going to be living in the dorm at MIT and it was good be extremely crazy. So she's going to be busy all the time. So I stayed back, but I did not want to work from home by myself, and I kind of didn't really want to interview anywhere. I was like, “Well, these Stripe guys seem cool and already asked me if I want to join them. Why don’t I talk to them again?” I talked to them. I went in on a Friday afternoon, sat there, encoded with them a little bit and got dinner with them, and that was that.

[01:12:24] JM: That's an interesting iceberg of a story, and we won’t go too much deeper into that. But I guess to wrap up, after spending a lot of time there, what’s your perspective on how the Internet of money is going to unfold?

[01:12:41] AG: I think it’s still very early. I think that we’re still in the sort of teenage years of online commerce. If you look at the stats, most commerce is still happening offline and cash is still king in many ways. Obviously, we’re going to see more and more and more of that move online in a lot of different ways.

So it's going to be very interesting to see what happens with commerce over the next 10 years, because I’ve also noticed the resurgence of individual stores online as supposed to just everyone going to Amazon. There are many more stores online that people are buying from that are not Amazon. But that doesn't mean that Amazon isn't growing, right? Obviously, this means that the entire market is growing. As someone who holds stock in Stripe, I am biased, but I think that Stripe is really well-positioned to take advantage of that opportunity.

[01:13:49] JM: Totally. Yeah, I also saw Amazon was buying up a bunch of vacant shopping malls. So we may see – You see these interesting cycles.

[01:13:59] AG: Oh, it's amazing. Yeah. It's so amazing to be in tech, because everything is changing all the time and the rate of change is accelerating. You can be part of the change and help make a difference. That's what's exciting to me, personally.

[01:14:17] JM: Anurag, thanks for coming back on. This is really fun.

[01:14:19] AG: Yeah, thank you. It's always really fun to be on the show.

[END OF INTERVIEW]

[01:14:32] JM: As a programmer, you thinking object. With MongoDB, so does your database. MongoDB is the most popular document-based database built for modern application developers and the cloud era. Millions of developers use MongoDB to power the world's most innovative products and services; from cryptocurrency, to online gaming, IoT and more.

Try MongoDB today with Atlas, the global cloud database service that runs on AWS, Azure and Google Cloud. Configure, deploy and connect to your database in just a few minutes. Check it out at mongodb.com/atlas. That's mongodb.com/atlas.

Thank you to MongoDB for being a sponsor of Software Engineering Daily.

[END]