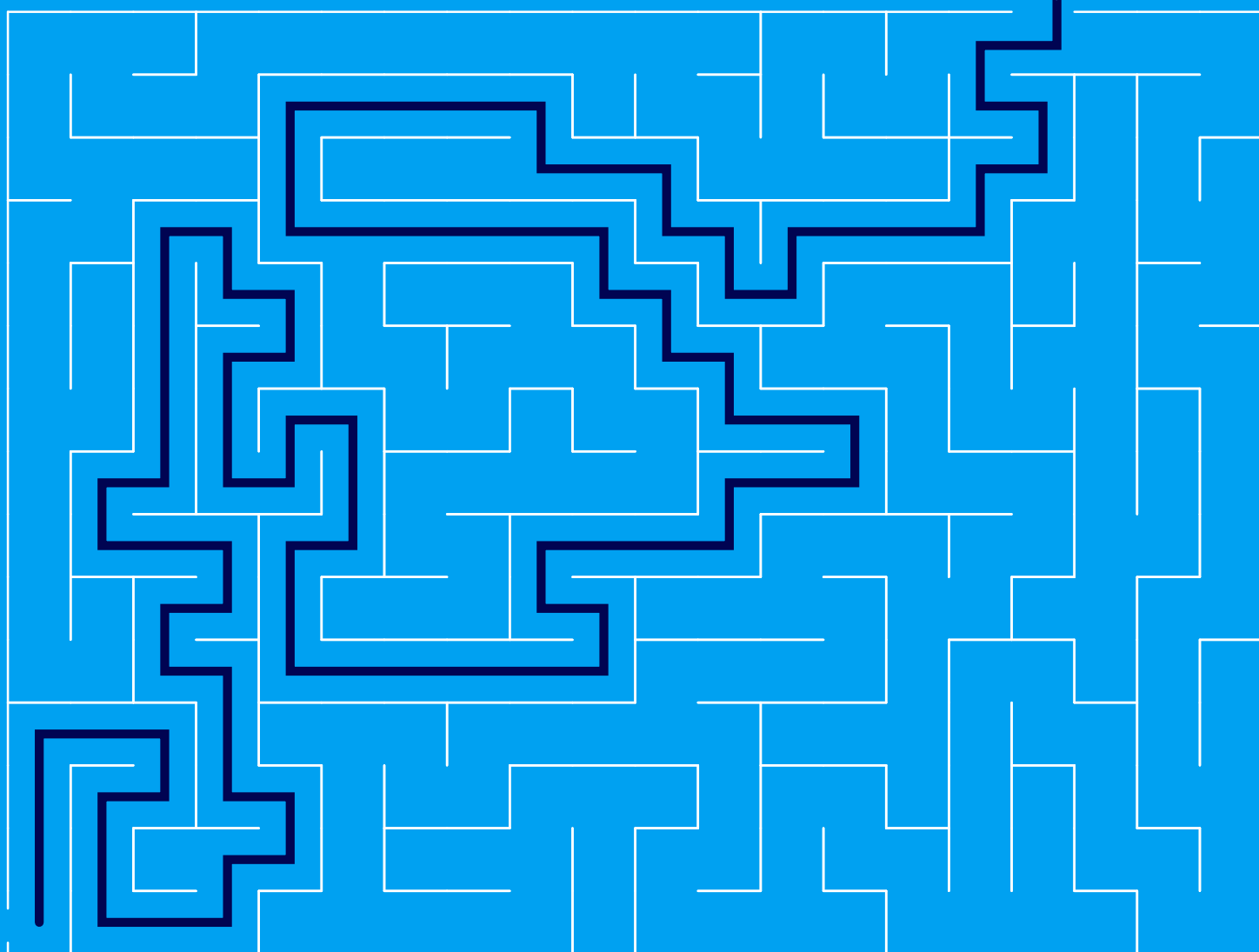


FOREWORDS BY YADIN PORTER DE LEÓN

KUBERNETES WORLD FINDING YOUR PATH

BART FARRELL, GULCAN TOPCU



Copyright © 2025 by LearnKube

All rights reserved.

No part of this book may be reproduced or used in any manner without written permission of the copyright owner except for the use of quotations in a book review. For more information, address: hello@learnkube.com.

This book was made possible through the generous support of Heroku from Salesforce.

Foreword

There's no manual for breaking into Kubernetes.

You can read the documentation, watch the talks, and earn the certifications. But none of that actually teaches you how to become part of the community—how to move from watching on the sidelines to becoming someone people recognize and trust.

This book exists because that gap is real, and it's bigger than most people acknowledge.

It's built from conversations with twelve people who've navigated this path themselves. Some started as wedding photographers, others came from political science, or spent years in closed-source corporate environments. They didn't follow the same trajectory because there isn't a single path forward.

What they do share is this: they showed up even when they didn't feel qualified. They asked questions when they felt stupid. They kept going when nothing seemed to be working. And somewhere along the way, they became the people who open doors for others.

The experiences of those we interviewed for the book

Bob, along with others we interviewed, needed someone to literally hand him a t-shirt and tell him to sit down before he felt like he belonged. Taylor was rejected from the CNCF ambassador program multiple times before finally being accepted. Katie watched Kubernetes grow from hallway conversations to 13,000-person events.

Lin and Kaslin have done it all—contributing code, building communities, and educating thousands of people. They've moved between different roles as their careers evolved, proving there's no single way to make an impact.

Cortney went from studying political science to running a CNCF project. Phil spent twenty years in closed-source environments before becoming a recognized voice in the container space. Prasanth started with an LFX mentorship and is now mentoring others himself.

Emily learned Kubernetes through whiteboard sessions with her team. Yasmin learned through YouTube shorts and ended up hiring people based on how they think, not just what they know. Jorge has been in cloud native for a decade, watching the entire ecosystem evolve. Whitney keynoted at KubeCon less than two years after writing her first line of code.

All of them figured out how to keep going when things got hard.

The people in this book started confused, out of place, and uncertain.

What they did next is the subject of this book.

Now you have the book they never had.

Yadin Porter de León

Kubernetes World Finding Your Path

Table of contents

Chapter 1: The People Behind the Platform	8
Getting Started: The Human Side of Entry	9
Mentorship and Leveling Up	12
Beyond the Code: Invisible Work	14
Community Values and Culture	16
Chapter 2: The Paths	18
The Contributor: Writing the Future	20
The Community Builder: Opening the Circle	21
The Educator: Making It Click	22
Crossing Between Paths	23
Finding Your Path	25
Your Combination	26

Chapter 3: Building Credibility (Without the Map)	27
When Nothing Makes Sense	28
The Power of Admitting You Do Not Know	30
The Work That Nobody Sees	31
The Skills That Actually Matter	32
The People Who Open Doors	34
From Invisible to Visible	35
The Long View	37
What Actually Works	38
Building Your Credibility	39
Chapter 4: Networks, Learning, and Growth	41
The Accidental Journey	42
Learning in Public	44
The Power of Not Knowing	45
Networks That Matter	46
The Long Game	47
Finding Your Way In	48
The Currency of Trust	49

The Community's Future	50
Your Turn	51
Chapter 5: Sustaining the Journey	52
You Can't Know Everything	54
Building Your System	55
When Your Role Changes	57
The Human Side: Small Contributions Matter	58
The Pressure and the Doubt	59
Staying In for the Long Run	61
Contributors	63

Chapter 1: The People Behind the Platform

Before the code, before the architecture, or any feature reaches production, there are people. People who showed up uncertain about where they belonged, those who kept asking questions when they felt out of place, and those who stayed long enough to understand how things worked.

The first chapter is about how these individuals discovered Kubernetes and what they learned while building something that mattered.

Getting Started: The Human Side of Entry

The path into Kubernetes often doesn't look like what you expect. There is no formal application process or a door marked "Contributors Enter Here."

People stumble in through side doors or keep showing up to meetings until someone finally remembers their name.

Bob Killen's entry sounds almost accidental, but it was not.

It was 2017. Bob worked at the University of Michigan Hospital, where his team had wrestled with container platforms for clinical workloads. They built their own system, then transitioned to Docker, and later briefly switched to Mesos when early Kubernetes proved too unstable. By 2016 and 2017, they were re-engaging with Kubernetes heavily, watching it mature into something they could use.

Bob had been following the project from a distance, paying attention to conversations and tracking its development. When KubeCon came around that year, he reached out to Jorge Castro, someone he had met at a meetup. He went down there, texted Jorge for directions, and ended up at the Kubernetes contributor summit. Bob admits he felt like an imposter: "I felt I shouldn't be there, as I wasn't a real contributor." Jorge handed him a t-shirt and told him to sit in the room. "That is how I pretty much became a contributor."

Inside that summit, Bob met Paris Pittman in person for the first time, along with Christoph Blecker, Ben the Elder, and Tim Hockin. They started talking, and Bob found himself drawn into ContribX, the contributor experience special interest group. Jorge was heavily involved at the time. Bob describes why

ContribX worked for him: "It was something I could engage with pretty easily as a hobby contributor without having to sink a lot of time into it."

Bob had been following Kubernetes for years and had real-world experience running it in production. He still needed someone to hand him a t-shirt and tell him to sit down before he felt like he could call himself a contributor.

Taylor Dolezal's path came through a different door, but it required the same persistence.

Taylor started as a Ruby on Rails developer. He was scared because all of his work was put out into the open. "What if I do something wrong?" Taylor remembers. As he continued working with Ruby and Rails, he moved into a role at the Cleveland Clinic. Docker had just come out, and Taylor started working with containers while transitioning from software engineering into infrastructure. He discovered Kubernetes at a meetup, drawn to how it simplified the process of running WordPress. At Disney Studios, the push for Kubernetes became official: "There was a mandate that we all wanted to use Kubernetes because it would make it easier to work across teams and have a common language."

As he gained confidence with the technology, Taylor wanted to get more involved with the community. He looked into the CNCF ambassador program and began attending Kubernetes meetings. The process wasn't smooth: "In its early days, it was difficult to become a CNCF ambassador. I got rejected many times before finally being accepted."

That feeling of rejection is something almost every contributor experiences. Taylor kept showing up anyway. Eventually, community members like Jorge Castro, Natasha Woods, and Kaitlyn Barnard reached out and invited him to join the shadow release team. Taylor started working in the communications role within the SIG release team. "Since Kubernetes 1.14, I've been deeply involved with the community, learning how the pieces fit together. I've never looked back."

Persistence changed everything.

Taylor breaks down how anyone can start contributing: "No change is too small. Step one, show up. Step two, stick around. Step three, community."

While Bob needed someone to hand him a t-shirt, Taylor needed to survive multiple rejections, and Katie needed to see the evidence with her own eyes before she believed the hype was real.

Katie Greenley started at the CNCF, working on the events team. Her first real

contact with Kubernetes came through setting up KubeCon Seattle 2016. She remembers the chaos of those early days, when Dan Kohn was being brought in and Chris Aniszczyk was serving as interim executive director. For her, it was "all the chaos of first forming the project and how exciting it was for Kubernetes."

Katie wasn't sure if the hype would materialize: "It was a groundbreaking phenomenon that you didn't really realize would become a phenomenon. It was like the Beanie Baby bubble: everybody was grasping at it, excited for its potential, telling you it was going to be big. I was skeptical, wondering if it was really going to be big." When she was setting up the first KubeCon in Seattle, people kept telling her it would sell out, that she didn't have enough space. "Sure enough, we didn't have enough space that first year."

That first KubeCon had just under 1,200 people, limited not by demand but by fire marshal laws and the physical constraints of the venue. Katie describes the unusual solution they created: "We were selling hallway tickets for people to come and be at the booths because we couldn't get people into the rooms. There was such demand that people wanted to be in the hallway talking to people and visiting booths, which was not a typical thing we did at Linux Foundation events. It's not something we've ever done since."

From that chaotic beginning in Seattle, Katie watched Kubernetes grow to Berlin in 2017, then Austin later that year, and eventually to the 12,000 to 13,000-person events happening around the world today.

Each of these three began with uncertainty and became something else through persistence. But once you are in the door, the next challenge is figuring out how to grow. How do you learn not just the technical details but the unwritten rules, the cultural norms, the way decisions actually get made?

The answer comes down to mentorship, the kind that develops naturally when someone recognizes potential and chooses to invest their time.

Mentorship and Leveling Up

Mentorship rarely begins with a title or a plan. It happens when someone takes a little extra time, answers one more question, or says, "Come sit with us."

For Bob, that person was Paris Pittman: "Paris was kind of my mentor in ContribX. She eventually stepped down, and I took her seat." Bob also credits Jorge Castro as "the one who really dragged me and Jeffrey Sica into contributing to the project."

Three people mentored Taylor: Bob Killen, Jorge Castro, and Jeffrey Sica. Taylor describes Jorge's energy: "Jorge was there from the beginning with me in Kubernetes and is just a wonderful soul who has never lost enthusiasm. I need to figure out his secret because I'm fairly enthusiastic, but Jorge has me beat. I don't know if it's something in the Michigan tap water, but Jorge has been a huge inspiration for me in how to embrace, encourage, and help people level up."

Taylor remembers his first meeting with Bob and Jeffrey: "We met in a hotel lobby at a KubeCon—they were sitting on a couch, and I sat across from them. We started talking and became lifelong friends. They helped pull me in, suggested things that might need changing, and gave me advice on how to interact with the community."

Taylor explains why people stay in the community: "The focus is on people, which is by far the secret power of the Kubernetes and open source community. Finding mentors and people to bounce ideas off is incredible. It's not just about the tech or writing the perfect issue or pull request. It's about having friends and people to grow with. It shouldn't be a solo experience—that's my firm conviction."

Katie's mentorship came from organizational leaders who saw her potential and encouraged her to grow beyond her current position.

Dan Kohn stands out first: "Dan Kohn was really pivotal in shaping CNCF in those early days when the sky was the limit. While many people thought it would just be a scheduler, for him, it was going to be everything—so much more than what we even think today." Working under Dan's leadership, Katie found someone who "really loved to grow people and give feedback to help

them be the best version of themselves."

Dan pushed Katie to think beyond her current role: "Anytime I interacted with him, he was always pushing me to think about presenting information more clearly or crisply, and to consider my career path: Where did I want to go next? Were events where I wanted to be, or something else? What was I capable of doing? I believe Dan was instrumental in helping me think about my potential within this ecosystem. He was also crucial in shaping where the ecosystem is today because he truly had a vision of where it was going."

Katie also credits Chris Aniszczyk for giving her confidence to transition into community support work: "He was one of the first people to advocate for me, strongly believing I would excel in that role. Anytime something came up, he'd say, 'Don't worry, Katie's got it.' Having people behind you who have confidence in your abilities—especially when you're experiencing imposter syndrome—makes you feel capable of so much more."

Katie also points to the projects team: "Beyond Chris and Dan, the projects team—Jeffrey, Bob, and Jorge—were incredibly patient. They were willing to teach me about different projects and the technical aspects, allowing me to ask countless questions without making me feel stupid for not having a deep technical background. Those individuals truly helped me realize I don't need to know everything to be successful."

Beyond the Code: Invisible Work

When people discuss contributing to open source, they typically focus on code, including pull requests, bug fixes, new features, and performance improvements. Behind that visible work lies another layer of effort that keeps everything running smoothly.

"I think there are many components to Kubernetes. When people think of Kubernetes, they think of Kubernetes core," Bob says. "But if you look at Kubernetes itself, there are 400 GitHub repos. There are a lot of moving parts and interconnected things."

Each of those repositories needs to be maintained, updated, and reviewed.

"This can go back to some of the invisible work, like documentation, testing, and contributor experience in trying to streamline the project itself and make it easier to contribute," Bob explains. "There is just a lot that goes into running a project of that size."

The scale is almost incomprehensible. "These days, Kubernetes has around 1,800 organizational members, and they've had something like 90,000 contributors," Bob says. "When it comes to thinking about a project of that size, it is like trying to run your own business or organization. It can become unwieldy and difficult to manage."

Bob's path into this work began with ContribX. It was something he could do as a hobby contributor, work that did not require being a full-time maintainer or knowing every part of the codebase, yet affected everyone who tried to contribute to Kubernetes.

This kind of work is hard to prioritize when maintainers are already short on time. "I literally get 50 plus DMs a week asking, _Teach me, _" Bob says. "To be honest, I ignore them unless the person has shown more initiative in their messaging. I don't have time to go through all of that."

Self-starters do well in the Kubernetes ecosystem. "Maintainers are often strapped for time and can't handhold as much as they might want. Self-starters who can learn with just a little guidance will go far." The scale makes one-on-one guidance difficult for most maintainers. "The biggest way people can level up in this ecosystem is to be engaged and ask meaningful questions. Many

people want me to help them contribute or teach me something, but they haven't done proper research." He draws a clear line between asking to be taught and showing that you have already started learning. "If you come to the table having looked at the codebase or a potential KEP, and say that you're interested in contributing to a specific feature, that you've read some of the code and its history, and that you have a few focused questions—that shows you're engaged and ready to do the work yourself."

Taylor sees the same challenge from a different angle. Through his work at Dosu, he focuses on first-time contributor experience and believes documentation reveals project health: "Documentation is always important, especially for developers. Typically, people want to focus on code and features, delivering value. What I've noticed is that docs often come last, which is not ideal—similar to security, it shouldn't be bolted on at the end."

Taylor emphasizes how much first impressions matter: "I'm excited about the work I'm doing at Dosu with the team and community, figuring out better ways to keep people updated and ensure a great first-time experience. Those first experiences matter. If documentation isn't clear, or if commands result in errors, there's a low chance people will return to the project."

Taylor defines invisible work beyond documentation: "I would say that typically it will be mostly around the community—around the demos they create, around the blog posts, around sharing things for events like KubeCon, CloudNativeCon, and how to write a compelling Call for Papers proposal. I'd say it really is about the artifacts and assets that come out of the community, the how-to, the show and tell, really embodying that culture."

Katie saw this during the COVID-19 pandemic, when the entire Kubernetes community shifted to virtual events: "I think this ecosystem matters. The one lesson I carry with me is from 2020, when we all went into shutdown. People could have stepped back and given up on contributing, and no one would have faulted them. However, this CNCF community really persevered and continued."

The community adapted and grew through that period: "We still had virtual events bringing in 20,000 to 26,000 people who wanted to learn and grow together. In that moment, I realized this was more than just technology. This was about the community and what it means and brings to each individual, beyond contributing and driving technology forward. It's about how the community supports you in your career, as an individual, and how they are there for you in times of need. When you're part of an open source community,

especially one as big as cloud native and CNCF, you're making tight-knit friendships and connections that go beyond project contributions. These become lifelong friendships."

Community Values and Culture

What keeps Kubernetes strong is a set of shared values that guide how people work together. These values emerge from daily habits, the choices people make, the examples set by leaders, and the stories shared about what matters.

Katie describes what newcomers should know: "Kubernetes and its community value openness, standards, and collaboration. Recognizing the history of the community and the project is key."

Joining can feel overwhelming at first. You might wonder where to start, which project to choose, or whether you're technical enough. Katie's advice is direct: "The ecosystem today can be overwhelming. Kubernetes itself is still not easy to learn. You can post questions in Slack channels, such as the new contributor channel, and indicate that you're new or have encountered an issue while contributing. People will jump in to help because the community wants everyone to succeed and find their footing."

Taylor offers his North Star for anyone choosing a path in the Kubernetes world: "Think about the impact you'll have, whether it's you right now or your future self. That's my biggest case for writing documentation: even if people say no one will ever see this, you might in the future, and it's a way to help yourself out. Start by asking what kind of difference you want to make. Are you saving yourself or someone else a few minutes? That adds up."

Bob describes what Kubernetes means to him: "a substrate or underlying thing that holds up the new web and infrastructure. There's the Linux kernel and Linux—and then there's Kubernetes. It's kind of crazy to think about where it is in the space."

Bob explains what motivates people to contribute: "I know it might sound like a cop-out, but everyone has their own reasons for getting involved. For everyone, contributing is different. Some do it as part of their job. Others,

especially students, see it as an opportunity. The project is interesting, and people who find their place often stay around."

People choose to build Kubernetes together, teach one another, and foster a culture that values openness, collaboration, and shared impact. Kubernetes was built for people. Building a lasting system begins with creating the people who make it possible.

Chapter 2: The Paths

At every KubeCon, you walk into the convention center with your badge and find yourself surrounded by thousands of people who are supposedly part of the same community. As you listen to conversations in the hallways, you realize they are all doing completely different things. Someone is talking about pull requests and CI/CD pipelines. Another person is organizing a local meetup. Someone else is running Kubernetes in production and dealing with network policies. A speaker is preparing slides for a talk.

They are all in the Kubernetes world, but their daily realities bear little resemblance to one another.

Lin Sun, head of open source at Solo.io, and Kaslin Fields, developer advocate at Google Cloud, have both been in this world long enough to have tried most of these paths. Not because they planned it that way, but because that is just how it happened.

Lin started at IBM in 2016 when her boss, Jason McGee, who was the IBM Cloud CTO at the time, said this was going to be the next big thing. Kaslin graduated from college in December 2013 and began working for NetApp, a storage company. Kaslin describes her early mentorship: "I had wonderful mentors. One in particular, Jonathan Rippy (who is still active in the industry and doing awesome things), was extremely excited about containers. He started to teach me about containers because I was new to the industry and wanted to learn about the exciting new technology."

Neither of them followed a map, and there was not one to follow anyway. They just started walking and figured it out as they went.

The Contributor: Writing the Future

Contributing code to Kubernetes sounds straightforward until you try it. Lin looks back at her first contribution: "I was amazed to see my first contribution was in 2016 to Kubernetes. It wasn't just a simple typo fix, but a more complex configuration issue."

But contributing is not just about writing code. Kaslin has watched enough people struggle with this to understand what really holds people back. She warns about the good first issue approach: "It's not just about writing code—you'll need to understand GitHub, source control, and work within an existing community and codebase."

The most significant barrier is not technical at all. Kaslin quotes Nadia Eghbal's book *Working in Public*: "people don't contribute not because of a lack of technical skill, but because they're afraid of committing a social faux pas."

Kaslin explains what makes onboarding work: "The most important thing with onboarding, especially for a community, is about making the community welcoming and making it clear that you're not going to make a mistake that's going to ruin your life forever. Instead, you're going to learn, and we're going to have fun with this together."

Contributing also requires patience and persistence. Lin admits the reality: "There were times I was contributing to the community, working hard, and running for a lead position without being recognized. As everybody else experiences, there are many chances to do great work without recognition."

The Community Builder: Opening the Circle

Kaslin did not initially think she would become a community builder. She describes her early attempts: "I had never done anything open source before. After a few failed attempts trying different Special Interest Groups (SIGs), I eventually found my way into SIG ContribX." The turning point came when Paris Pittman asked a simple question: "Paris showed up and said she was starting up a new thing and asked if I considered communications."

That is often how it works in open source communities. Someone opens the circle for you, and suddenly, you find where you belong. Lin thinks about this deliberately in the community meetings she runs. When she sees a new face, she asks them to introduce themselves so they feel welcome. Lin uses a conference metaphor: "It's like at conferences when people are chatting in a circle, and a new person comes by who may not know anybody but is curious about the conversation—perhaps they saw someone on stage and want to be part of it. So you lean and open up space in the circle for that new person to join."

She also applies this to virtual spaces: "When a new person comes in, you open up the circle and give them a chance to introduce themselves and tell others why they are part of the community and what their interests are. This is tremendously beneficial for the new person and for everyone else in the circle, as they can understand the person's intention—are they a contributor, a user, or a potential community advocate?"

They include a shout-out section in the community meetings: "We also give a weekly spotlight to a community user—someone who submitted a merged PR, wrote a blog, spoke at a conference, contributed documentation, or even opened an issue identifying a problem. Having this spotlight can generate a very positive social effect for the project."

Kaslin has taken this work to another level with new contributor orientation: "Last year, we started a new contributor orientation because many folks join community meetings across different teams wanting to grow their skills and do something meaningful. However, it's difficult to understand how large the Kubernetes community is and how it works." The orientation is designed to be

accessible: "Attend a few meetings, understand the information, and if you want to give a presentation, just let us know. We need people to help others get involved. You don't need to be deeply embedded in the community—anyone can join, and experienced members can help answer questions."

But there is a challenge.

When new folks join, excited to get hands-on, Kaslin needs to find a project for them, or they will lose interest: "The challenging part is when they're excited to get hands-on with something. I need to find a project for them, or they'll lose interest."

The Educator: Making It Click

Kaslin discovered something counterintuitive early in her career. When people ask how she stays current, her answer surprises them: "One tip for anyone who wants to stay informed and stay on the cutting edge is that creating your own content is pretty much the only way I've found to really stay on top of things because I've got to be learning to create new content for other folks to learn."

Her path into education started organically. She moved to a team working on experimental prototyping with containers and was involved in the local meetup community: "I started to get involved with the community and leaned more into public speaking, which came naturally to me from my school days."

Kaslin explains what drove her: "In the technology industry, there are so many new things to learn. What sticks is the technology that resonates with you, where you can make progress one step at a time. I wanted to help people find those steps to level themselves up and stay motivated. That's how I dove more into contributor experience and the community aspects of this technology."

Kaslin Fields is currently a Developer Relations Engineer and Developer Advocate at Google Cloud, where she also serves as a manager. She runs the team for cloud runtimes developer relations at Google Cloud, which means Google Kubernetes Engine and Cloud Run, their serverless offering. She is also the co-chair of the Special Interest Group for Contributor Experience in the open-source Kubernetes community and leads its communications

sub-project. She has built her career on explaining the basics of what Kubernetes does and why it has value in the industry.

Kaslin acknowledges the scale of what needs to be explained: "We always say in events, especially during last year's 10th anniversary, that no one can understand the whole of Kubernetes, given how large it is now. The focus in the future has to be keeping it both extensible and flexible for future changes, while not overwhelming it with functionality and technical debt that people can't understand."

Lin approaches education from a different angle. She explains the psychology of learning: "Marketing is critical in the world of tech. People care about two things: themselves and how others perceive them. When educating people on a particular topic, consider how it relates to them and use a metaphor that resonates with their personal experience. The key is to provide value to your audience and not push solutions they don't need. By putting yourself in their shoes and understanding their perspective, you can effectively educate people."

Kaslin is also considering how AI is changing education: "Sometimes it's difficult to talk about AI all the time, but I think it plays a vital role in how we think about education and advocacy in the technology world going forward." She points to kubectl AI: "There's a tool that some Google engineers came up with, which is open source, called kubectl AI. It's a command-line tool that is well-versed in kubectl. It's really nice to have this kubectl AI tool where I can ask, *I know there's a command in kubectl to do this thing. How should I do that?* And it can explain it and run it right there in the console."

Crossing Between Paths

When people assume Lin is juggling everything at once, she is quick to correct them: "I don't believe in multitasking. At a certain time, I'm 100% focused. When I'm traveling and speaking at events, I try to be as focused as possible on those events. I won't participate in other activities, such as joining a weekly meeting, while connecting with the local community."

Lin is realistic about what these roles demand: "If you look at the roles I'm

serving, like the CNCF TOC member, it's a part-time role that demands a few hours per week. The maintainer role is also part-time. I participate in a few communities and don't necessarily appear every week, but I do show up when looking at a monthly or quarterly timeframe. I help people resolve issues, and they feel they can trust me to provide feedback about our project. I help connect them with other maintainers who can value the feedback and potentially resolve project issues."

Lin explains what she has learned about knowing your limits: "One thing I've learned in tech is that you don't want to overwhelm yourself. I know my limits and always try to do my best when I focus on one thing."

Kaslin has a different approach but the same principle. She describes the challenge: "At Google, we have a system of OKRs with measurable team goals. Wherever possible, I try to find a balance between delivering metrics and using opportunities within the community to dive into the technical side of things."

Kaslin seeks natural connections between work and community: "When a product is developing a new feature, it's usually a great opportunity to explore and try out that feature so I can tell people about it. Staying on the cutting edge means continuously learning and trying new things. Right now, our team has a goal for upskilling because this is the era of AI, and we all need to understand how it's disrupting everything."

Recently, she published an AI chatbot application entirely run on GKE: "I used this upskilling opportunity in AI to try out these technologies and help others level up."

Finding Your Path

So how do you figure out where to start?

Lin has given this considerable thought: "There's no one recipe that works for everybody. Ideally, it would be nice to have someone willing to fund your work, giving you more time to contribute to the project. But I understand not everyone has that luxury."

Lin's framework for starting is practical: "First, think about what excites you, and then find one or two projects you would be passionate about. Choose something you would be proud to be part of and potentially speak about in public."

Then comes self-assessment: "Look at your background and find where you would shine. Some people are great at finding bugs and submitting fixes, and can start with first-time labeled issues. Others might excel at improving usability or documentation from a user's perspective, or at following project tutorials and identifying areas for improvement. Another valuable path is being an advocate for the project. This doesn't mean direct code or documentation contributions, but rather writing blogs about the project, speaking at meetups or conferences, and sharing project-related content."

Lin emphasizes the importance of promotion too: "It's important to promote your work—not in a sales-like way, but by sharing your contributions. Attend work group or community meetings, share what you've done, and ask how else you can help. If you're speaking or blogging, consider posting on social media and sharing it with the project. Remember, recognition doesn't happen naturally; you must advocate for yourself."

Lin sums up her advice: "You want to think about what excites you. Second, look at what's available in the market. At the end of the day, it's hard to do meaningful work if you're not getting paid. When you get paid, you have more time, energy, and resources to do something impactful. You want to consider both the market opportunity and what truly excites you. Find something that makes you want to wake up, be at your desk every day, and even speak at conferences. Focus on what genuinely excites you—that's most important. Ideally, find funding for your work because if you're putting in four hours while someone else is putting in 20, it will be challenging to compete."

Kaslin describes how growth happens over time: "I felt like in every meeting with Bob, I would come up with an idea—identifying a problem in the community and proposing a solution. Bob would have all the background on what had been tried in that space before, which would always blow my mind. Now I get to be that person for other people."

Your Combination

There is no blueprint to follow.

Everyone creates their own combination of these roles based on what works for their life, skills, and circumstances. Lin has been through every path at different points. Kaslin has built a career that spans multiple archetypes. Neither of them planned it perfectly. They just kept showing up and figuring out what worked.

Chapter 3: Building Credibility (Without the Map)

When people see someone speaking at KubeCon or running a community project, they assume that person has always known what they were doing. That they studied computer science, that they have been coding since they were twelve, and that they understood containers and orchestration from the very beginning.

Cortney Nickerson has heard this assumption more times than she can count. When people find out she has a degree in political science and philosophy and spent years playing softball in university before entering the tech industry, they want to know how she ended up as head of community at Nirmata, running the Kyverno project, and speaking on stages at KubeCon. The honest answer is not what they expect.

Phil Estes spent fifteen to twenty years working in closed-source software before he ever touched open source. When he finally did, it was not because he had some grand plan to become a principal engineer at AWS or a recognized voice in container technology. His employer suggested that he get involved in Docker because containers were gaining popularity, and he had a background in Linux packaging. That was it. That was the plan.

Prasanth Baskar started with an LFX mentorship because he discovered that you could contribute to open source and get paid for it. Prasanth explains what sparked his interest: "The main spark of my interest was when I started using Linux and its open source. Then I started looking into other source code. I found this vibrant community, and the main reason I'm contributing to CNCF and other things is because I get to go to KubeCons and meet a lot of people."

None of them followed a map because there was not one to follow.

When Nothing Makes Sense

The first time Cortney encountered Kubernetes, it made no sense. She had joined a startup in northern Spain that developed a web application security application. They had customers adopting Kubernetes, which is how she first learned it existed. Then she had the opportunity to join a tech community

focused on Kubernetes, and they needed someone to write a newsletter.

Cortney describes her first six months: "It made no sense at all for probably six months. Every time I think it makes sense now, I discover something else that makes me realize it still doesn't always make sense. It's such a complex environment. At least for me, the first six months were spent trying to understand the difference between a cluster and a pod, a resource, and what that even meant. I was trying to understand what YAML was and figure out all the different components."

People kept explaining it to her in very technical ways. Cortney points out the problem: "I think that's one thing not done especially well in this space: technical documentation for newbies. Everyone always writes from an intermediate standpoint. My initial introduction was quite bumpy and remains so at times. My journey has not been a straight line by any means."

When Phil jumped into the Docker community, he had a different kind of confusion. He knew Linux and packaging, but this was his first time being deeply involved in open source in over a decade. Phil describes what excited him: "What really excited me about the Docker community was the experience of creating a code change, having it reviewed, and being accepted into the project. As simple as it sounds, there was a dopamine effect. Over time, as I contributed small bug fixes and changes, I became more knowledgeable about the codebase."

Prasanth's confusion was about where even to start. He had found a mentorship program and a project called Harbor, but how do you contribute when you don't know the history, the people, or the way things work? Prasanth explains his method: "One thing I do that most people might not do is watch previous meeting recordings. I will go through the meeting notes. If I encounter an issue, I will verify if it has been resolved. If I don't have prior knowledge, I will check if there are any similar closed issues or ongoing discussions. I will review Slack threads, email threads, and previous meetings to address any new issues. This way, you can understand how the community works internally, how people communicate, and how the entire community is orchestrated and keeps going."

The Power of Admitting You Do Not Know

Cortney has developed a playbook for breaking into technical communities without a technical background. Step one surprises people because it goes against every instinct about looking competent and professional.

Cortney's first rule: "Step number one: be vulnerable. Tell people you don't know. The best thing you can do is find an expert and tell them you don't know because they can explain it to you in terms you will understand, and then build from there. Before you ask that expert, do a little background research and tell them what did make sense so that they can see you made an effort."

She defines what effort means: "In this space, things are one hundred percent based on effort. Please make an effort consistently, every day. Share with people the effort you're making and be clear about what you want or need to learn. The people around you will very much help you get there."

Cortney is honest about her own expertise: "I wouldn't say I'm an expert in anything at all, not even in myself most of the time, but I do try to have a broad understanding and grasp of what's happening around me."

Phil has watched this play out from the other side, as someone who is now in a position to help newcomers. He has a favorite story: "My best story is about Sebastian, who showed up in the Docker community and said, *I don't know Go yet. I'm unlikely to contribute code, but I'd love to help organize your issues*. He became an encyclopedia of every open issue, helping link similar issues and providing context. Now he's a Docker employee and containerd reviewer."

Phil explains what Sebastian's story shows: "Communities need all kinds of help, and it's not always about code contributions."

Phil explains how he treats first-time contributors: "When I see on GitHub that it's a first-time contributor—like when someone opens their first issue or pull request—we can sometimes be short in our responses because we're busy. I tend to be overly wordy to ensure they know: *Thanks so much for opening an issue or I'd like to help you*. For example, if they didn't sign their commit, I'll point them to a guide and encourage them to reach out. Being overly

empathetic is key: we're glad they're here and trying to help, even if they didn't follow the guide perfectly."

The Work That Nobody Sees

Prasanth describes how to build credibility: "First, be regular in community meetings. I'll be there in most meetings and start making small contributions and getting involved with Harbor. This slowly builds trust. People will think, 'This is someone I've been seeing for the past three to four months.' They start to recognize you, and then people begin assigning you tasks because they see you regularly and get to know you."

Prasanth's formula: "Be active, be consistent, and be persistent. Don't give up after just a month or a week. Wait for your opportunity to come. You will surely get a chance if you're patient."

Prasanth warns against unrealistic timelines: "One thing everybody does is think of it as a grind, like 30 days of DevOps or 100 days of coding. But coding is not something you complete in 100 days. Coding is a lifetime of learning. It's not like learning Java for 100 days will suddenly enable you to contribute to Jenkins. For example, learning Go in two weeks and then trying to contribute to Kubernetes—a project with 10 years of work—means you can't make a meaningful contribution within a month or two. It's going to take time. People often overestimate and aim to achieve things within a short window, which is the main problem."

When Phil joined AWS four and a half years ago, something unexpected happened. Young engineers at Amazon began sending him messages on Slack, saying they had watched every one of his talks and learned a great deal about containers from him.

Phil describes what built that recognition: "In my previous work, no one ever approached me to say they loved what I do or had watched my talks. It didn't come after just one or two talks or a year or two, but rather from a decade of generating content and speaking at conferences. It took a lot of hard work—boring tasks, working on difficult problems, and consistently crunching away

on maintainer tasks."

Cortney connects her athletic background to building a career: "A good amount of success comes from figuring out your end goal and setting up a schedule for training, studying, reviewing, or reading that is actually attainable and sustainable over time. Smaller goals are better. Slow and steady usually wins the race, not trying to shove everything in at once."

She explains what the athletic mindset teaches: "There's a lot of muscle memory that goes into everything, including tech. Outside of that, it's about making a decision that you're going to do it. Remind yourself daily that you're capable of doing hard things. The more you believe you can do hard things, the more capable you will become. If I can figure out how to set up a kind cluster and put an application in it, anyone can. You decide that you can do hard things, and you will."

But credibility is not just about showing up and doing technical work. Other skills matter, ones that people in tech often do not want to discuss. Skills that feel uncomfortable to admit you need.

The Skills That Actually Matter

Cortney defends sales experience: "Sales is often frowned upon by most people in tech, especially our end user community. That's unfortunate for several reasons. First, salespeople are gritty, rigorous, and disciplined. If they're good at their trade, most sales professionals truly care about helping people."

She explains why sales matters: "None of us can pay our own bills if our sales teams aren't successful, especially when building technology for technologists. We need to understand that technologists won't be as successful without the right tools. They likely won't discover the most advantageous tools without an explanation from someone aware of their general pain points."

Cortney connects sales to platform engineering: "The third point is about platform engineering. We're constantly talking about platform engineers serving their customer—the dev team. But how can you provide good customer service if you don't understand sales? Often, you're selling your

platform, hearing customer needs, and providing a product that addresses those needs. Salespeople do the same with their product teams, reporting when what they're selling doesn't match customer requirements. This leads to product evolution—the same process happens in web applications and services. There are many things people can learn from sales professionals and apply to the world of software and Kubernetes."

Phil has a different version of this same idea.

Phil describes how technical work creates content opportunities. He worked on implementing user namespaces in Docker, a challenging technical task that took considerable time. Then he had something to say: "Then I wanted to tell everyone about user namespaces and why you should use them. I started submitting talks to conferences and getting some accepted."

Phil explains the key: "To me, the awesome thing about having deep technical work under your belt is that your talk can be more relatable to developers and engineers. I'm not just pitching something my company is doing; these are things I actually did. I want to share what I learned, how I hit a roadblock, and overcame it with a solution."

Prasanth has learned this lesson from a different angle.

When Prasanth gave his first KubeCon talk, it was a Project Lightning talk representing Harbor. The talk was scheduled for 30 minutes, but he completed his presentation in 15.

Prasanth describes what happened: "The remaining 15 minutes were dedicated to discussion, with people geeking out and showing genuine interest. I sparked something within the audience—they saw solutions to things they had been wondering about in the past. The interaction was so engaging that it motivated me to continue presenting, to get more people excited about Harbor, Kubernetes, and the entire ecosystem."

Prasanth explains what matters in presentations: "Preparing the slides would be the time-consuming part because speaking is easier. Discussing conversations is better. However, the slides must be accurate and technically correct. People often revisit slides if they find the talk useful, so you need to ensure quality. Don't just generate slides with AI-generated content and showcase them. Make sure the slides are revisitable, allowing the audience who attended the talk to revisit and take notes, continuing to learn from the material."

You can do everything right, show up, contribute, stay consistent, and build something real. But somewhere along the way, you will find that the most

critical moments came from people who saw something in you before you saw it yourself, and chose to open a door.

The People Who Open Doors

Every person who succeeds in this ecosystem can point to specific individuals who opened doors for them. Cortney credits James Spurin, Saiyam Pathak, and Brad McCoy—who signed her up for her first talk at KCD Pakistan without asking. Phil learned from Michael Crosby at Docker, whose quiet guidance launched his open source career. Prasanth's mentor, Vadim, taught him that "people matter more" and that relationship building is fundamental.

The pattern is the same: someone believed in them, opened a door, and pointed them in the right direction. But what happens after that door opens? How do you go from being the person someone vouched for to being the person others vouch for?

From Invisible to Visible

There is a shift that happens somewhere along the way. You go from being the person nobody knows to being the person people recognize. However, that shift does not occur in a single moment or accomplishment. It happens through accumulation.

Prasanth is now a Cloud Native Ambassador. He explains formal versus informal credibility: "Formal credibility puts you in the spotlight. Everyone will approach you. If you are a Cloud Native ambassador presenting at KubeCon, people will find your badge and approach you to talk. They want to learn more about your achievements and journey. Informal recognition and credibility are necessary to obtain formal recognition. These are two sides of the same coin."

Prasanth describes how to build informal credibility: "Show what you do by putting it out there so people can see your work, rather than constraining it within yourself or keeping it in a private repository. Write a blog, make a video, and create posts to let people know what you're doing."

Prasanth explains his approach to social media: "I share processes, such as a release done. I use social media more as an announcement channel rather than sharing all my work. Instead of posting *Come, everybody. See,* a good social media post would be: *I have tackled this entire problem. Here is how I did it.* To create a good online presence, background tasks require research and substantial work to back up the post."

Cortney explains what accelerated her career: "The biggest activity that's accelerated my career is 100% networking. This means networking on various levels, not just meeting someone to name-drop them. It's about saying, *I see you're involved in this. I'm very interested in the specific part of what you do. Can you share some of your expertise?* networking means being willing to approach someone at a conference and say, *Your talk was great. Can you tell me more?* It involves following people on LinkedIn, not by sending connection requests to strangers, but by engaging with their content. If something interesting appears in your feed, engage by leaving comments and asking questions. I believe the more proactive you are about networking—getting to know people and letting them get to know you—is probably the fastest way to move yourself forward. This approach helps you understand what you need to

learn and where the industry is heading, ensuring you're in the right place at the right time."

Cortney describes the difference between followers and respect: "Viktor works insanely hard at understanding technologies, the community he's working with, and how things are built by different people in different places. He forms his own solid opinion based on that and shares it with the world. He does not care if people agree or disagree and welcomes both. Being able to say, *My knowledge level might be a three on a scale of 30, or it might be a 35 on the scale of 30. Still, I have a solid opinion about what I do know* which makes a world of difference in terms of people respecting you as a person, not just following you because you took a nice picture."

But perspective, respect, and credibility —none of it arrives overnight.

The Long View

Phil reflects on what built his reputation: "It was a cool feeling that all of that added up to a reputation of helping people learn about containers and cloud native technologies. I never set out with the goal of having people walk up and say they learned something from me, but it's a pretty amazing outcome."

Cortney spent three or four years trying different explanations until she found one her family understood: Kubernetes is like a head chef in a restaurant. Your cluster is the kitchen; Kubernetes is the chef, ensuring everything runs smoothly, that the recipes are followed, the meals are served on time, and the customers are taken care of.

Cortney explains what moves the needle: "The level of impact of your contributions very much depends on your end goal. I've taken a track of being involved in the community, giving back time by serving as a track chair or reviewing CFPs to learn more and provide a good track schedule based on people's interests."

She warns about certifications: "Certifications are great, especially for understanding global concepts. However, there is a tendency to think that a certification doesn't actually make you capable of doing things. Any certification you complete or are working on must be accompanied by a project where you can put those skills into practice, so people can see that you not only understand the concept but can also apply it."

Cortney's principle: "No matter what route you take in combining different types of contributions, the most important thing is to ensure that whatever you're learning or doing can be applied to a specific project. Always ask yourself: Why am I doing this? How does this apply to the community space or the job I'm interested in? By always thinking about your next step, you'll eventually build the castle you want."

Even when you are clear on your purpose, the path forward rarely takes the expected course. The things that actually move you forward, the ones that open up new chapters, often start with something far simpler: a real conversation, a shared question, a small moment of connection.

What Actually Works

Cortney describes how her most meaningful opportunities came from genuine friendships: "I've been very fortunate to make friends on Slack by reaching out with a curious question. After a week of back and forth, we'd often say, *Maybe we can have a call. It would be faster for you to explain it to me in five minutes.* I've made some very random friendships that way, as well as meeting people at different conferences and events and actually getting to know them as human beings, not just by their day job."

Her keynote at KubeCon China was made possible through friendship. She had been friends with Amit DeSouza for a couple of years: "He said, *Hey, the deadline for KubeCon China is tomorrow. You want to submit something?* I said, *Sure!* We jumped on a call that was really to catch up, came up with a submission off the top of our heads, and ended up enjoying a week in Hong Kong to keynote."

They ended up speaking about Crossplane, but not in the usual way. Everyone talks about Crossplane as if it is the solution to everything, but no one stops to ask what problem it is actually solving. That became their angle: "Crossplane is the answer. What's the question?"

Phil's advice for people starting: "I have always told people who want to get involved in communities: go find where that community hangs out and meet who's involved. In our virtual world, this could be Slack, some older open-source projects' email groups, or platforms where you can view community interactions. I find that early contributors succeed faster when they get involved in the community rather than just submitting their first contribution without any prior interaction. It's hard to understand their purpose when no one knows them."

Prasanth explains what helped him build credibility: "Then you start helping others by being consistent. Once you start helping others, you also gain many valuable insights. So, share and help others."

He is now an LFX mentor himself: "I mentor people in a three-month term and help them understand open source contributions, including how to contribute, manage projects, and participate in community meetings."

It is a cycle that keeps going. Someone sees something in you before you see it in yourself. Later, you pass that belief on to someone else. That is how the community grows, quietly, steadily, one connection at a time.

And eventually, all the scattered pieces begin to align, but no formula works for everyone.

Building Your Credibility

Cortney studied political science and played softball. Phil spent twenty years working in closed-source software. Prasanth got his start through a mentorship program. They each built credibility, but they took different paths to achieve it. What ties their stories together is not where they came from or which strategy they used. It is that they were honest about what they did not know, they kept showing up even when it felt like nothing was happening, they helped others before anyone asked them to, and they understood that credibility does not come quickly.

When you see someone speaking at KubeCon or listed as a maintainer on a significant project, you are seeing the result of all the silent work that came before.

You don't see the first six months, when nothing makes sense, or the hours spent watching meeting recordings, just trying to understand how things work. You do not see the early contributions that go unnoticed, the Slack messages that answer someone's question without fanfare, the blog posts barely anyone reads, or the talks delivered to a half-empty room.

But all of that matters.

It builds trust slowly, almost invisibly, until one day, people see you as credible without even realizing when that shift occurred.

Cortney advises people seeking to package their work: "They're used to packaging things to make sure the best parts stand out. Find somebody who manages a team of SREs and ask them for a coffee chat. Interview them about the skills that stick out, what they're looking for, and how somebody demonstrates these things. The more you can interview them and understand

what they're looking for, the easier it will be to package your capacities to find a good fit—not just for you, but also for the people who are hiring you."

Phil's advice: "Technical work and sharing can go hand in hand, but there's no magic bullet. It takes extra effort to think about how this work will turn into talks, blog posts, and other content."

Prasanth keeps it simple: Wait for your opportunity. Be patient. It will come.

In a space that moves quickly, credibility builds slowly, and that is the nature of it.

Cortney's final message captures everything: You decide you can do hard things, and you will.

That confused person at their first KubeCon, standing in the hallway trying to figure out where they fit, wondering if they belong because they don't have a computer science degree or ten years of experience, is exactly where Cortney was, where Phil was, and where Prasanth was.

The people who appear to belong all started just as lost. They just kept walking anyway.

Chapter 4: Networks, Learning, and Growth

When you talk to people who've made it in the Kubernetes world, you expect to hear about their technical breakthroughs. The moment they finally understood container orchestration, the first time they successfully deployed something complex, the bug they fixed that changed everything.

But that's not what Emily Long and Yasmin tell you.

Emily, now CEO of Edera, will tell you about sitting across from James Petersen at Anchor while he wrote Professor James on a whiteboard behind him. Yasmin, COO at CloudBolt Software, will tell you about watching Justin Garrison hit a bucket of water in a YouTube video.

The Accidental Journey

Emily never planned to end up in the cloud native world. She'd spent over a decade at KPMG, one of those accounting firms where everything is spreadsheets and audit reports. Then, in 2017, someone recruited her into the tech industry to work in human resources. Emily describes her accidental entry: "It was honestly by accident. I began my career in finance and accounting at KPMG, one of the Big Four accounting firms, over a decade ago. I was recruited into tech to do human resources in 2017." But in 2019, something shifted. She took a job at Anchor as their Chief People Officer, and suddenly, she was surrounded by containers, Kubernetes, and a host of unfamiliar terms that meant nothing to her. Then the company offered her the COO role, which involved managing technical teams, including professional services and technical support. Emily admits: "I didn't even know what a container was when I started working at Anchor because my job was in human resources operations, supporting people around me."

Emily wanted to understand what her teams were actually doing, so she asked for help. That's when James Petersen entered the picture.

James had this way of explaining things that didn't make you feel stupid for not knowing them already. Emily describes their whiteboard sessions: "He did that with me, quite literally having a whiteboard behind him where he wrote Professor James and started explaining: *This is a container, this is what an*

orchestrator does, this is Amazon, and why—what is EKS, what's EC2, what's a control plane? We went through the whole container lifecycle together."

Emily explains what that mentorship meant: "Over the years and different companies, James continues to be instrumental in taking a step back and teaching, which has been incredibly important to me—being able to be helpful for those around me and be a good sounding board for people working directly with Kubernetes day to day."

They've now worked together at three companies: Anchor, then Chainguard, and now Edera. That's how it works in this ecosystem; you find people who get you, who teach you without judgment, and you stick together.

Yasmin's entry into Kubernetes was different but equally unplanned.

Yasmin moved from working with traditional virtual machines to exploring Kubernetes. She explains how it happened: "I was at Puppet for a while, and someone reached out from a company called StormForge that was doing Kubernetes optimization. Kubernetes has always been a space I wanted to explore, transitioning from the traditional VM world. I happened to luck out that I liked the people and wanted to learn more about the technology."

Learning in Public

When Yasmin started at StormForge around 2022, she was overwhelmed by Kubernetes concepts. HPA, VPA, autoscaling, pods, namespaces—everything was abstract and confusing. Then she found something that changed everything: "I was introduced to Kubernetes concepts through Justin Garrison's Containers on the Couch shorts. The one with the HPA and VPA, hitting the bucket of water, really helped me visualize the concepts. It made learning the technical side much easier. People joke that short-form visual formats are taking over, but the reality is they did make the concepts more accessible."

Years later, when Yasmin finally met Justin at KubeCon, she had one of those moments that made you realize how small this community is: "I even met Justin at a KubeCon and was a bit starstruck, telling him, *Thank you. You taught me everything I know about Kubernetes.*"

Emily has her own version of this story about how the community teaches. At Edera, they have team members deeply involved in the Kubernetes community: "My co-founders grew up in open source, with the Dart programming language, Alpine Linux, and similar ecosystems. We have many team members who are well plugged into the Kubernetes community. A lot of what I do is learning through them. I go to every KubeCon, and Cloud Native Rejekts is one of my favorite conferences where I can learn from people using and interacting in the community."

Cloud Native Rejekts is a conference built around talks that didn't make it into KubeCon. The sessions that never reach the main stage are often the ones with the most honesty, the most lessons learned, and the ideas that actually move the industry forward.

The Power of Not Knowing

Emily explains what Kubernetes's complexity creates: "The benefits of Kubernetes being so complex—with its different layers of storage, networking, databases, and observability—mean that you can't really be an expert in everything. There's a welcoming nature that allows you to ask what might seem like a 'stupid question' (though I don't believe such questions exist), and you're met with an eager willingness to teach and help. I don't think I've ever learned so much about something so complex so quickly anywhere else. At KubeCon, even if you're lost halfway through a talk, people will walk you through it and are eager to do so. It's taught me a lot about humility, openness, and learning new things. To me, it's so much bigger than just technology—it's shaped the way I see learning in tech."

This curiosity about learning and collaboration also shapes how Yasmin thinks about people—especially when she's the one doing the hiring.

When Yasmin discusses what she looks for when hiring, she focuses on problem-solving over expertise: "Everybody in an interview process wants to prove their technological knowledge and how good they are with Kubernetes. However, what I focus on is how they approach problems and how creative they become. Within the Kubernetes landscape, there's something for every problem, and you can stitch together pieces. It's less about how well you know one piece of technology and more about how you think creatively to bring all those pieces together."

No one enters the Kubernetes world as an expert. Yasmin and Emily began with uncertainty—Emily coming from HR and Yasmin from virtual machines—and each found mentors and communities that made learning possible. Curiosity, openness, and the willingness to ask questions are the foundation for real progress.

Networks That Matter

Emily emphasizes the power of networks: "I've gotten most of my jobs through my network. I want to shout out to James Peterson—this is our third company together."

Yasmin has observed this pattern from both sides of the vendor-end user divide. She started as an end-user around 2016, deploying her first Docker container when Docker was still in its early stages, and the site would often go down. Yasmin describes what she learned: "Fast forward over 10 years, and it's amazing how far we've come. Being an end-user helped me understand the mindset when evaluating technology to solve a problem—the whole build versus buy approach. Being on the vendor side is cool because you get to see that lens across every industry. This career transition helps people empathize with those on the other end."

She shares something that surprises people: "At Staples, we sold office supplies, yet the team internally built an entire platform using Go in the early days. We even hosted a Go meetup around 2016-2017. You don't typically think of a retail company selling office supplies as being involved in the tech community. But it is possible. End-users often drive innovation by pushing vendors to meet their needs."

Innovation happens everywhere, networks form across traditional boundaries, and the person you meet at a meetup today might be your co-founder, colleague, or mentor tomorrow. These early conversations quietly shape the opportunities that come years later.

The Long Game

Emily describes the current Kubernetes job market: "Over the last decade, I remember job descriptions requesting X years of Kubernetes experience before Kubernetes was even that established. Now that it's been a decade, there's a common understanding that anyone entering a cloud-native role or working with containers should have a background or familiarity with Kubernetes, specifically. The job descriptions I'm seeing now often include AI components, which I find strange since they're not always directly related to the work. However, I think the evolution of understanding backend infrastructure is becoming more important and complex. The ability to comprehend deeper levels of the technology stack is becoming increasingly valuable."

Yasmin sees the next wave of opportunity in FinOps Kubernetes engineers: "The other area I'm particularly interested in is FinOps Kubernetes engineers. These are engineers who handle the platform's day-to-day operations and are increasingly being asked to provide insights into platform spending. Kubernetes has many shared costs, and many teams struggle with chargeback mechanisms. Developers often view resources as free because they aren't directly paying for them, but these are actual business expenses. Because of Kubernetes' complexity, a typical FinOps person in the finance team can't easily break down costs. You need someone who truly understands the technology and platform."

Success in Kubernetes comes from people, not just technology. Growth happens through community, mentorship, and the willingness to learn and help others. The most valuable people are translators, serving as bridges between the technical and business worlds, those who can explain why something matters in terms that different audiences understand.

However, with all the shifting roles and changing expectations, how do you even get your foot in the door?

Finding Your Way In

For individuals seeking to enter the Kubernetes world from non-technical backgrounds, both Emily and Yasmin know what helps. Emily addresses the mindset: "It's hard. Part of the journey is knowing it will be challenging. Sometimes, acknowledging the difficulty helps because you don't feel like you're constantly failing. Instead, you understand it's an uphill battle, and you continue to chip away at it."

Yasmin offers a practical roadmap: "Start with the basics of core infrastructure. Initially, forget Kubernetes and focus on understanding how to provision infrastructure, such as basic EC2 instances. Learn what cloud means and get comfortable with those concepts."

Emily sees remote work as an environment that facilitates learning: "I'm a huge proponent of remote work. It has opened opportunities for people to learn and explore new career paths more than ever before." But remote work also requires something specific: "Remote work requires understanding that it is your responsibility to participate in the community. There are challenges when you can't simply stop by someone's desk for a conversation. Living in digital ecosystems like Slack means you must be diligent about problem-solving and interaction."

Breaking into Kubernetes is about patience, persistence, and steady learning. Emily and Yasmin reveal that progress occurs when you acknowledge the difficulty, establish a solid foundation, and take ownership of your own ability, particularly in remote and distributed environments.

This sense of responsibility and self-direction naturally leads to the next theme—trust. In a community built on openness and collaboration, trust becomes the real currency.

The Currency of Trust

Emily explains her philosophy on compensation: "We do a global pay band for the most part to ensure we're not doing location-specific compensation. We believe it's easier and fairer to say that everyone at this level receives the same amount. My advice is to be very open about understanding your opportunities and weighing what you're going into a job for. Sometimes it's for base compensation, and sometimes it's for the upside, such as stock options. You'll see different compensation variables. Being unafraid to ask questions upfront about how salary is determined is really important. Most companies, even smaller ones, could have an answer for you."

Yasmin explains the role of certifications: "CKA is definitely very important, especially because as we work with customers or prospects, that credibility is crucial. We want to ensure that when we claim to know something, we actually have proof to back it up. To me, it's less important whether you have a CKA when you join—it's more about whether you're excited about getting one and have the aptitude to take that path. You're going to learn along the way. I know very seasoned Kubernetes experts who failed their first CKA. It's not about passing on the first try, but understanding the pieces to reach that certification."

A certification shows you're committed to learning, willing to do the hard work, and capable of staying with something complex until it becomes clear. But what sustains the community long term?

The Community's Future

As Kubernetes enters its second decade, both Emily and Yasmin see challenges ahead. Yasmin describes the organizational challenge: "The community needs more organization. It's challenging to have a community that is both geographically dispersed and diverse in terms of technologies. Different individuals have various initiatives—from observability to infrastructure provisioning—all working towards the same goal. This complexity is evident in our events, and we've been participating in numerous CNCF and Kubernetes events. Organization is probably the biggest challenge. Prospects often ask about upcoming Kubernetes features, but it's difficult to provide insights because features can remain in alpha for extended periods before switching to beta. When you have many people working on a project that isn't vendor-backed, coordinating efforts becomes complicated."

Emily sees the community's openness as its greatest strength: "This community is wonderful, but breaking in involves networking more than anything else."

The real challenge is keeping that door open as technology becomes more complex and the stakes increase. What makes the community special is its willingness to teach, share, admit when you don't know something, and help others find their way.

Your Turn

The stories of Emily and Yasmin show that there's no single path into Kubernetes. Emily came from different disciplines, yet both became leaders in cloud-native.

Their journeys remind us that Kubernetes is also about orchestrating people, not just infrastructure. It's about honesty when you don't know, generosity when you finally do, and relationships that outlast companies and titles. Tools and frameworks change; what stays are the people who answer your late-night questions, remember your first talk, and bring you along to the next job.

The community's true strength lies in its people. You don't need a technical background to enter; you need curiosity, a willingness to ask, and the drive to build connections.

Chapter 5: Sustaining the Journey

There's a moment that comes after the excitement fades. You've figured out your path, made connections, given a talk, or shipped something people use. The community that felt impossibly large now feels familiar. You know where things are, who to ask, and how to contribute.

And then one day, you realize you've been doing this for three, five, or ten years.

The question stops being "how do I get in?" and becomes "how do I stay?"

Jorge Castro has been in cloud native for a decade: "I've been around for 10 years already, which I didn't realize." He started at Canonical, attended early SIG cluster lifecycle meetings, and eventually connected with Craig McLuckie and Joe Beda at Heptio, where he served as their community manager. Years passed. He worked on contributor experience, moved to Cloud Custodian, and then to Kubeflow. Now, he's at the CNCF doing "a bit of everything" on the projects team.

Whitney Lee's path was different but equally deliberate. She spent over ten years as a wedding photographer: "I really hated my wedding photography job by the time I was done." At what many would consider too late to start over, she had something like a third-life crisis. She spent all her savings returning couples their wedding deposits, dissolved her business, broke up with her partner of eight years, and put everything in storage to tour as a musician for a year.

When she returned, she worked in restaurants for a few years before her son, who was studying computer science, suggested she try coding. In January 2019, she wrote her first line of code. By November 2019, she had a job as a cloud developer at IBM. Less than two years later, she was keynoting at KubeCon. Now, she's a senior technical advocate at Datadog.

Jorge describes his current role: "My job title doesn't really make sense anymore. I have a new one, but I'm not sure what it is yet."

That's what ten years looks like. Your role evolves faster than the titles can keep up.

Both Jorge and Whitney have figured out how to stay in the game, but the strategies that got them here aren't the ones keeping them here. Sustainability requires different skills than entry does.

You Can't Know Everything

The thing nobody tells you when you're starting is that the feeling of being overwhelmed never completely goes away. It just changes shape. Initially, you're overwhelmed because you know nothing. You assume that once you learn enough, that feeling will fade. Then years pass, and you realize the overwhelm didn't disappear—it evolved. Now you're overwhelmed by how much has changed, how many new projects exist, how many technologies you've forgotten.

Whitney openly admits she doesn't try to master everything permanently: "Here's the big secret: I don't try to know everything. I accept it's not possible. Whatever I'm focused on right now, whatever resource I'm making, I learn everything about that. Some of my older knowledge falls away. But when I need it again, when it comes back up, I'll find what's current about it at that time. I think it's a fallacy to believe any one person can stay on top of everything. Sometimes when you try to, you lose focus on being able to go deep on one thing."

Jorge uses a different strategy: "I'm using Notebook LM where I put the SIG's last few talks into AI and say, *Summarize the last five important things that Argo is going through*, for example. I'm deferring a lot to AI tools, which is working out pretty well."

Jorge explains his focus shift: "I mostly focus on the people. By the time we're discussing things, I no longer read the change logs for Kubernetes. It takes on a more macro perspective. It becomes more about the greater context as opposed to the little details. I hope that makes sense. I have no idea what's going on in Kubernetes."

A decade in, with all that experience, Jorge still feels like he's trying to keep up. The difference is he's made peace with it.

When Jorge explains Kubernetes, he doesn't talk about pods or containers:

"I'm not going to say it's just five components. The way I explain it is distributed computing. It's about how we scale to solve problems. On the software side, that's Kubernetes. On the community side, it's also Kubernetes." He considers the project an ant colony rather than a pack of wolves with a rigid hierarchy.

Building Your System

What distinguishes people who last from those who burn out isn't superhuman capacity—it's having deliberate systems that work for them over the long term.

Whitney's approach started with a practical problem. She was transitioning from carefully rehearsed IBM Cloud Lightboard videos to live streaming, and she was terrified: "The question and answer part was the scariest because it wasn't practiced, and I didn't know what people would ask."

Her solution was systematic. She used Anki, a spaced repetition learning tool, to make flashcards for the entire Kubernetes glossary: "I didn't want any Kubernetes concept to catch me off guard." She created flashcards for how containers work, DNS, hacker rules from a GitHub repository, and the 12 Factor app: "At the time, I didn't know how these would be useful—I was just curious. But later, I discovered they were relevant to more things than I initially thought."

Whitney explains what the flashcards were really about: "I never pretend to know something that I don't know, and I'm quick to admit when I'm wrong." However, having the foundation meant she could learn out loud without constantly getting lost in jargon.

Her long-format show became a vehicle for building relationships on a large scale. Through two-hour live streams where she learns vulnerably from experts, she's met maintainers of 50 or 100 different CNCF projects: "When opportunities arise, people know who I am. I've repeatedly proven that I've been vulnerable, kind, and delivered on my promises."

Whitney's advice: "I make content I'm curious about. Don't worry about what will get the most views. If you find value in it, chances are someone else will too. You learned something from creating it, even if no one watches it. If you're obsessed with boosting your reputation, people watching your content see that, and it doesn't feel genuine."

Jorge has learned to be ruthlessly efficient: "I'm very much in efficiency mode lately—focusing on the least amount of things we can do to get people moving."

He's stopped waiting for perfect consensus: "Sometimes you want and feel like you need consensus so badly that you'll sit on something for too long. Our team is finding that sometimes it's better not to ask for permissions: go for it, and you'll get consensus as you're doing it."

At conferences, Jorge warns about energy management: "The first day can drain your energy for the entire week if you're not planning carefully. I try to manage audience energy, bounce ideas off coworkers, and take care of myself by taking breaks, drinking water, and wearing comfortable shoes. I recommend getting to know the people at neighboring booths. We're all going through similar experiences. Connecting with others helps take the stress off."

Jorge advises newcomers to set clear goals before attending conferences to maximize learning and professional value: "Try to set specific goals for yourself. For example, aim to meet at least three people in leadership positions or attend three talks about topics you're interested in. Consider what's important for your work and what value you can bring back to your organization. If your work paid for you to attend, start building a professional network. Disseminate the information you learn to your coworkers to demonstrate your value and expertise. For instance, share insights like booth conversations about project roadmaps or reference architectures from end-user sections. Don't just wander around aimlessly—you could spend an entire week learning nothing without a strategic approach. What you don't want to do is go home with homework. You want to go home having learned about three important things. This gives you something to do between KubeCons."

Jorge recommends shadow programs: "Consider finding a mentor or joining a shadow program. I shadowed the release team for a considerable amount of time. Sometimes, simply listening to meetings can help you become more effective in your job. Look for positions with lower deliverable pressure where you can observe and learn the subtle things no one talks about."

The systems Jorge and Whitney have built are about being human. They're about being strategic—knowing your limits, playing to your strengths, building knowledge deliberately, and showing up in ways you can sustain.

When Your Role Changes

Jorge describes how roles evolve: "In the early days of your career, you were doing things like managing GitHub permissions for the project. A few years later, you're discussing macro things like funding the maintainer summit for everybody. We all cycled out, but stayed in cloud-native. Now three of us are on the project team. One was on the governing board. Christoph, I think, is the chair of the governing board now."

Jorge explains what happens when you stick around: "It's honestly really challenging for me because I have to stay relevant and remain useful to the community. I'm always looking for a new angle or perspective. By the time you've been around so long, it's about introductions and personal relationships. I'll find an expert who knows how to do something and connect people."

This transition can feel uncomfortable. You started because you loved the technology. Somewhere along the way, your job became more about people than code.

Jorge has made peace with it by focusing on efficiency and recognizing that his role provides a different value. Your role change doesn't mean you're doing something wrong. It means you're accumulating knowledge that only comes from staying around long enough to watch generations cycle through.

Jorge describes what new people bring: "Seeing new people come in doing our old jobs is great because they're not the same jobs we had. They're usually more efficient and have way more energy. Many things are done differently these days, with a greater focus on sustainability and maintaining a long-term vision. In the early days, we were growing so fast, trying to keep the ship together."

He points to Kaslin Fields as a classic example: "When she started, she was a Kubernetes ambassador doing small things. Eventually, she got more involved in Kubernetes. When my generation cycled out, her generation came in, and she became a sig lead. While I was away, I would walk by the Kubernetes contributor summit and see that they were doing similar things we had done before—maybe with different outfits and themes. I would check in and find they would build on previous work, trying to get more shadows involved, so they eventually become contributors. The other day, I saw her on a keynote for

Google Cloud in front of millions of people, and I thought, 'I remember when we started.'"

Jorge measures people by whether they carry the community with them after leaving: "Sometimes I've had really good friends I thought would be in cloud native forever, and then they leave—or might come back. I left for a while and returned. I focus on whether they carry the community with them when they leave. Do they think fondly of us when they've moved on in their career?"

The marker of success isn't whether someone stays for a decade. It's whether their time was meaningful enough that they carry something with them.

The Human Side: Small Contributions Matter

One of the most damaging myths in tech is that only big, visible contributions count. That you need to be Kelsey Hightower or have commits to core Kubernetes to matter.

Jorge fights against the myth that only immense contributions matter: "Many get wrapped up thinking, 'I'm not Kelsey, so I must not be helping.' But even small things matter—like helping someone find a session at KubeCon when the map is complicated." He tells a story: "Maybe they helped in a tiny way, like moving coffee tables when there weren't enough staff. I told someone, *If it weren't for you, we would have had to delay by 15 minutes.* There's a cascading effect most people don't consider."

Jorge on community culture: "Many people in our community will move on. The number of people who stick around for a decade is small compared to those who attend just one event. I went to a KCD in Hungary this year, and it felt just like KubeCon—same culture, even though I'd never been to a KubeCon there."

Whitney learned a similar lesson through a book: "There's a book I love called *Range: How Generalists Triumph in a Specialized World*. That really helped cure my imposter syndrome and made me feel like all my experience—from restaurants to serving as a musician, wedding photography—is relevant and has

helped me become who I am today."

Whitney describes her mindset shift: "I was raised to think that people doing the same thing I am doing are my competition—that I need to be the best. That mindset does not serve you well in this situation. Many smart people surround you, and it's great, amazing, and a gift. If you reframe it as a gift and not as 'I don't belong,' it's powerful to see everyone around you as your community—people you lift and who lift you, not as your competition."

Whitney recommends starting with collaboration: "If you're learning about a CNCF project, maybe see if someone from that project wants to get involved and help you make a resource together. Many projects often claim to need contributors. One way to contribute is by creating content for that project. If you reach out to a project like Argo CD on CNCF Slack, stating that you want to create a blog post or content, they may be receptive. While ArgoCD might be a challenging example due to its size, sandbox projects especially will welcome you with open arms. You'll have someone to collaborate with, someone to review your content to ensure accuracy, and a supportive team around you."

The Pressure and the Doubt

Jorge admits facing the dual pressure of staying friendly while delivering on technical obligations: "All of us sitting there being friendly and trying to help are also under pressure to deliver. That's the consequence of running everything. We have to take the project seriously. We must move forward, ensure technical debt is addressed, and make the project sustainable. But we can't have maintainers crashing and burning."

Jorge thinks about creating "a productive bubble where you can relax." He names role models: "Tim Hockin never panics. Tim St. Clair never panics. They're even-keeled but passionate, showing that tackling hard problems quickly leads to a longer, happier path." He asks in meetings: "Is somebody going to suffer in nine months because of this decision?"

Jorge admits his own imposter syndrome: "Imposter syndrome is real. I also

have no idea what I'm doing. I feel intimidated when I go to the summit and hear people talking—I have no idea what they mean, even if they're using the same words. Don't sweat that. We need to organize meetings, regardless of whether you understand the technology or not. Running a well-organized meeting is a skill. Sometimes I go to a Kubernetes meeting that's very technical but disorganized—that's where someone with project management skills might excel. By the time you get here, someone will have seen something in you. Maybe you brought energy to a local meetup, or someone recognized your potential. You're there for a reason, even if you don't know exactly where you'll fit."

Whitney faces similar doubts. Her keynote took place less than two years after she began coding. Her co-speaker, Mauricio Salatino, "had to deal with all of my imposter syndrome, forcing him to slow down and not use jargon."

Whitney learned to get comfortable with public ignorance: "I eventually learned I could simply say *I don't know*. Even better, I started saying, *I don't know the answer right now, but if you contact me on LinkedIn, I will find the answer for you. I know who to ask*. As long as I feel genuine, I can be big, loud, and proud of who I am. But if I were to tell a lie or try to cover up a gap in my knowledge, it would make me feel scared."

Whitney notes that imposter syndrome doesn't disappear with seniority: "The higher your position, the worse imposter syndrome can get because you're expected to know even more. The idea of staying on top of everything in Kubernetes is impossible."

Even the creators of Kubernetes wouldn't claim to know everything about it now. The goal isn't to finally learn enough. The goal is to get comfortable with not knowing while still being useful.

Staying In for the Long Run

The ecosystem moves fast enough that standing still feels like falling behind. Part of sustainability is learning to work with that pressure instead of against it and becoming comfortable with imposter syndrome, rather than waiting for it to disappear.

The people who've been around for five, ten years aren't the loudest voices. They're not producing the most content or speaking at the most conferences. What they have in common: they figured out how to show up in ways that work for them over the long term.

Jorge focuses on people and connections rather than trying to keep up with every technical detail. When asked what his North Star question would be for newcomers, he says: "Keep going. Follow the crowd, though it seems counterintuitive."

Whitney built a sustainable practice around learning out loud and genuine curiosity, rather than metrics-driven approaches. Her teaching philosophy centers on clarity: "Don't use acronyms, or if you do, spell them out, even for terms like SOC. SOPS stands for Secrets Operations—though no one ever calls it that. As a detailed and literal thinker, if someone says SOPS and I don't know what they mean, my listening stops, and I start wondering about the acronym."

Neither is trying to be everything to everyone. Whitney and Jorge found what they're good at, what energizes them, what they can sustain.

The community needs people who last, not just people who sprint. It requires the institutional memory of people like Jorge, who can say, "We tried that before," and connect the right people to move things forward. It needs the authentic learning of people like Whitney, who model that it's okay not to know things and that diverse backgrounds bring valuable perspectives.

That person at their third KubeCon who's starting to feel burned out, wondering if they can keep up this pace, questioning whether they still belong—they're at a critical juncture. This is where people either figure out how to make it sustainable or leave.

The ones who stay aren't superheroes. They're people who have learned to say no, who focus on what actually matters, who build networks of support, and

who permit themselves not to know everything.

This work is a marathon. You need to pace yourself accordingly. The community needs people in year five and year ten who provide context and continuity, who remember what was tried before, and who mentor the next generation.

The path from newcomer to sustained contributor isn't about perfection or knowing everything; it's about learning and growth. It's about finding your angle. It's about following your curiosity. It's about showing up consistently in a way that doesn't overwhelm or exhaust you.

You don't have to be the best, most productive, or most visible. You have to find a way to keep going that doesn't require you to be someone you're not. That's the sustainable contributor: someone who figured out how to stay in the game by being themselves, consistently, over time.

Contributors



Bob Killen

Senior Technical Program Manager

How they keep up to date: Bob stays current through mailing lists, SIG meetings, community channels, and conferences. He tracks each release enhancement tracking sheet and monitors K-dev.

Favourite books:

Favourite repos: [kubernetes/contribute-site](#)

SIG/WG participation: contribex, steering, architecture



Cortney Nickerson

Head of Community @ Nirmata

How they keep up to date: Cortney follows KubeFM, Cloud Native Podcast, Cloud Therapist, and her own PolicyBytes podcast. She watches Thunder, Kubesimplify, and AI & DevOps Toolkit on YouTube.

Favourite books: Decoding Platform Patterns by Shweta Vohra and The Art of Community by Jono Bacon

Favourite repos: Kyverno and Keda

SIG/WG participation: SIG-foodies, Deep Roots, Women in Cloud Native (WG-Merge Forward under SIG-Contributor Strategy)



Emily Long

CEO

How they keep up to date: Emily watches edera-dev YouTube channel and colleagues' content recommended by Lewis including wiggitywhitney, DevOps Toolkit, and Rawkode Academy. She attends KubeCon talks.

Favourite books: Hacking Kubernetes (recommended by Lewis) and Diary of a CEO by Steven Bartlett

Favourite repos: AstroNvim (Lewis's favorite for learning vim)

SIG/WG participation:



Jorge Castro

Developer Relations @ CNCF

How they keep up to date: Jorge listens to FAFO and Fall of Civilizations podcasts. He watches smaller K8s channels without a Kubernetes presence, Linux content from Chris Titus Tech, and ecology content from Lindsay Nikole.

Favourite books: God Emperor of Dune

Favourite repos: cncf/landscape

SIG/WG participation: emeritus



Kaslin Fields

Developer Relations Engineering Manager

How they keep up to date: Kaslin co-hosts the Kubernetes Podcast and watches Krazam on YouTube. She stays updated through SIG meetings, release notes, community calls, and peer networks.

Favourite books: Working in Public by Nadia Asparouhova and Writing for Developers by Piotr Sarna and Cynthia Dunlop

Favourite repos: `kubernetes/community` and `kubernetes/community/communication/contributor-comms`

SIG/WG participation: SIG ContribEx (Co-Chair), Contributor Communications subproject (co-lead)



Katie Greenley

OSS Community Manager @ Chronosphere

How they keep up to date: Katie listens to DevOps and Docker Talk, Kube Cuddle with Rich Burroughs, and KubeFM. She watches Techworld with Nana, WiggityWhitney, and Kubesimplify.

Favourite books:

Favourite repos: cncf and fluent/fluent-bit

SIG/WG participation: Platform Engineering



Lin Sun

Head of Open Source @ Solo.io, CNCF TOC member/Ambassador

How they keep up to date: Lin follows the Kubernetes podcast from Google and watches Tedx and CNCF content. She tracks releases/KEPs, maintainer chatter, and user feedback.

Favourite books: Steve Jobs

Favourite repos: istio/istio

SIG/WG participation: Istio, kgateway, kagent, and agentgateway



Phil Estes

Principal Engineer, Container/Cloud Native Open Source Strategy

How they keep up to date: Phil listens to Cloud Native Devops & Docker Talk with Bret Fisher and watches CNCF conference talks to catch up on events.

Favourite books: Where Wizards Stay Up Late: The Origins of the Internet

Favourite repos: containerd/containerd and estesp/manifest-tool

SIG/WG participation: SIG-Node



Prasanth Baskar

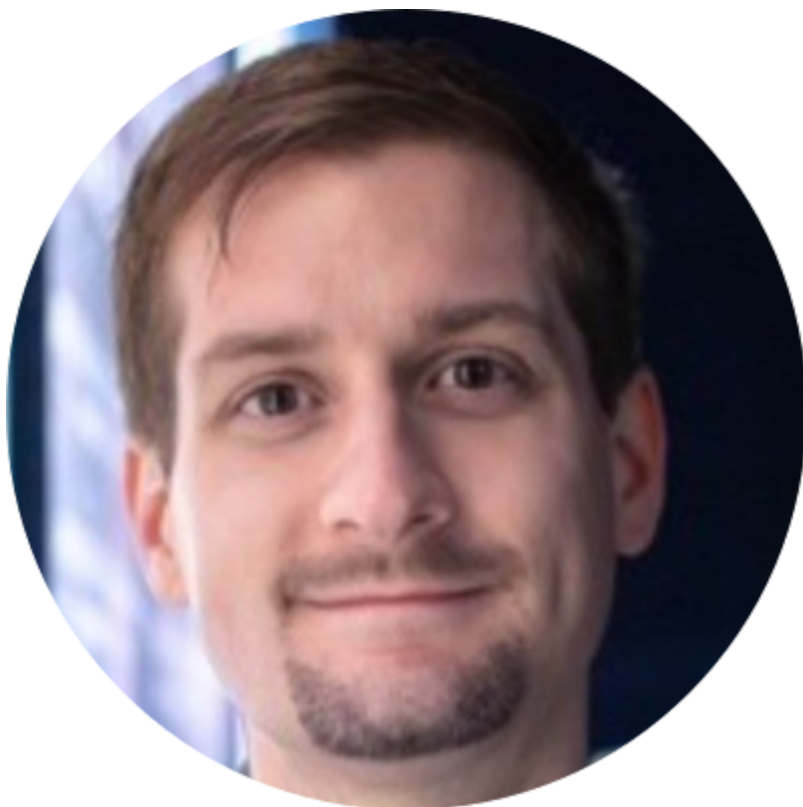
Software Engineer

How they keep up to date: Prasanth follows KubeFM and The Pragmatic Engineer podcasts. He watches Fireship and theprimeagen on YouTube, and attends mailing lists and meetups.

Favourite books: Atomic Habits

Favourite repos: goharbor/harbor

SIG/WG participation:



Taylor Dolezal

Head of OSS @ Dosu

How they keep up to date: Taylor listens to Accidental Tech Podcast (ATP) and watches Kitboga on YouTube. He tracks releases/KEPs, maintainer chatter, and end-user feedback.

Favourite books: Antimemetics: Why Some Ideas Resist Spreading

Favourite repos: assafelovic/gpt-researcher

SIG/WG participation: Previously participated in release teams and working groups, considering SIG Docs



Whitney Lee

Senior Technical Advocate @ Datadog

How they keep up to date: Whitney creates her own podcast Software Defined Interviews and watches DevOps Toolkit on YouTube. She reviews her own content on the wiggitywhitney channel.

Favourite books: Julia Evans Wizard Zines

Favourite repos: killer.sh

SIG/WG participation:



Yasmin Rajabi

COO @ CloudBolt

How they keep up to date: Yasmin listens to Good hang with amy poehler and watches Justin Garrison and Containers from the Couch on YouTube. She stays informed through Kubernetes events (KubeCon, KCDs), conversations with peers, engineering team/Slack channels, and podcasts.

Favourite books: How to Win Friends and Influence People

Favourite repos: CloudBolt's docs repo (not public)

SIG/WG participation: