5 Ways to Foster a Positive Developer Experience

splunk>
a CISCO company

A tale of two developers

Let's peer into the lives of two software developers. Sage and Kai may live entirely separate lives, but they have a lot in common. They start their workdays with strong coffee in quirky mugs. They're both creative problem solvers and whiteboard enthusiasts. They both have a knack for bringing abstract ideas to life. With the right set of tools and support, Sage and Kai have the power to dream up and build practically anything they — or their organizations — could imagine.

However, when we look more closely, we see that the experiences of these two developers are vastly different. Kai struggles with outdated tools. His codebase is a tangled mess that stifles his creativity, and he's constantly firefighting. He often works overtime and leaves work feeling burnt out. In contrast, Sage thrives with cutting-edge tools. Her org has a robust platform engineering team and a well-organized, modular codebase, inspiring creativity and innovation. She can make proactive and strategic improvements, allowing her to leave work feeling energized and satisfied.

While both developers are capable, only one is empowered to reach their full potential and is well-positioned to succeed.

A developer-centric culture: Why it matters

Developer experience doesn't stop at Sage and Kai — it has a ripple effect throughout their organizations. Sage and her team are in a flow state of creativity, so they innovate quickly and constantly. Like a well-oiled machine, they glide through robust software releases and seamlessly deliver game-changing solutions for customers. The company thrives because of this. With innovative software and processes, they see fewer outages, happier customers, and more revenue.

Let's also not forget about job satisfaction and retention. Kai's company struggles to retain top talent, and developers constantly search for new jobs to escape the chaos. This causes even more setbacks for the company, dragging down morale and productivity. Conversely, Sage works alongside a close-knit team, in which each developer is contributing at their full potential. Their productive and happy senior developers have time to train newer team members, and everyone partners to write new features instead of constantly troubleshooting.

Both now and in the long term, Sage's company will benefit from properly maintaining one of their most expensive resources: talent. According to DEV Community, recruitment costs can amount to 30-50% of a developer's annual salary. Onboarding and training for new developers adds an additional expense — this process can take up to nine months and 20-30% of a developer's annual salary. Finally, low retention also leads to productivity losses, which cause the biggest financial impact: losses of 50-70% of a developer's annual salary.

According to DEV Community, recruitment costs can amount to

of a developer's annual salary.

Setting the stage for developer success

Clearly, developer experience has a massive impact on your organization's productivity, employee retention, and bottom line. So, how can you (and Kai) follow in the footsteps of Sage's team to achieve the ideal flow state? Here are some tactics to implement.

1 Platform engineering

Platform engineering has emerged as the vanguard of modern engineering, with an ultimate mission to empower developers. It frees developers from the infrastructure management responsibility, lightening the cognitive load and granting greater code ownership. This leads to higher innovation velocity, reduction in shadow IT risks, and of course, a more positive developer experience.

Luckily, Sage's company leadership knows that an investment in platform engineering is an investment in its developers — and everyone involved reaps the benefits. Her team collaborates frequently with supportive platform engineers who understand how developers write and deploy code. This means Sage and her team have access to solutions that align with their needs and workflows, and their productivity is at an all-time high. Although platform engineering isn't necessary for every organization, it abstracts away vendor implementation details so developers can focus on reaping the benefits of their tools rather than wasting cycles on provisioning and configurations.

2 Automation

Another key tactic for transforming the developer experience is automation. Automation enables teams to innovate more quickly and efficiently. With strong CI/CD pipelines, developers can eliminate manual, time-consuming production processes and reduce the overhead that comes with release engineering teams.

Sage's team uses automated processes to allow developers to focus on creative problem-solving and building new things instead of repetitive tasks. Their goal is to deploy constantly to keep up with the rapidly changing business landscape. With higher deployment frequency, they can accelerate feedback, reduce risk, and improve efficiency.

3 Seamless observability

Kai and his colleagues are smart. They know how to fix problems. But more often than not, they lack the visibility to even identify what's broken in the first place. Because of this, they struggle to work efficiently and can end up in costly war rooms. With their modern distributed systems, they need observability to keep their code up and running and their customers happy.

Observability helps your developers avoid alert fatigue, and fatigue in general. Instrumentation should be a standard development practice (ideally integrated within service templates delivered by platform engineering teams). When developers deploy new code, they need the ability to easily validate that it's working. They want to be confident in their code — they don't want to guess around or get paged. This can be accomplished with automated testing and monitoring in their observability platform.

According to our State of Observability 2024 report, organizations with advanced observability achieve a 22% higher change success rate (a key DORA metric) for production application code. Even more impressive: These organizations say the changes are successful 90% of the time or more. On top of that, organizations with advanced observability spend about 38% more time on innovation versus routine tasks like maintenance, alert handling, and configuration.

4 Alert hygiene

Kai knows all too well that too many alerts equal developer fatigue. He and his team struggle to sift through all the constant irrelevant and low-priority alerts, which make their environment noisy and stressful. A high volume of alerts makes it difficult to tackle the ones that really matter, allowing incidents to slip through the cracks. They often end up in costly war rooms, keeping them from concentrating on what matters. All of this is a telltale sign that Kai and his team are having a poor experience.

When alerts are meaningful and actionable, developers can work more efficiently to ensure incidents don't cause downtime. Sage's team avoids alert overload by unifying monitoring, fine-tuning detectors, intelligently grouping alerts, and practicing good alert hygiene. And when an incident does happen, her team of developers feels supported and confident. They have processes and policies around incident management, runbooks, escalation policies, communication channels, and postmortems to reduce the chances of repeating an incident.

5 Allow them to focus on their strengths

Anything that pulls developers away from their code can impact their experience. On Kai's team, developers are constantly context-switching, which distracts them from important tasks and leaves them feeling exhausted. It's no surprise — according to Atlassian, 45% of people report that context switching makes them less productive.

Kai's team often fails to achieve their planned roadmap milestones due to these distractions, which is a sign that the workplace is not developer-friendly. He and his coworkers are stuck in an endless cycle of side tasks, constantly forced to figure out new service and tool configurations, plan releases, and parse through alert noise. This keeps them from completing their deliverables and isn't a good use of their time — instead, they should be focused on writing the feature-related code.

In contrast, Sage's developer environment is highly optimized. Thanks to practices like platform engineering, observability, and automation, the developers enjoy long stretches of uninterrupted coding time. As a result, they consistently stick to their roadmap and innovate rapidly, delivering value to the business while maintaining a healthy work-life balance.

To keep developers happy and your organization running smoothly, it's crucial to invest in the right technology. Splunk offers unified observability that automates processes, eliminates the need for context-switching, and ultimately makes developers' lives easier (as well as the rest of your team.)

Want to keep learning? Check out this e-book to discover how to empower your team with unified observability.

splunk> a CISCO company

Splunk, Splunk> and Turn Data Into Doing are trademarks and registered trademarks of Splunk LLC. in the United States and other countries. All other brand names, product names or trademarks belong to their respective owners. © 2025 Splunk LLC. All rights reserved.

25_CMP_ebook_5-ways-to-foster-a-positive-developer-experience_v6