Life of An SRE with Shannon Brady and Theo Klein

[THEME MUSIC] [NON-ENGLISH SINGING]

THEO KLEIN: Hi. My name is Theo Klein, and I've been a site reliability engineer here at Google for just over three years. Currently, I work on the reliability efforts around the infrastructure that powers the ingestion, curation, and storage of all of the data that powers Google Maps. And this is essentially my first SRE job, my first real job out of college, and it's been super awesome.

Yeah, sure. So I came into SRE pretty much straight out of college. At university, I was studying cognitive science and computer science. And I really had no experience in computer science prior to university.

In university, I didn't really understand what the industry was like. And I stumbled into an internship at Microsoft where I was doing software engineering, like, pure software stuff. You know, I was working full stack on Azure.

And I knew that I wanted to be in New York City after university. And unfortunately, Microsoft didn't have any offices, any engineering offices at the time in this area. So I applied to Google. And at the time, the only position that was open was in site reliability engineering.

And I said, what is site reliability engineering? I have never heard about this before. What is this? And then initially, I was really skeptical. You know, I thought, OK, I'm gonna be doing a lot of DevOps stuff. I'm not necessarily interested in DevOps. What I like working on is algorithms and improving code.

So despite that skepticism, I knew that I really wanted to be in New York, so I applied. And I ended up getting the job. And the thing that really, like, held me into this position initially was the fact that, if I didn't like it, I had the opportunity to transfer from the position called SRE-SWE to full SWE. So SWE means SoftWare Engineering.

So I told myself, I'm gonna give it six months and see if I like it. And if I really don't like being a site reliability engineer, then I'm going to change teams internally after six months of being here at Google. And I'll just give it a shot.

And so I joined Google. And you know, for a variety of factors that I'll get into later, I actually fell in love with site reliability engineering. And so I've stayed in SRE for the past three years. And I've actually stayed on the same team for the past three years.

MP ENGLISH: Our skills complement each other, truly. Mm-hmm.

Mm-hmm. Yeah, that's a really great question. And I totally agree with you that individual members on my team have specific expertise in different areas. And you know, with that said, if you have an interest in an area that you are not an expert in, I think that, at least in my team, we're very encouraging to allow people to explore.

But in particular, with my journey and finding my expertise, I had always signaled an interest in doing coding projects. And so from the very first project that I worked on, I was essentially, like, working, embedded in a specific developer team to improve the reliability of their system through software engineering projects.

So I think my very first project was working on a very important core piece of Google Maps infrastructure. And it was running on some very legacy binary dating back to, say, like, 2008. And no one had touched it, right? The only thing that was updating in this binary was the data that was being served by this binary.

And you know, times change. Systems change. And the requirements change for the production environment. And none of those changes had been implemented in this old binary. And so my goal, my responsibility for this first project, was, go into this extremely important piece of system and modernize it. Bring it up to date with the best practices of SRE today.

And so can you imagine as, like, a new grad coming out of college and then being handed a project that says, take this central piece of software? If it breaks, you break Google. Right. But if it breaks, you break Google. But you are now responsible for bringing it up to times.

And that was so crazy. You know, that first day after I left work for that day, I was so scared. But at the same time, I was also really excited because I knew that I was going to be working on a really important project.

And so it took me a few months, and I launched it. The launch window was several months long for different parts of Google. And it was a huge success. And so that sort of told me that I needed to find the scariest problems in the problem space that we're on call for and find ways to solve that with engineering.

So over a few projects after that, I kept on doing similar types of projects where we take old systems and we want to modernize it, or we want to migrate backends. And so I'm not necessarily building new features, right? The API layer of the system remains the same, but the internals are changed such that the reliability is increased.

THEO: The last project that I've been working on is a project that we call Zero Outages. And the goal of this program is to essentially eliminate all serving outages from the services that we're on call for. And in doing so, the way that we do this is that we're actually automatically analyzing all of the dependencies in the system and ensuring that the dependencies that are in our system are meant to be there.

And so we have a system that-- or, like, an auditing tool that will flag risky dependencies. And then we do engineering projects to remove those risky dependencies. So in that way, it's sort of like a two-pronged approach. We've got the auditing tool that will generate alerts. And so I put on my program manager hat to manage all of the alerts that are generated. And then I do engineering projects to mitigate or resolve those risky dependencies.

What I really like about this is that it really demonstrates the variety of areas that SRE has expertise in. You're working in quite a low level, in like the tech stack, so to speak, right? You're working very deep on the internals of Linux and how Linux operates and how all of us at Google get to use Linux. Whereas, I work at the very top of this stack where I'm thinking only about the business logic and, more abstractly, how the different components of business logic interact with each other.

And at the same time, like, we need SREs and-- throughout this whole stack, right? We need SREs that focus on the internals of Linux. But we also need SREs to think about the broader view of how our systems operate or how the business operates through the code.

Yeah, and to add on to that, the postmortem culture, especially in SRE, I think is extremely strong. And it has allowed me, as an individual, but my team to try out risky projects and risky actions without the fear that we will suffer the blunt of the repercussions if the risks don't pan out in the way that we would like them to.

And so something that I always think about when I go on call, for example, is that my-- is something that my manager repeats often. And it's that my manager will always have my back for every decision that I make when I'm on call, even if my action-- as a consequence of my action, the system operates even worse than at the beginning of the incident.

And we've had moments where, you know, I have issued a command or a colleague has issued a command that has globally taken down a service, right? And the way that we interpret that is not that the-- that this engineer has made a mistake and that they should be reprimanded. Rather, the way that we interpret that is that the system was designed in such a way that allows this vulnerable action to be made by an engineer, by a human.

And therefore, the system is at fault. And we should change the system to be more safe and prevent those actions from being made in the first place. And that nuance, I think, is so important because it allows us to make faster decisions and to worry less about the potential negative effects that may happen as a result of those.

And I guess another point here is that, by changing the system as opposed to changing the person, we can ensure that, when that person leaves the company and a new person joins, that same mistake won't happen again, right? So these rules are encoded in the system as opposed to the person and their mental model of how, you know, on-call operates.

One thing that I think is really cool as well is that this postmortem culture has sort of bled into my personal life and my social life. So you know, I have a partner. And at the beginning of our relationship, it was very much me versus them. I can't believe you didn't do X. Or I would receive, like, I can't believe you didn't do the dishes. Like, you have to think about this and that, right?

And with this blameless postmortem culture, we're sort of reframing the disagreements or perhaps the negative things that happen as a factor of how the system got us into that situation. And so now it's much more about me and them against the system. And that has become so beneficial for my relationships and for my social network.

I was giving a, you know, a small informal talk to some folks at-- when I was traveling. These were high school students. And I was giving a talk on this blameless postmortem culture. And I was stepping through how we do this at Google. And I was giving very technical examples.

And at the end of the talk, a teacher came up to me and said, oh, my gosh, I wish you had been here six months ago. And I say, what do you mean? And this teacher elaborated that they had gone on this international trip to Germany. They're based out of Argentina.

And in this trip, there were many things that went wrong, right? There were certain incidents, so to speak. And the students that experienced these incidents decided to-- or they were very frustrated because there was something about maybe, like, a bus not working properly. Maybe there weren't enough seats, or perhaps they were late.

And the students were understandably frustrated. But they didn't really know how to express it. And so there was a lot of blaming going about. You know, some teachers were blamed. Some administrators were also blamed for whatever happened. And they realized that they could have used this blameless postmortem culture to analyze the system in a way that would have been much more constructive.

And so after I spoke to them about this, they actually reanalyzed all of the failures that had happened on this trip. And they were able to make some changes, which I thought was so cool, right? So this culture of blaming the system as opposed to blaming the people, I think, has ramifications beyond just SRE and, you know, computer science, right? But it's a very social thing that, I think, allows us to make more efficient progress.

Yeah. To add on to that, I actually-- I would be even more-- I would be even stronger in my opinions about postmortems in particular. I love writing postmortems. And I love talking about postmortems, specifically because I think it's so cool to see how our systems break and then to just sit around in amazement and say, oh, my gosh, like, these were the steps?

Like, this is so complicated. How on earth did-- you know? Da-da-da-da-da, right? And I think that that is so exciting and fun. And so I guess that this description shows you that you can reframe these incidents in a positive light as an opportunity for learning as opposed to a scary experience, like you were describing, Shannon.

Yeah, yeah.

And SREs, in some ways, we need postmortems because a lot of times, engineering work that we that we implement comes as a result of those postmortems because they identify systemic risks in our systems. So it's part of our lifeblood.

Yeah, so psychological safety is really important to me. And it means a few things. And I guess the best way to describe what psychological safety is, for me at least, is to give some examples.

So psychological safety to me means being able to be vulnerable with my coworkers, so sharing parts and details that may put me potentially in a bad light. Let's say maybe I wrote something that didn't work properly. I'm able to share that without the fear of being accused of something.

And in some ways, that's sort of like a growth mindset, right? We're all working to become better individuals and better coworkers and better engineers. And every opportunity that we have in life is an opportunity to grow.

So I'm able to be vulnerable with my coworkers. I'm also able to provide feedback, opportunities to grow, to my coworkers and I'm also comfortable enough to receive feedback from coworkers. And I have that space to process that feedback without necessarily taking it the wrong way.

I also, as I said earlier-- I think this-- you know, we had some examples, both Shannon and I, where we are able to take risks and make mistakes without the fear that I'm going to get fired, right? I know that I'm not gonna get fired even though I'm going to make mistakes at my job. Again, because we're learning, we're going to make mistakes. And the only way to learn is to try things out, try new things. And sometimes, those efforts don't go in the right way.

So additionally, psychological safety also means that, in the rare case where something is uncomfortable socially or technically, I know that I have someone that I can reach out to that isn't necessarily HR. You know, I know that I can go to my manager, and I can share very vulnerable things. And I can bring up conflicts, or I can even bring up technical issues. And I know that my manager has my back. My manager will go up to bat for me, and that makes me feel really safe.

So in short, I'm able to be vulnerable with my coworkers and show the opportunities where I can learn. I can also trust that my coworkers feel the same way. And I know that my manager is able to provide that safety net. In the case where, let's say, other teams have issues, or maybe they are trying to push for things that we're not comfortable with, I know that my manager will step up to bat for us and protect us.

That, I think, is something that has become even harder with the pandemic and with COVID-19 because we were all working from home. So the way that I think many people would de-stress after working is by taking some time during their commute to go home. And maybe you're in a different space physically, so you can potentially leave that stress outside in the office.

And that's how I dealt with the stress of on-call and other sorts of work-related issues. However, at the start of the pandemic, when we all went to work from home, I was working 2 feet away from where I was eating, which was, 10 feet away from where I would sleep and 15 feet away from where I would shower, right? All of these things are within 20 feet of each other, maybe give or take 10 feet, right?

And my partner would work behind me. And so that caused a lot of issues because my world just shrank. And I had no space to leave my stress when I would end the workday.

It was initially really challenging to separate work from life. And you know, I tried a few things-- vegging out on the couch, scrolling on TikTok, on Reddit, et cetera. And that worked to a certain extent. But actually, what ended up allowing me to separate work from life was actually getting a dog.

This isn't why I got a dog. I had wanted a dog for a long time. But it was one of the happy side effects of raising a puppy because dogs actually have needs, like humans. And they are very communicative as to when they need something. And that has actually played a really big role in how I start and stop work, both at the beginning of the day and the middle of the day and at the end of the day.

And you know, for example, my dog, Biscotti, who is super cute-- and now she's curled up in a ball, sleeping on her dog bed. But she essentially starts my workday right before it because I have to walk her in the morning. And I take a 15-, 20-, 30-minute walk in the park, where I'm essentially meditating just by virtue of walking in the park. At lunch, I get the opportunity to spend time outside, get some fresh air, come back for lunch, not thinking about work, but thinking about my walk in the park and my playing fetch with Biscotti.

At the end of the day, at 5 o'clock on the dot, on the dot at 5 o'clock, she comes up to me. And she pokes me on the foot. She nudges me. She sits down next to my desk and tells me the workday is over. Playtime is now. Let's go to the park. And so I realized, like, OK. Like, let's go, right?

[DOG BARKING]

Oh. There she is, right on cue. And it's almost noon. So I have to take her out soon. So she has given me those specific points where I understand, like, this is the end of the workday. And by going outside, by walking her, by playing with her, I am able to reset myself and then come home not in work mode, but in life mode.

MP: I totally agree, particularly because you can silence an alarm, but you really can't tell a dog to-- I don't know-- to deal with their needs, right? They need us for a lot.

THEO: I think your point about proximity to tech and silencing notifications is a really salient point. And it's something that I don't do enough of. And I think this problem is compounded by the fact that a lot of us are really passionate about what we do. And so we want to continue working, right? At least, I want to continue working on the problems that I care about.

And so it's super easy-- if a notification brings us back into the workspace, it's super easy to say, oh, OK, yeah. Like, let me just do this for another 5, 10, 15 hour, you know? So it's really important to be rigorous about when the workday ends and hold yourself to that because it's super-- especially in the life of-- or in the world of work-from-home, it's extremely easy to revert back into the workspace after the workday ends.

Now, I also want to make another point, which is that I'm making these clear distinctions between work and life, and I'm also adding sort of time boundaries. But on the flip side to that, I also know that in the case where life comes up and a pressing issue appears, I know that I have the ability to take time out of the day to handle those.

So again, with respect to my dog, the vet is only open from 9:00 AM to 5:00 PM, which is usually when people work. I know that I can take two hours out of the day to handle a vet appointment and then come back to work, knowing that my coworkers understand that life comes first. So that flexibility, I think, is really valuable.

You can have the regular cut and then the director's cut. And the director's cut could just be the raw stream. Exactly.

[THEME MUSIC]

[NON-ENGLISH SINGING]

VOICEOVER: "Prodcast," the Google SRE production podcast is hosted by MP English and produced and edited by Salim Virji. Engineering by Paul Guglielmo and Jordan Greenberg. Javi Beltran composed the musical theme. Special thanks to Steve McGhee and Pamela Vong.