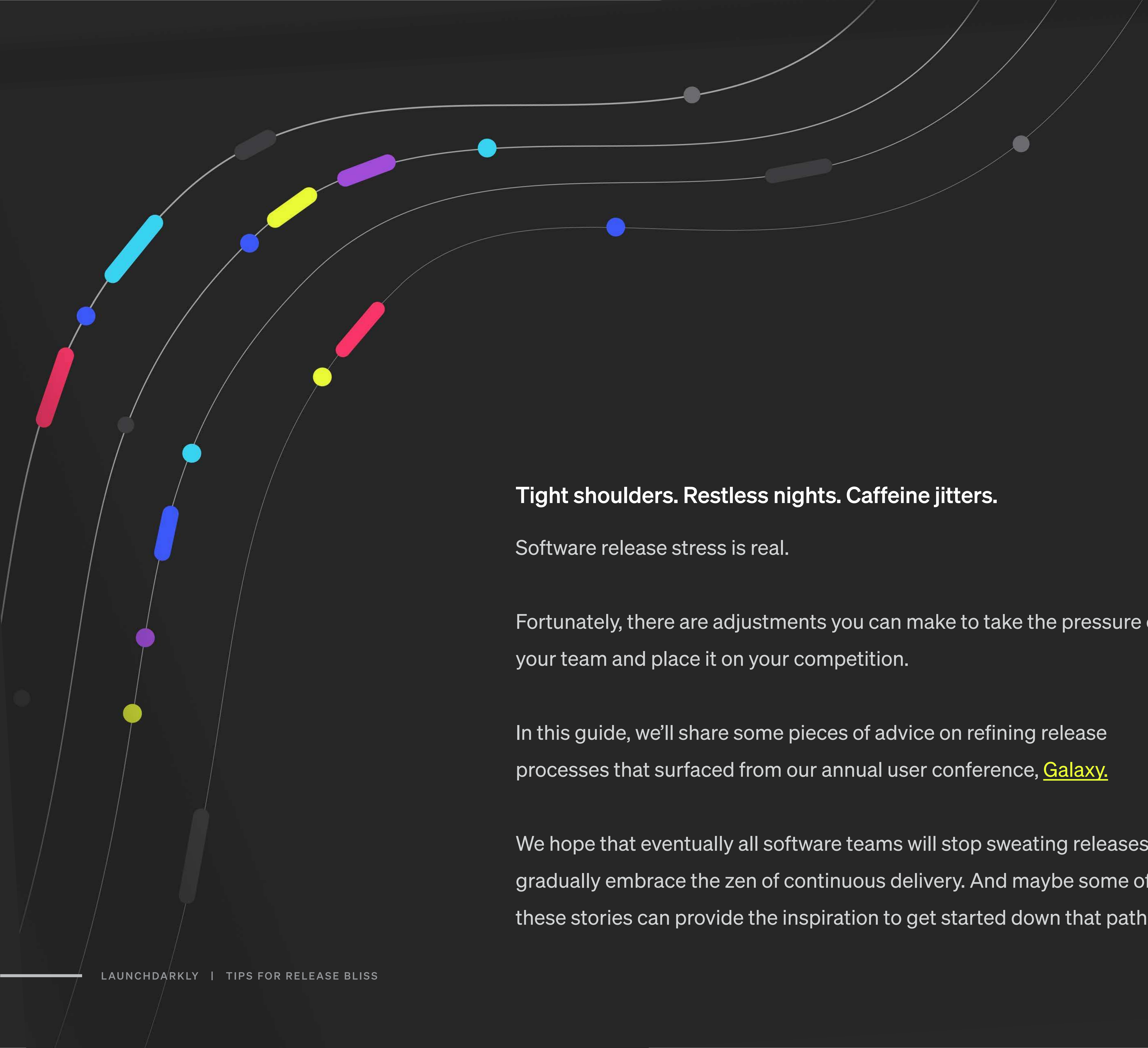
7 Tips for Stress-Free Software Releases

Sage advice from software professionals on testing in production, ditching unworkable systems, and questioning your process.

LaunchDarkly

Contents



Tight shoulders. Restless nights. Caffeine jitters.

Software release stress is real.

Fortunately, there are adjustments you can make to take the pressure off your team and place it on your competition.

In this guide, we'll share some pieces of advice on refining release processes that surfaced from our annual user conference, Galaxy.

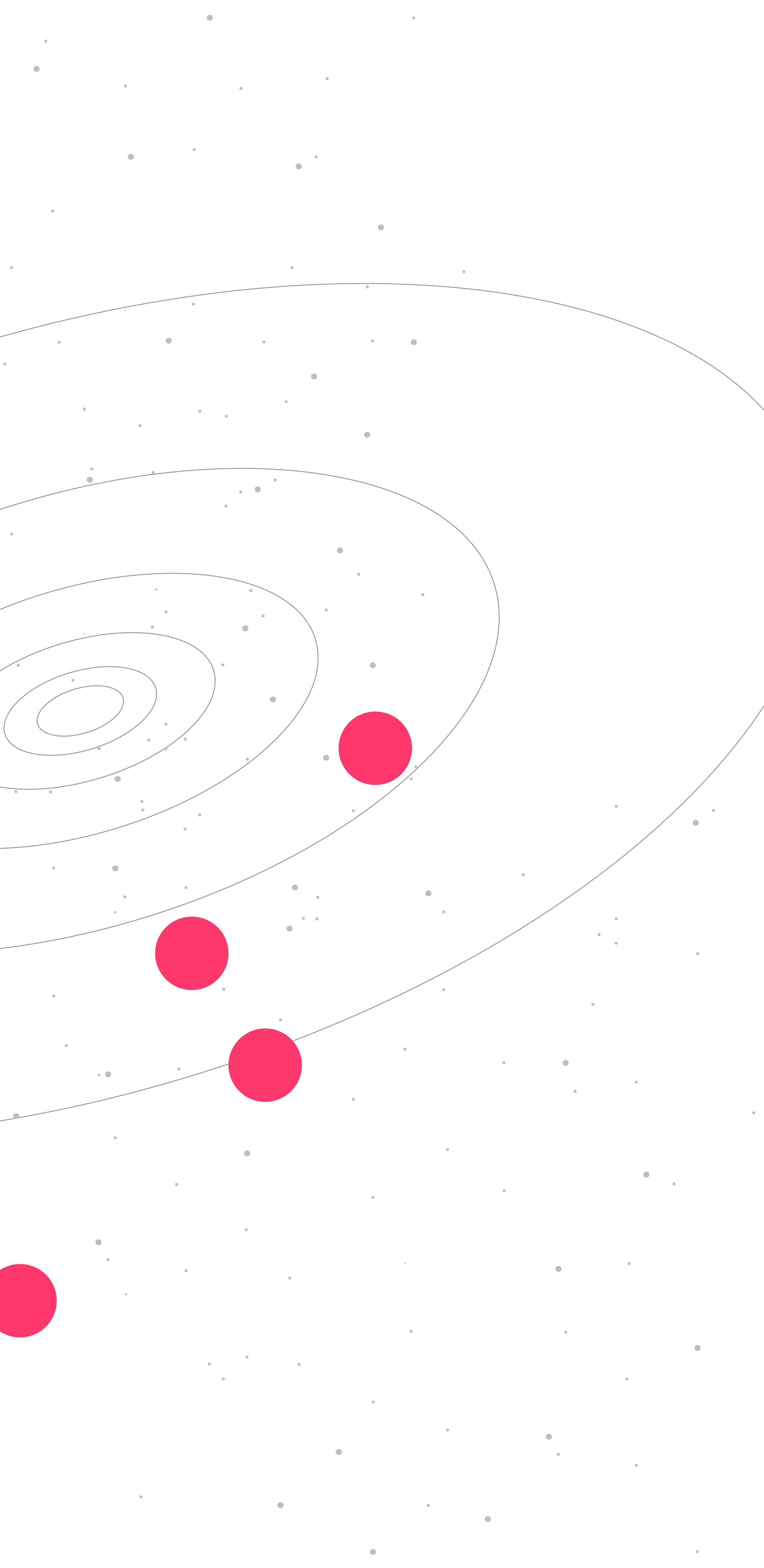
We hope that eventually all software teams will stop sweating releases and gradually embrace the zen of continuous delivery. And maybe some of these stories can provide the inspiration to get started down that path.

TIP #1

Don’t fear testing in production

The term “test in production” is polarizing, but that’s largely because it’s misunderstood.

Some see the phrase and immediately feel like it’s telling developers to be reckless, when our view is the exact opposite.



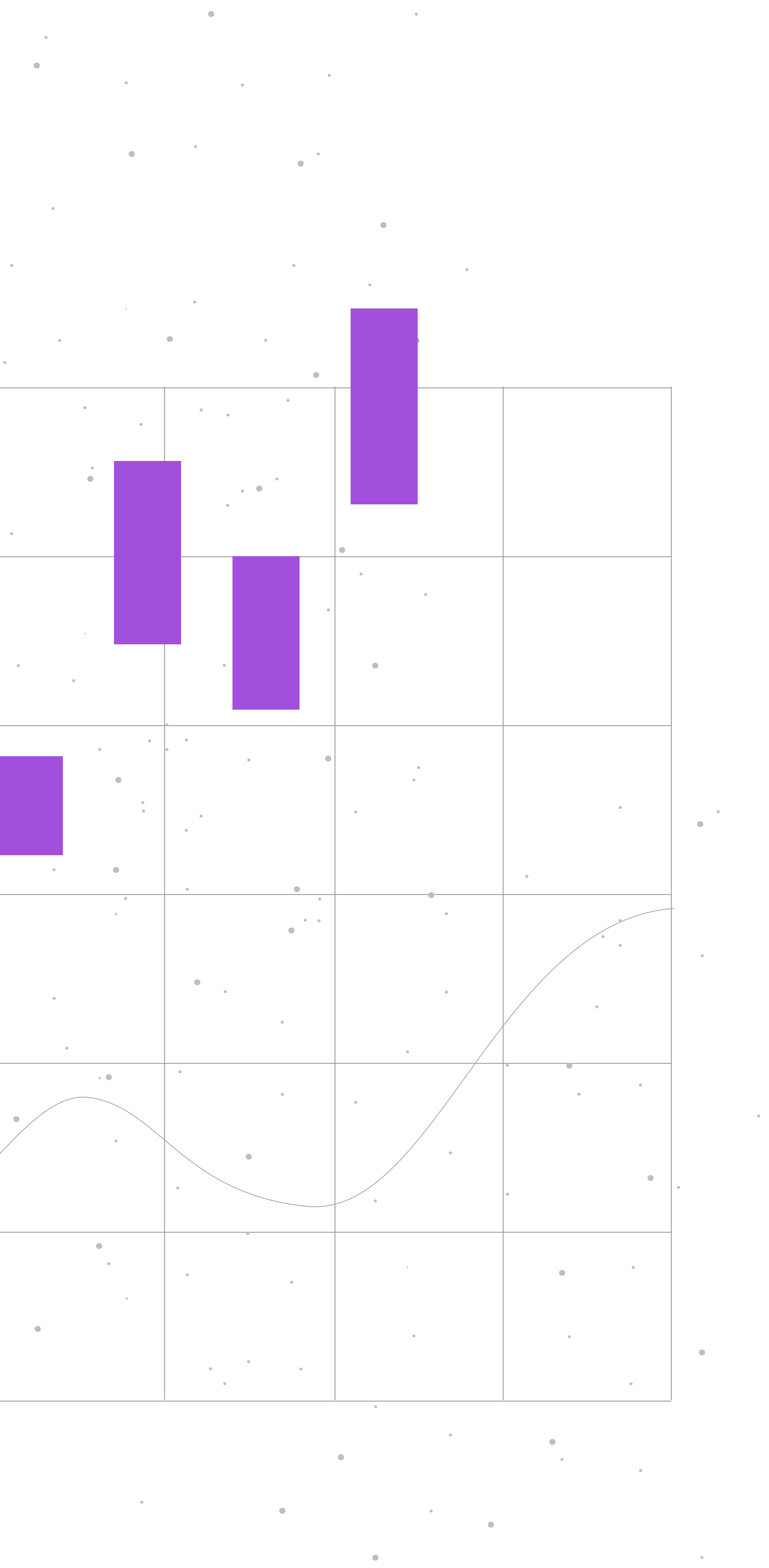
“Testing in production” refers to the practice of running code on production servers, using real data from real users, without showing the new behavior to the majority of users. These tests are frequently run during the final stages before releasing software to a broad audience.

To be clear, testing in production is not a substitute for quality assurance (QA), or a shortcut to eliminate unit testing or integration testing. Instead, it is an extension of testing and QA control points into the most realistic environment possible: the real-world.

Testing in production becomes a lot less scary when you have the ability to immediately turn off a test through a feature flag if something goes wrong.

Feature flags are a software development process that can enable or disable functionality remotely without deploying code. This allows you to deploy new features without making them visible to users. Feature flags help decouple deployment from release letting you manage the full lifecycle of a feature.

And using feature flags can give you the confidence you need to do things you never thought possible—like testing in production on Black Friday.



In North America, Black Friday is a massive kickstart to the holiday shopping season, wherein retailers of all varieties offer significant discounts on goods and services.

In years past, Black Friday might have conjured images of people lining up at big box stores, hoping to score some deals whilst also avoiding any sort of drama over, say, limited quantities of a new Xbox.

With online shopping skyrocketing in recent years, the more current scene is shoppers in pajamas sitting at home navigating the same sales via their phones and laptops. In 2020, consumers spent $9 billion on Black Friday, which was up 21.6% year over year, according to data from Adobe Analytics.

For developers, the pivot to online shopping by consumers has put more pressure on software performance in the face of massive traffic surges to ensure e-commerce storefronts don't go down on the biggest shopping day of the year.

That's why it was impressive to hear how feature flags gave StockX the confidence to try something new on a recent Black Friday in the midst of unprecedented traffic rushes. StockX is a live marketplace for some of

“

We turned it right back on a week later on Black Friday—the biggest day of the year—with the most traffic we have had until that point, which is a very scary thing.

the hottest consumer goods available, from exclusive and limited edition sneakers to streetwear, handbags and watches.

On Black Friday, StockX used LaunchDarkly to A/B test a new home page against an old one to see which would perform better. “We had this new technology, and what better day to showcase it than Black Friday?,” says StockX’s senior software engineer, Kyle Schrade, in a recent talk at LaunchDarkly Galaxy. “It’s the biggest shopping day of the year so if our site went down, that would be a disaster.”

Shrade says StockX started testing its homepage a week before Black Friday, but had to turn off the A/B testing because there was a problem on the backend. After making some tweaks, they decided to just go for it on the big day.

“We love punishment sometimes,” Shrade says, “so we turned it right back on a week later on Black Friday—the biggest day of the year—with the most traffic we have had until that point, which is a very scary thing.”

The result was exhilarating, because Shrade’s team began noticing the new homepage’s elements quickly outperforming the old.

Galaxy Talk: The Art of Shipping Broken Code

On Black Friday, StockX A/B tested a new home page against an old one to see which would perform better.

Check out the full talk.

“We were just caching things better,” he says. “We had way more hits than we had misses compared to the old stuff, which was huge. And by the end of the day, we actually turned on more people through the new home screen.” Not only was the new homescreen causing fewer problems, but the test began immediately paying off in other tangible ways.

“The new home screen won out by a very, very wide margin, based on people actually buying product,” Shrade says. “So, that new home screen was so nice that people actually bought more shoes and more items.

The lynchpin behind this sort of high-stakes A/B test was the confidence Shrade’s team gained by having a safety net provided by feature flags, should something have gone terribly wrong. If you release software behind a feature flag and it’s not performing to your satisfaction, you can simply turn it off instantly.

“Everything you ship should have some type of kill switch or safety net,” Shrade says.

TIP #2

Question your organization’s release process

As organizations grow, it’s tempting to stick with the way things have always been done.

And yet, stepping back and asking questions can yield big results.

Maybe your organization's level of complexity feels unnecessary. Sharing lengthy spreadsheets and documents around to track the status of certain pieces of your release process, for instance, can get messy. Or maybe your QA team is becoming a bottleneck.

When we're talking about something as critical as your company's external-facing software presence, you can't really afford to be stuck in some archaic process that only ends up slowing you down and causing confusion.

A little while ago, VTS Canada was experiencing some level of seemingly unnecessary complexity in their release process, according to Caley Brock, a senior software engineer at VTS, as relayed in a recent talk at LaunchDarkly Galaxy.

In order to release a feature, Brock's team would write the code, write their test, and then create a custom environment just for the branch of code that held the feature. The QA team would then go into the environment, test everything, and only after the code was approved would it be merged to the main branch and released.

20% reduction

in cycle time while still maintaining quality metrics

Brock’s team thought there was a better way, so they introduced something called asynchronous QA or “AsyncQA” for short. In this process, her team’s code is written and tested behind a feature flag. This means the code gets immediately merged and deployed. Because the changes are safely hidden behind a flag, QA can test anytime within an integrated environment or in production.

AnsyncQA eliminates the need for one-off environments associated with the traditional QA process. And once QA tests and approves everything, the team can turn the flag on for all users. Afterwards, the ticket comes back to the developer and the team removes a flag from code and from the tool.

The results have been inspiring. Brock’s team no longer has to wait for QA to merge, which means no more stale code and allows the new code to be integrated much faster. After tracking cycle time—the time from starting a ticket to when the users are seeing value—they’ve seen an over 20% reduction while still maintaining quality metrics.

Supercharge your release pipeline with LaunchDarkly configs

Hear directly from Caley Brock, senior software engineer at VTS, about the LaunchDarkly setup that enabled the VTS teams to ship faster and streamline testing while avoiding getting bogged down with feature management.

Watch the full talk.

“This means that users were seeing our changes faster and we’re still not seeing any more bugs than before, which is great news,” Brock says.

Her team continues to push for more efficiencies and has gotten them thanks to the introduction of LaunchDarkly, which has helped make feature flags easier to find and manage, among other things.

As the adage goes, the only bad question is the one you never asked.

TIP #3

Ditch your homegrown systems if they're not serving you

As the great Paul Simon once said, there must be 50 ways to leave your homegrown software system.

We're paraphrasing Mr. Simon, but of course not all lovers, homebrewed systems, or even open source options are bad.

Migrating to a new feature flag system

Autodesk wanted its new division to switch to LaunchDarkly, and there were some key learnings along the way for any other organizations curious about making the leap.

Watch the full talk about how Autodesk did it.

When you're an up-and-coming startup with a small team, for instance, the DIY route can be preferable. But eventually, things are going to get pretty gnarly once you begin to scale. For Autodesk—a global leader in in 3D design, engineering and entertainment software—a newly-acquired division of its organization had been relying on a homegrown feature management system with mixed results.

The homegrown system was easy to use and had some cool features, but among its main drawbacks were that it had no central UI and that the maintenance of the solution often required its developers to take valuable time away from working on its main product. Dragging your developers away from their main priorities to perform maintenance on the in-house solution, thereby risking delays on tight release schedules, is never ideal.

Once the move was made to go with LaunchDarkly, Rick Riensche, a senior software engineer at Autodesk, says it more than paid off. "In addition to getting to use the features of LaunchDarkly, and no longer having to maintain our homegrown system, we reaped some pretty impressive performance awards," he says. "Overall, response times dropped dramatically and became drastically less sensitive to load when we eliminated millions of Redis calls that were happening per minute in the old system."

TIP #4

Find opportunities to improve your process in turbulent times

Launching new software features in February 2020 was a lot different a month later.

Once the COVID-19 pandemic hit the world in earnest, everything changed.

“

The one piece of advice I can give you if you know everything else... Everything should have a flag on it.

Companies scrambled to adapt their internal working processes, as well as keep up with demand fluctuations from customers.

In March of 2020, the team at Ally Financial had been working on a new employment verification feature, which would verify the information about customers for federal regulations.

The employment verification process feature was set to release that month, just as the team was scrambling to get all of its employees set up for working from home during the pandemic.

When the barrage of news came down that thousands of people were now out of work due to the drastic shift in business, the company decided it wasn’t the best idea to release a new feature around employment verification.

Fortunately, the feature was behind a feature flag. So instead of creating a new build or deploying new artifacts to change things, the team simply left the feature off. And that pause eventually gave the team the opportunity to test the feature in production, which was something they had been unable to do prior to using a good feature flag platform like LaunchDarkly.

Choosing the Right Tool

Using the right tool has created several new possibilities for Ally Financial to safely and quickly deliver features to delight its users.

Watch the full talk.

And, since feature flags also provide the ability to be a lot more nimble in rolling with the waves of turbulent changes that the times were bringing, the team became resilient.

“In the early days of the pandemic, we had a flurry of other changes that were designed to be temporary: call center hours, fee waivers, special messaging,” says Jeremy Cox, application manager at Ally Financial. “And we wrapped everything in a feature flag so that we could turn it off at a moment’s notice because we didn’t know how long these features would last.”

Using feature flags in LaunchDarkly has allowed the Ally team to open the door for full continuous delivery and given them the ability to automatically roll back features if there’s ever a problem.

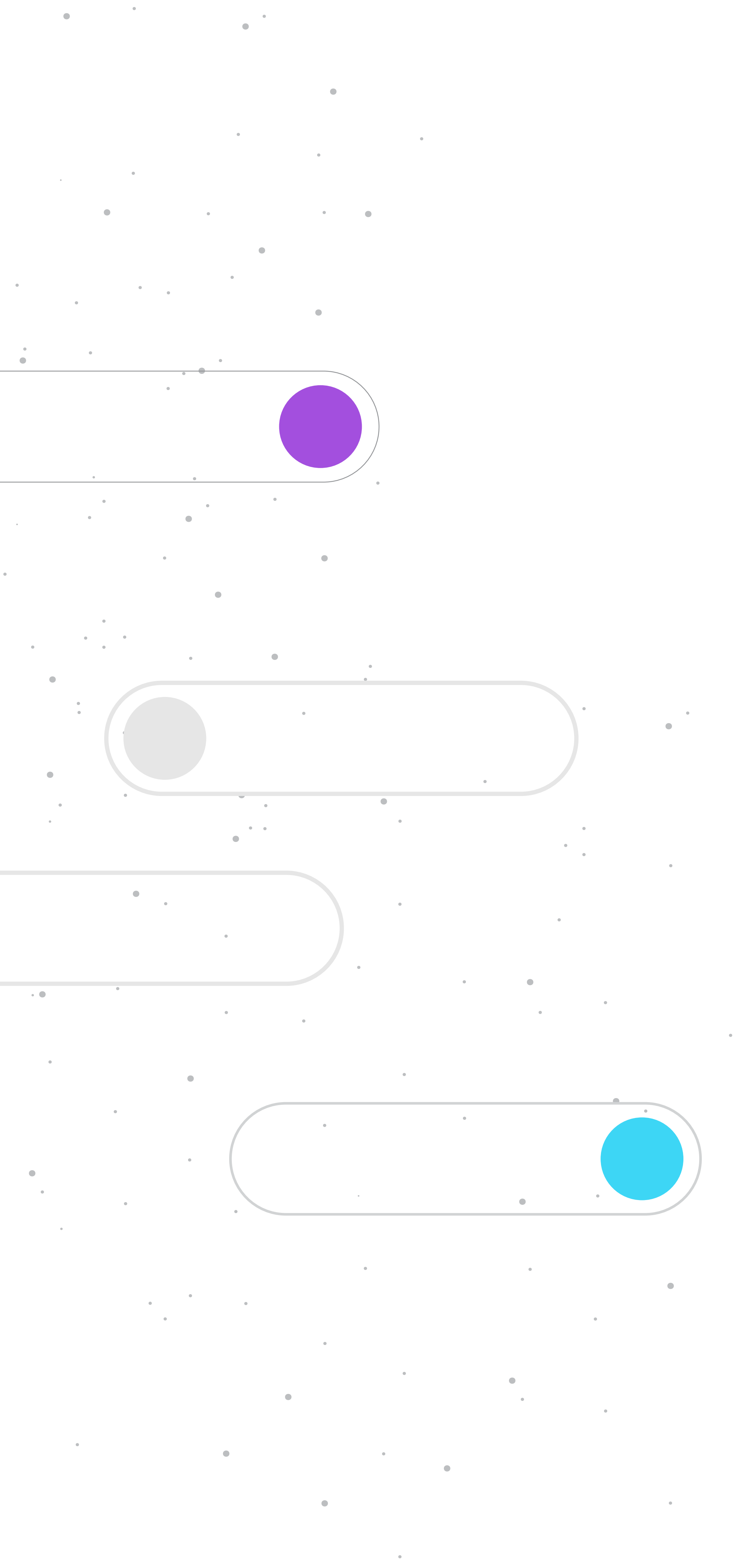
“The one piece of advice I can give you if you know everything else,” Cox says, “Everything should have a flag on it.”

TIP #5

Bring order to your chaos

“From chaos comes order” is a famous saying.

That may be true for some things, but when it comes to not having a strong handle on things like naming conventions and tech debt, chaos can create more chaos.



Let's say you're a company either practicing continuous deployment, or working your way towards it. You begin adopting feature flagging as a way of speeding up your process, and your team takes to it like second nature. At some point, maybe months or even years later, you may reach a point where you realize you've lost track of what each flag is doing, how long they've been in production, who is managing them, when they're going to be removed, and more.

You may have started your feature flag journey under the pretense flags would be temporary and removed shortly after a project was completed, but that hasn't been happening, for whatever reason.

Something similar happened to iPipeline—which focuses on life insurance software—and the development team had to come up with a solution, otherwise risk an endless cycle of mounting technical debt.

In this case, one of the answers was applying fitness functions.

A fitness function essentially is something that puts checks in place to assess how fit for purpose a piece of functionality is, and ideally these checks are automated. Implementing fitness functions can help assess

Fitness Functions and Flagging Conventions

In this talk, Mark Burry, senior developer at iPipeline, recounts the journey his company embarked on with feature flags, and how it eventually found fitness functions and flagging conventions as best practices.

Check out the full talk.

flags throughout their life cycles from as early as the naming of the flag all the way through to its deletion.

Along with the implementation of code references, naming conventions, a tagging strategy, running automated assessments on flags, and implementing web hooks to provide continued visibility, iPipeline is now able to run 330,000 plus daily flag evaluations across all the features in its US and UK products with full visibility into every flag and its status.

“Never leave a flag behind,” says Mark Burry, senior developer at iPipeline.

TIP #6

Repetition is the mother of learning

In many cases, it’s better to learn by doing.

And the more you do something, the easier it is to find tricks and efficiencies that will improve your process.

A test automation journey

See how LaunchDarkly helped Northwestern Mutual simplify releases, provide better bug isolation, reduce manual testing time, improve productivity, and better manage any damage scenarios.

Watch the full talk.

That was the case for Sapriya Balasubramanian, Assistant Director, Test Engineering, at Northwestern Mutual.

Her team was noticing a lot of repetitive testing in multiple environments that was adding up to prolonged debugging times and failures.

Among the reasons behind these challenges is that maintaining flags in an expected flag state in multiple environments can be difficult. When flags are not in the expected state before running automation tests, it can lead to extended test running times, raw failures, and false positives.

Luckily, Balasubramanian and her team discovered a correction.

“As the onion layer of innovation peels, we added another layer of automated flag creation and set flags to the right statuses and integrated that into our testing pipelines,” she says.

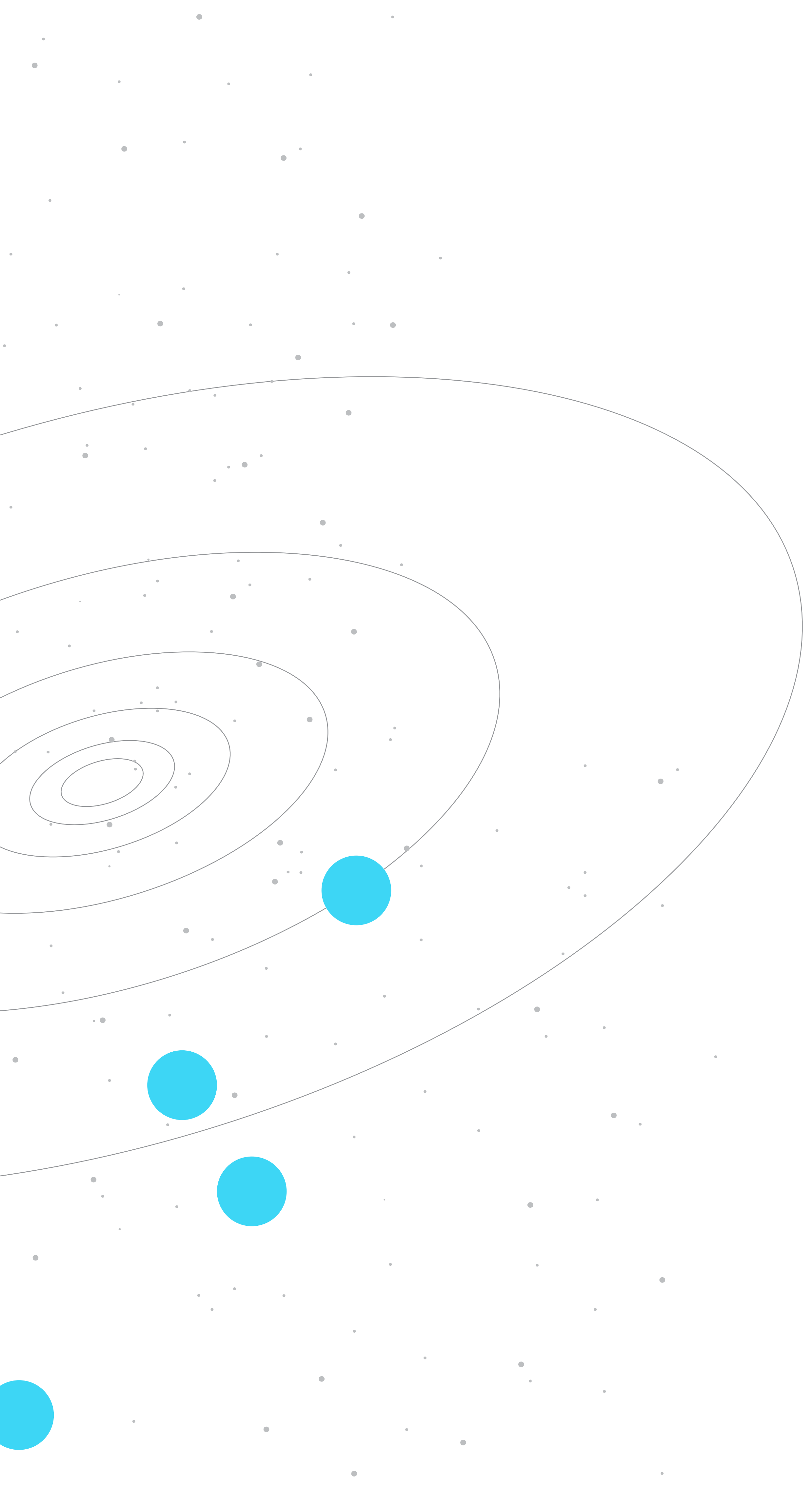
The result has saved significant testing time, reduced repetitive testing, and also lessened risk by ensuring the tests are run with the right setup on the right environments with the right flags and also in the right state.

TIP #7

Build consensus around improvements

You don't need to be at an organization with a reputation for being on the bleeding edge to make huge leaps in your process and products.

Great things can happen on any team, but major jumps forward usually have to have some organizational-consensus around them first.



In terms of your release process, there are lots of efficiencies that can be made on the team level. When it comes to bringing in a new tool to help you take a step towards continuous delivery, for instance, it's not always as simple as a credit card swipe.

From technical and security vetting to rollout and adoption, there are a lot of logistical elements to consider. You'll likely have many different stakeholders that will need to realize the software's value both ahead of the actual purchase and, perhaps more importantly, in the months afterwards. And your stakeholders are ultimately going to be the ones that help make your purchase a reality.

This can feel initially daunting to a new buyer, but if you're assured of the value the software will bring to your team, there are lots of resources available that can guide you.

In that spirit, Aaron Kaka, solution architect for Pearson—the world's leading learning company—provides some key points of consideration when you're attempting to purchase and implement a new tool. In his case, the solution he was trying to bring onboard was LaunchDarkly, although his ideas extend beyond that.

Launching darkly at large organizations

How do you realize the advantages of feature management and A/B testing at a large software-driven business that has yet to make it a best practice?

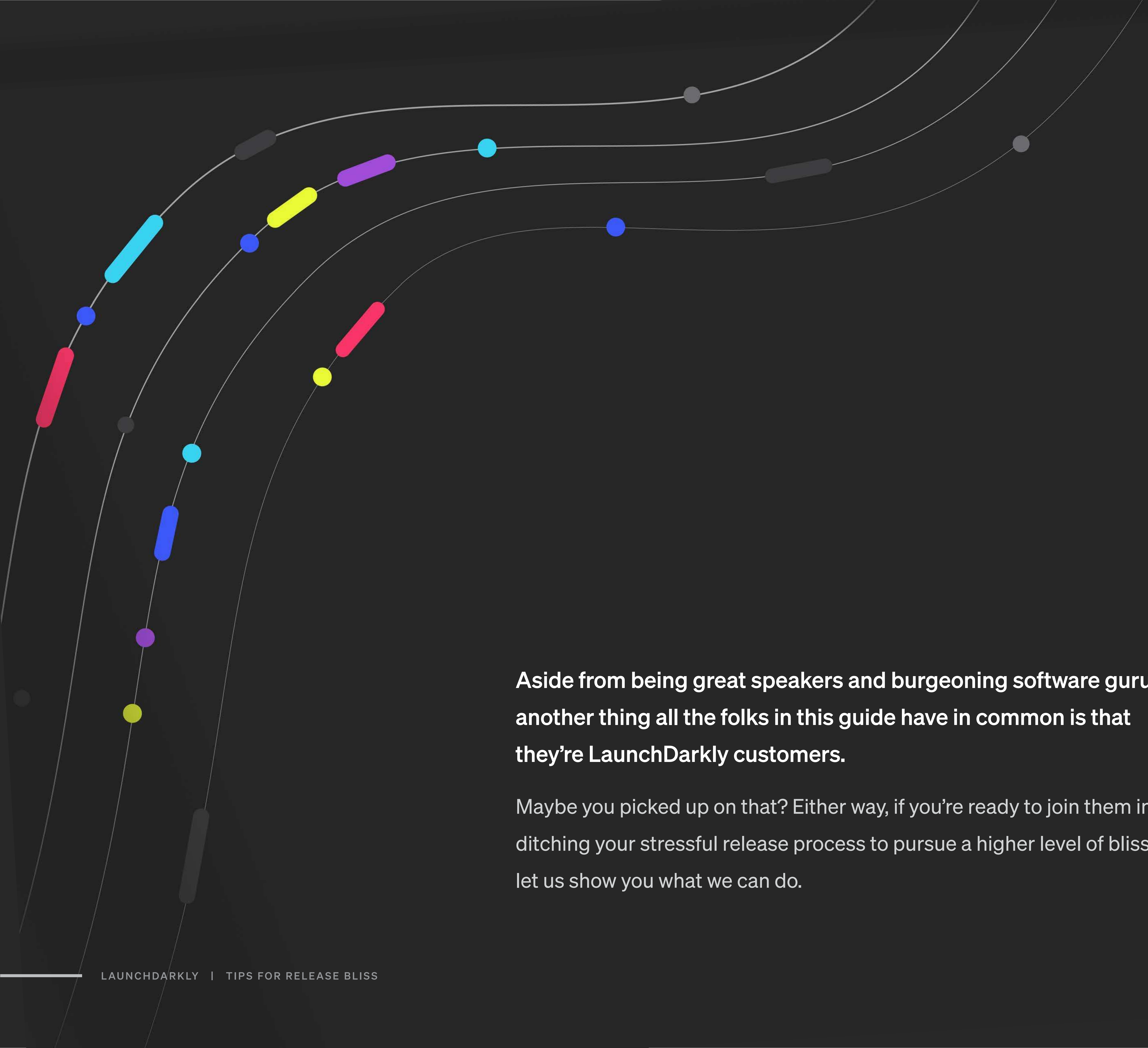
Watch the full talk.

"Do a live demo to the stakeholders," he says. "This includes not only the product QE and engineering managers, but leadership from partner teams, enabling services, and director-level management. This is critical for getting buy-in from your stakeholders because you're going to have to show them a fairly comprehensive demo with some wow factor."

"Perhaps things like how you can update the flag external to your app and in less than 200 milliseconds, the update occurs in your app without a page refresh. Or the ease of building dynamic business rules for a variation based on the core start date. And showing the ease of setting up a percentage rollout."

Kaka also recommends recording your demo and storing it on an internal, company site to increase awareness and accessibility.

Once the team is convinced, Kaka strongly suggests making a best practices guide for your tool to share with the rest of the company, and leaning on the solution's customer support team for rollout and implementation.



Aside from being great speakers and burgeoning software gurus, another thing all the folks in this guide have in common is that they're LaunchDarkly customers.

Maybe you picked up on that? Either way, if you're ready to join them in ditching your stressful release process to pursue a higher level of bliss, let us show you what we can do.

Empowering all
teams to deliver and
control their software.

launchdarkly.com

sales@launchdarkly.com

LaunchDarkly