LaunchDarkly

Fast vs. safe: The great dilemma in modern software development

3 keys to deploy more, risk less, and release with confidence.

Table of Contents

INTRODUCTION

Are you asking your developers to achieve the impossible?

Software is an economic driver. In a global survey spanning multiple industries, McKinsey found that nearly 70% of “top economic performers” use their own software to differentiate themselves from competitors.[1]

But software is only as good as the development and delivery processes beneath it. Indeed, research shows that teams with high software delivery and operational (SDO) performance achieve “exceptional organizational performance”—including hitting or exceeding profitability targets.[2] What’s more, McKinsey found that high developer velocity is associated with 4-5X faster revenue growth.[3]

For these reasons, business and technology leaders want to achieve engineering excellence. This travels downstream to developers, who feel constant pressure to build and deliver better features faster.

But the reality of modern development undermines developers, forcing them to choose between moving fast and breaking things or endlessly testing each release to reduce risk. This tension leads to:

- A frantic developer experience
- An unacceptable customer experience
- Low productivity

LaunchDarkly gives developers the power to overcome the defects of modern development. In this guide, we'll explain how your developers can move faster and more confidently by separating *deployment* from *release* and gain more control over rollouts. We'll also show how to recover from errors instantly and spend more time on what matters.

Who is this content for?

This guide is intended for software development and platform engineering leaders, as well as executives responsible for software or digital experiences.

[1] "Three New Mandates for Capturing a Digital Transformation's Full Value," McKinsey Digital, June 2022.

[2] "2022 Accelerate State of DevOps Report," DORA, 2022.

[3] "Developer Velocity: How Software Excellence Fuels Business Performance," McKinsey & Company, April 2020.

01

Top 3 barriers to fast, de-risked software releases

Today, software developers are stranded between fast-and-risky and slow-and-safe. Here’s why.

01

Fear of production creates a logjam of pre-production testing

Pressured to deliver perfect, bug-free releases, developers endlessly vet code in pre-production staging environments, then let it sit with quality assurance (QA). This artificial testing eats up considerable resources and slows developers down.

Testing in production can accelerate the process. But developers tend to avoid testing in production due to the perceived risks of the practice. Ironically, any time a developer deploys code, they are, in fact, testing in production. They're just doing it with their entire userbase. And to be fair, developers typically lack the control required to test in production deliberately and safely. Nevertheless, development teams will continue to slog if they excessively test code in staging.

Every developer tests in production... Some admit it and plan for it.

3 simple principles of modern software development:

#1 You can't release a feature to production until you're sure it works.

#2 You can't be sure a feature works until you've released it to production.

#3 Go to 1.

02

Errors bring down the application, disrupt customers, and take ages to fix

Most developers lack the ability to quickly revert to a known safe state after a bug or outage. This forces teams to maniacally test code before a release to reduce the risk of impacting customers.

When a bug causes an outage, software engineering teams often need to perform a manual rollback to stop the bleeding. That means routing all production traffic—potentially millions of users—to an older version of the application and muscling changes through the deployment pipeline.

It's a cumbersome, stressful, painful exercise that developers will go to just about any lengths to avoid. Even worse, revenue losses and user frustration will mount as developers scramble to bring the application back online. Users have little patience for downtime or broken features.

$137 - $9,000 per minute[4]

Average cost of application downtime, depending on the size of your business

4 "Calculating the Cost of Downtime," Atlassian, accessed June 23, 2023.

03

Homegrown feature flag systems devour precious time

Many of the challenges we've outlined can be mitigated with feature flags. Feature flagging is a modern software development concept that allows developers to enable or disable a feature without modifying the source code or redeploying their application.

Software engineering teams adopt feature flag systems to increase development speed and reduce risk. But, with homegrown solutions, the additional time spent building, managing, optimizing, and repairing the tool can easily cancel out any gains in developer efficiency.

Typically, homegrown solutions lack an intuitive user interface and require developers to create duplicate flags for each application environment, platform, and service—or risk providing a poor, inconsistent user experience. Not to mention, developers must keep their homegrown tool from causing data breaches, incurring excess technical debt, and degrading system performance.

It all adds up to a significant burden on developers—one that pulls their focus away from creating features that serve customers and generate revenue.

02

How LaunchDarkly can help

With LaunchDarkly, you no longer have to choose between slow-and-steady and fast-and-reckless software development. Our feature management platform lets you develop and deploy features faster, reduce risk, and release with confidence.

In this section, we'll explore how three specific benefits of LaunchDarkly feature management address the challenges previously outlined.

Feature management

Feature management is a class of software development tools and techniques rooted in feature flags. Feature flags are conditional statements (e.g., if-else) that allow developers to deploy features to a production environment without making them visible to end users (i.e., decouple *deploy* from *release*).

Feature management enables teams to use flags on a massive scale across a variety of sophisticated use cases. Such use cases may involve targeting, multivariate flags, and experimentation as well as the need to support multiple teams, environments, and applications. Feature management increases developer productivity, reliability, and software quality.

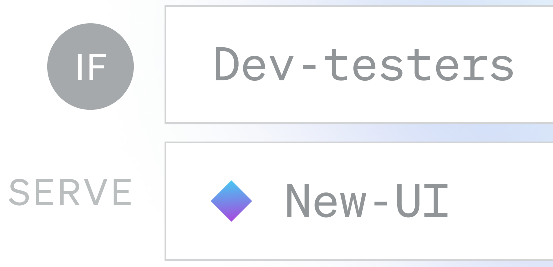
For the TL;DR crowd, here's a chart that previews the next section of content :

	Before	With LaunchDarkly
Testing in production	Source of great fear	You're in the clear
Broken code	Brings down the app	Turned off in a snap
Developer time	They don't have enough	Spent building cool stuff

01

Safely test in production

LaunchDarkly lets teams release features for validation in production before exposing them to customers, reducing the likelihood of bugs and outages that impact the business.



Limit your blast radius

With LaunchDarkly, your developers can deploy features and code changes in production to a small subset of internal or external users—thus allowing production testing while limiting your blast radius. With fine-grained targeting, you can easily define a hyper-specific audience segment across any number of parameters, then deploy to that audience instantly with a simple toggle.

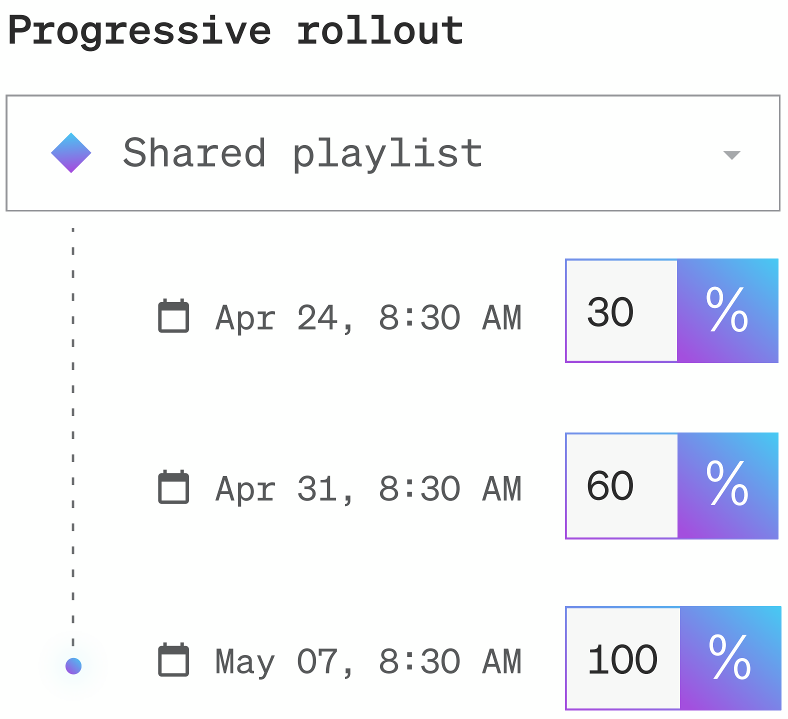
Without feature flags, when developers release a feature to production, they release it to all users and environments at once. If the feature causes a bug, it impacts *all* users (= high risk). Whereas, when using flags to release to a small subset of users or environments, that same bug would impact a much smaller audience (= much lower risk). Shrinking the blast radius of potential errors makes testing in production safer.

Release progressively

LaunchDarkly also enables progressive delivery, i.e., the ability to incrementally roll out new functionality, often starting with a small group of internal testers and expanding slowly and methodically to customer segments.

Our platform allows your teams to perform this in a controlled, automated way. For example, you can release a new feature to a random 5% of your user base on Day 1, then gradually expand to 15% on Day 3, then to 25% on Day 7, and so on until reaching 100%. Alternatively, you could be more intentional—choosing to roll out the feature to specific users, at particular times, or with other rules that you define. And you can schedule and automate the execution of these release patterns ahead of time.

Progressive delivery mitigates risk by enabling your teams to start small and catch problems early with real data. What's more, it gives your development teams the control and confidence to move faster.



Build efficiently

By enabling developers to safely test in production, LaunchDarkly reduces the need for pre-production staging environments. Conventionally, developers will build, maintain, and test changes in staging, despite the fact that staging often fails to adequately mimic production. In skipping pre-production testing, teams cut costs and improve efficiency.

Another way LaunchDarkly increases developer efficiency is through enabling trunk-based development—a method of version control branch management that seeks to reduce complexity and user error. LaunchDarkly allows developers to wrap incomplete code in feature flags and merge it with the mainline ("trunk") on a daily basis. If the mainline gets deployed to production, developers can rest easy knowing that their code is safe behind a flag. Trunk-based development is more efficient—and less painful—than periodically integrating long-lived feature branches as a part of big merge events. Ultimately, trunk-based development lets your teams ship fixes and changes to customers faster and with fewer conflicts.

$3.37 million

Three-year cost savings achieved by using LaunchDarkly to reduce the need for pre-production staging environments[6]

6 "The Total Economic Impact™ Of LaunchDarkly," Forrester study commissioned by LaunchDarkly, January 2021.

02

Instantly deactivate broken code

Errors in production happen, but they shouldn't take down your entire application. LaunchDarkly acts as a safety net by letting your teams revert to a safe place at any time without disrupting customers.

Instantly release and roll back

LaunchDarkly offers kill switches that allow your teams to remotely turn a feature off in production, without having to push code through your deployment pipeline or change approval process. In the absence of feature flags, when an incident occurs, it's not uncommon for multiple engineers to spend hours—sometimes in the middle of the night—trying to resolve the issue.

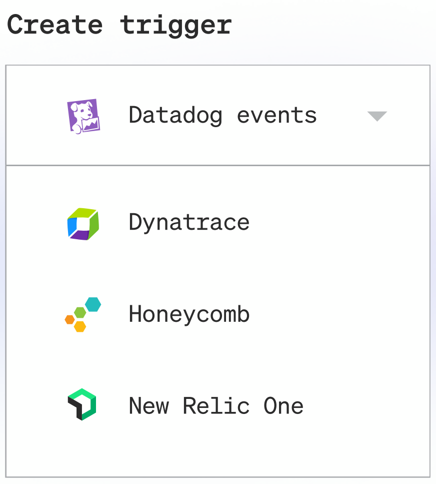
But with LaunchDarkly, a single developer (or anyone, really) can toggle the flag associated with the error and instantly roll back to a working version of the application. Then, they move on with their life.

Automate outage remediation

Flag triggers in LaunchDarkly take the kill switch a step further by allowing you to "automatically" deactivate broken code. They provide an additional layer of risk mitigation, enabling your teams to resolve incidents in near real-time.

Our platform integrates with popular observability and application performance monitoring (APM) solutions. In cases where performance metrics like error rates exceed a certain threshold, your APM will trigger

LaunchDarkly to disable the feature flag associated with the errors automatically. Flag triggers are available for Datadog, Dynatrace, Honeycomb, New Relic One, and SignalFX. LaunchDarkly also supports generic triggers that you can use with other tools.



Maintain visibility

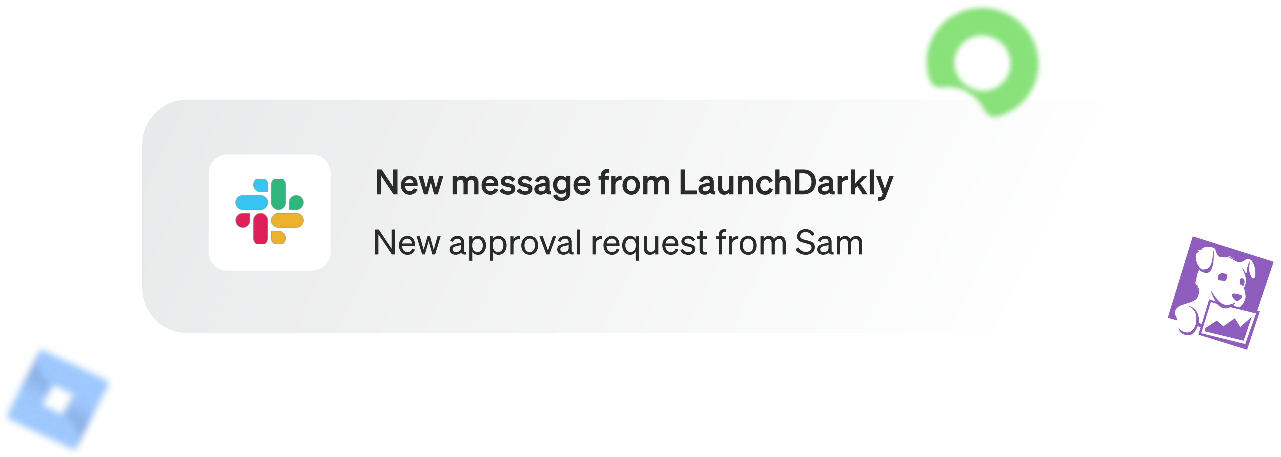
LaunchDarkly doesn't just give your developers the power to instantly deactivate broken code—it also allows you to maintain visibility and track flag changes alongside other events across your team's development tools. Unlike siloed solutions, LaunchDarkly unlocks the value of feature flags across your stack through integrations with Jira, ServiceNow, Slack, Microsoft Teams, and Datadog.

Here's an example of how this might look in practice. Sam, a developer, builds a feature and wraps it in a LaunchDarkly feature flag. She connects the flag to an existing Jira issue, thus giving the broader product delivery team visibility into what she's building for an upcoming release.

Using LaunchDarkly's Approvals feature, Sam also assigns a change approver among one of the engineering managers. The approver must review and approve Sam's feature before it can be deployed to production. When Sam creates this change approval request in LaunchDarkly, it automatically generates a standard change request in ServiceNow, thus ensuring Sam complies with company-wide change management policies. And the approver sees the request appear in Slack, where they can approve or deny it. Once approved (in Slack), the change gets deployed.

If Sam's feature causes a bug, Sam and the ops team can look at their Datadog dashboard to see that (a) error rates are surging, and (b) Sam had just turned a flag on. They can quickly correlate the error to Sam's feature and resolve the incident accordingly.

LaunchDarkly's many integrations with other DevOps tools enables greater visibility for everyone in product delivery. This increases speed and reduces risk.


New message from LaunchDarkly
New approval request from Sam

03

Spend more time on what matters

LaunchDarkly provides a scalable, reliable, and intuitive framework for enterprise feature management that empowers developers to spend more time on what matters.

Manage features globally

Serving 20 trillion flags daily, LaunchDarkly's enterprise-grade architecture delivers unrivaled speed, security, and reliability—enabling your developers to consistently and securely ship features anywhere, without sacrificing performance.

LaunchDarkly's architecture leverages server sent events (SSE) and streaming technology to deliver flag changes to endpoints within 200 milliseconds. Our resilience spans multiple layers, starting with a multi-region core service backed by over 100 points of presence globally. These layers, collectively known as our Flag Delivery Network, ensure that applications are always able to resolve flag values and evaluate rules even when connections are interrupted.

Supporting a feature management architecture like LaunchDarkly's would be costly in the extreme if done in-house. And it would consume precious developer time that ought to be spent building revenue-driving software.

Control and automate releases

Businesses must deliver targeted product experiences to compete. But building targeting capabilities in-house requires heavy development work. With LaunchDarkly, your developers can release features and changes to exactly who you want, when you want with advanced and secure targeting. And you don't have to bear the engineering burden.

Developers can leverage targeting to progressively release changes and automate feature entitlements, meanwhile business stakeholders (e.g., product managers) can easily run beta tests and deliver personalized product experiences. By letting LaunchDarkly handle targeting, your developers can get back to building and delivering the features customers demand.

With LaunchDarkly workflows, you can put even more time back into your developers' days by automating, delivering, and controlling software with speed, safety, and consistency. Schedule changes to your flag's targeting rules for a future date and time, mandate and manage approvals for changes to critical flags, and automate changes based on metrics from external tools. What's more, use workflow templates to build a consistent release process in LaunchDarkly that can be reused across all your teams and features.

Enable operational scale

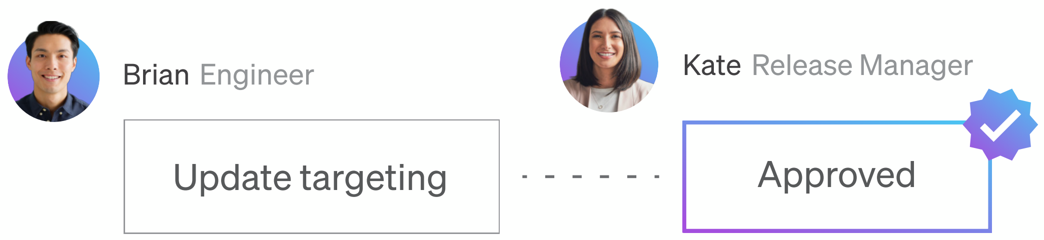
LaunchDarkly enables you to manage the feature lifecycle at every stage of development, from creation to archival, while also scaling consistent practices across all your development teams. These capabilities improve efficiency and save developers time in many ways.

For example, our approval workflows let developers move faster by decentralizing software change approvals. They also spare your change advisory board (CAB) from having to review every change by enabling peer reviews with the right safeguards in place. And they automate compliant release processes, so you can weave change controls into repeatable release templates and fix bugs in real-time while adhering to regulations.

Further, LaunchDarkly lets you advance platform standardization, so you can scale low-risk, efficient releases across your organization. It continuously delivers value by removing release bottlenecks, boosting developer productivity, and increasing customer satisfaction.

Your developers can also save time and avoid headaches by using code references to reduce technical debt and maintain impeccable code hygiene when deploying feature flags at scale. Automatically update flag references with every commit to GitHub, Bitbucket, and other Git tools to ensure that all relevant code is wrapped in the correct feature flags. And easily find and remove stale flags to keep your code base clean and consistent.

Ultimately, LaunchDarkly takes care of feature management, so your developers can devote their energy to projects that drive the business forward.



03

Customer success stories

LaunchDarkly solves the speed vs. risk conundrum by allowing your developers to test safely in production, instantly deactivate broken code, and spend more time on what matters.

Here are two examples of how our platform helps developers at leading organizations deploy more often and release without risk.

Paramount improves developer productivity 100x with LaunchDarkly

Challenge

Paramount's developer teams had bi-weekly dedicated release dates involving high inter-team dependencies. The 48 hours preceding these launches were characterized by stressful branch merges, QA regression tests, and last-minute fixes. And if a launch caused a bug in production, it took up to a week to fix.

Solution

Paramount used LaunchDarkly to ship changes on-demand across 14 applications. They reduced inter-team dependencies and now practice trunk-based development.

//

The ability to ship and merge code to environments safely without wincing every time we hit the 'deploy' button has been huge for us.

Dan Skaggs
Technical Director of Content Engineering at Paramount

Results

With LaunchDarkly, Paramount has achieved a 100x increase in developer productivity—from two deployments per month to 6-7 per day. What's more, the company can now resolve production issues in a single day, compared to the up to seven days it previously required.

Read: Full Paramount case study →

vodafone

Vodafone enhances digital experiences

Challenge

Vodafone combined feature updates into quarterly “big-bang” releases that were not standardized across teams. The company needed to reduce risks, optimize operational efficiency, and deliver a better customer experience.

Solution

Vodafone used LaunchDarkly to increase their deployment frequency and create a more stable customer experience.

Results

Vodafone now releases continuously and can resolve incidents immediately. The company releases around 363 times a month—compared to the previous cadence of once per quarter—and maintains a 98% success rate when deploying to production.

Read: Full Vodafone case study →

> Even though Vodafone has released a record amount of software changes, the company saw its highest availability this past year [with LaunchDarkly].

04

Get started

If you're ready to standardize fast, low-risk software development by decoupling deployment from release, instantly disabling broken code, and activating next-level user targeting, then request a customized, personally-guided demo.

Ready to get started?

Get a demo