

LaunchDarkly →

Take control over your mobile releases

How to deliver best-in-class mobile app experiences on your schedule, not the app store's.

Table of Contents

Introduction	2
01 Top 3 challenges in mobile app releases	4
Slow releases and mandatory app store approvals	
Unplanned downtime with no immediate path to resolution	
Poor user experiences	
02 How LaunchDarkly can help	8
Release on your own terms	
Mitigate bugs without shipping updates	
Improve in-app experiences	
03 Customer success stories	21
Climate LLC	
General Motors	
04 Get started	26

INTRODUCTION

Why are mobile app releases so challenging?

Today, there are 3.8 billion smartphone users in the world that spend roughly 4.2 hours per day inside a mobile app. And in 2023, mobile application revenue is expected to reach \$935 billion.¹ Mobile applications offer a multitude of benefits to organizations, ranging from boosting brand awareness to enabling personalized experiences—all of which drive higher user engagement, brand loyalty, and revenue.

However, mobile developers face unique challenges that limit their control over their own software applications. Unlike with standard software delivery, when mobile development teams are ready to deploy the latest version of their app, they must first submit it to the app store for review. This and other challenges inherent in mobile development lead to:

- Minimal control over feature release timing and targeting
- Inability to easily validate new features in-app
- Lack of real-time error resolution
- Poor user engagement and low feature adoption
- Frustrating, glitchy user experiences



LaunchDarkly empowers developers to take back control over their mobile applications and deliver best-in-class mobile experiences. In this guide, we will explain how developers can confidently deliver new features to end-users on their schedule instead of solely relying on app store approval windows or end-users updating to the latest app version. We'll also show how LaunchDarkly enables in-app personalization, helps validate and test new features, and promotes real-time error recovery.

Who is this content for?

This guide is intended for mobile app software developers and leaders, as well as product managers and leaders responsible for coordinating mobile releases and enhancing the mobile user experience.

¹ "55+ Jaw Dropping App Usage Statistics in 2023," Tech Jury, July 2023.



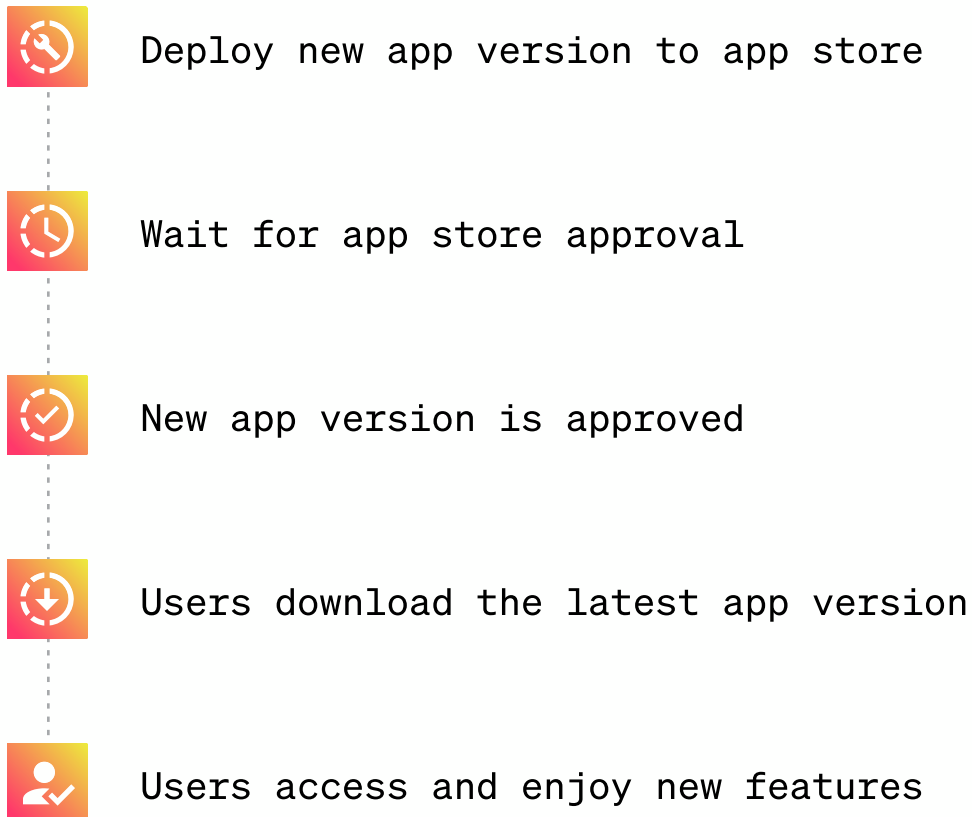
Top 3 challenges in mobile app releases

Mobile app developers face a host of challenges, from cumbersome app store approvals to the stress of pushing out a hotfix through the app store and consistently needing to meet high user expectations. Let's examine these challenges.

01

Slow releases and mandatory app store approvals

App stores were initially designed to help provide a centralized, consistent platform for users to discover, download, and update their apps. And while app stores help ensure organizations follow guidelines to provide safe and consistent user experiences, they also deprive organizations of full control over their own apps. Due to app store approval processes, developers lose the ability to control their release timing, test new features in-app before releasing to end-users, and perform gradual rollouts. Developers are gated from releasing new app versions as they wish.



02

Unplanned downtime with no immediate path to resolution

What happens when you ship code that causes app degradation, or worse, an outage? With app store approval processes, there's no quick way to address underlying app issues, roll back to a previous app version, or easily identify which user segments are impacted by a buggy feature. Mobile app developers must go through multiple steps to resolve the incident: identify the issue, build a fix, submit the resulting binary back to the app store for an expedited review, etc. All of this can take days to complete, causing developers undue stress. And with user expectations being so high, any amount of app unavailability can lead to user churn and lost revenue.

Costs of poor app availability:

- #1 Lost revenue and transactions
- #2 User churn and uninstalls
- #3 Developers' and support professionals' time
- #4 Reputation and negative app reviews



03

Poor user experiences

Unless you deliver personalized mobile app experiences, you will likely fail to achieve adequate levels of user engagement over the long-term. But delivering targeted experiences to the right users across varied device types, app versions, operating systems, etc. is challenging. It requires a sophisticated targeting engine.

Secondly, syncing mobile and web releases (e.g., ensuring the same features appear on your mobile and web applications at the same time) is also a challenge. Typically, it involves a lot of manual and duplicative work when it's time to release a new feature across all platforms.

Finally, mobile developers often lack the ability to run experiments on new features in production. This prevents them from gathering critical engagement data, which, in turn, makes it hard to know which feature variations perform the best. The inability to effectively deliver targeted experiences, ensure consistency across platforms, and run mobile experiments with real users leads to a poor digital experience in the end.

58%

of smartphone users feel more favorable toward companies whose apps remember who they are and their past behavior.²

² "App Personalization," [Business of Apps, July 2023](#).



How LaunchDarkly can help

LaunchDarkly empowers mobile app teams to release features in real-time, making it possible to deliver on their schedule, roll back changes faster, and provide superior experiences across OS versions, devices, screen sizes, locations, and more.

Jump straight to a benefit section:

- #1 [Release on your own terms](#)
- #2 [Mitigate bugs without shipping updates](#)
- #3 [Improve in-app experiences](#)

Feature flags

[Feature flags](#), also known as feature toggles, are used in software development to enable or disable a feature without modifying the source code or requiring a redeploy.

For mobile applications, this means developers can toggle features on or off without a redeploy to the app store. Simply wrap the code in a feature flag, deploy the new feature to the app store for approval, and then, once the new version is approved and users have installed it on their device, a developer can release (or roll back) features at will.

With feature flags, mobile development teams can decouple app store deployments from feature releases to users—thus enabling them to manage the full lifecycle of a feature.





Schedule

Mon, Apr 12 2023



Android update

01

Release on your own terms

App store approval windows prevent teams from delivering mobile app features at the pace or time they need. Furthermore, once app versions are approved by the app store and made available for installation, developers often notice an adoption lag before a critical mass of users install the new app version.

With LaunchDarkly, you can release new capabilities to your users on your schedule, without reliance on app store approval windows or end-users updating to the latest app version.

Deliver on your schedule

Gone are the days when mobile developers lacked control over the timing of their releases. Instead, with feature flags, mobile developers can decouple the process of deploying a feature to the app store from releasing that same feature to end-users. By wrapping code with feature flags, you can keep features that are not ready for end-users toggled “off”. That way, once the app is approved by the app store and made available to download or update, users will not see any features that are toggled “off”.



This means you can strategically choose when to release features based on things like:

- ✓ **New app version adoption**
Release features once 90% of your userbase is using the latest app version.
- ✓ **A larger marketing campaign**
Coordinate marketing efforts and ensure the new marketing campaign is released on your mobile app at the same time it's launched on your website.
- ✓ **User behavior or seasonality**
Release features based on seasonal trends, holidays, and user behavior patterns to boost engagement.

By leveraging feature flags, mobile developers can also employ common development strategies such as:

- 🔘 **Progressive delivery**
- 🔘 **Trunk-based development**
- 🔘 **Feature branching**
- 🔘 **Canary deployments**
- 🔘 **Blue-green deployments**



Test safely and efficiently

Testing new mobile features and app versions poses unique challenges relative to other software testing strategies—mainly due to the highly disparate mobile ecosystem and wide range of possible end-user configurations (i.e., different device models, form factors, operating systems, app versions, etc.). Mobile apps also often utilize device-specific capabilities that vary from model to model like microphones, location services, biometric authentication, and cameras that can make testing and simulating real-world scenarios in a pre-production environment challenging.

With LaunchDarkly, mobile developers can [test features in a production environment](#) without exposing them to end-users—thus reducing the need to test in a staging environment. Instead, it's as simple as deploying new features to your app and only exposing them to a subset of internal users for testing. By testing in production, mobile developers can see how a feature performs in a real-world setting while limiting the impact on end-users and other app functionality. This also enables you to more comprehensively test new features, identify bugs, and fix them before a release.

Release progressively

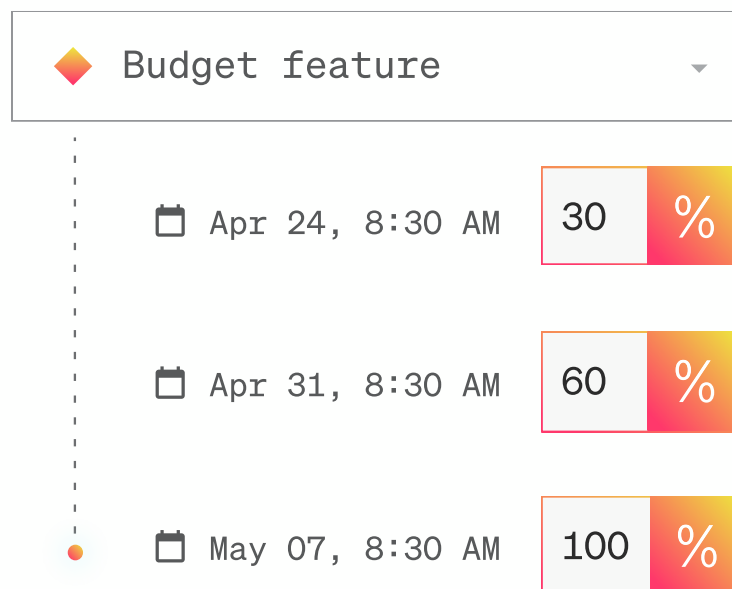
Now that you've internally tested your latest features, use LaunchDarkly to perform gradual rollouts rather than pushing out every new feature to every device all at once. With [feature flag targeting](#), mobile developers can share and test new features with beta customers in a production environment to collect feedback, all while hiding those new features from general end-users.

Additionally, mobile developers can conduct and automate targeted, [percentage-based rollouts](#) to reduce their blast radius and the risk of a “big-bang” launch. Instead, you can even leverage [granular targeting](#) to define highly-specific audiences for progressive rollouts. For example, you can



choose to progressively release a new feature in 5% increments every hour to users only located in North America on an iPhone device with your latest app version. This rollout will continue automatically unless you manually stop it or the rollout reaches 100%. This type of rollout mitigates risk by enabling you to start small, monitor feature performance, and assess the overall impact a feature has on an app as traffic increases.

Progressive rollout



02







Mitigate bugs without code changes or app store approvals

We know mobile app users have a low tolerance for slow and buggy mobile apps. To maximize user adoption and minimize churn, your app must meet—or ideally, exceed—users' expectations, no matter where they are.

Resolve errors in real-time

Bugs are inevitable. Thankfully, LaunchDarkly provides an immediate path to resolution through [kill switches](#).

Without a kill switch, when something goes wrong in an app, there's no quick or easy fix to minimize customer impact. Instead, mobile development teams must:

-  Identify the buggy code
-  Implement a fix
-  Validate the fix and run the app's full test suite to check for regressions
-  Resubmit the fix to the app store
-  Await app store approval—this can take hours to days
-  Wait for users to update to the latest version to receive the fix



Mobile developers are all too familiar with the time, money, stress, and opportunity loss a bug in production can cost. As they work tirelessly to deploy a fix, their app misses out on potential revenue, frustrates customers, spikes support tickets, and more.

But, with a kill switch, you can simply toggle off the feature causing app degradation without redeploying code and waiting for app store change approvals. This minimizes customer-impacting bugs and unplanned downtime, and it provides a safety net for developers. Once the buggy feature is toggled off in real-time and no longer impacts users, mobile developers have the time they need to identify the bug, build a fix, and resubmit through the app store. What's more, they don't have to stress about losing revenue and users with each passing minute.

62%

of people uninstall an app if they experience crashes, freezes, or other major errors³



New feature



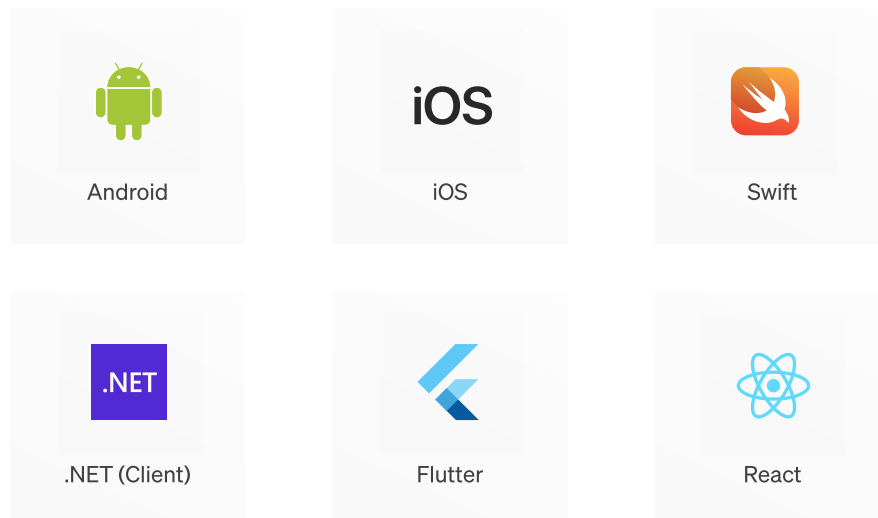
³ ["40 Fascinating Mobile App Industry Statistics \[2023\]," Zippia, March 2023.](#)



Deliver reliably at scale

Ensure consistent experiences globally across millions of users and deliver real-time flag changes without sacrificing your app's response time, battery life, or overall performance. LaunchDarkly's architecture was built to ensure availability. Through our [Flag Delivery Network](#), our self-healing SDKs leverage streaming technology to retrieve and cache feature flag values in memory for quicker evaluation. And while flag changes are delivered in real-time to your mobile app users that have your app foregrounded, once the app is backgrounded, LaunchDarkly will disconnect to save on device battery power. But not to fear—once your app is backgrounded on a user's device, the flag values will persist so that when your app launches again, it will have retained the previous flag values until establishing a new connection.

LaunchDarkly Mobile SDKs

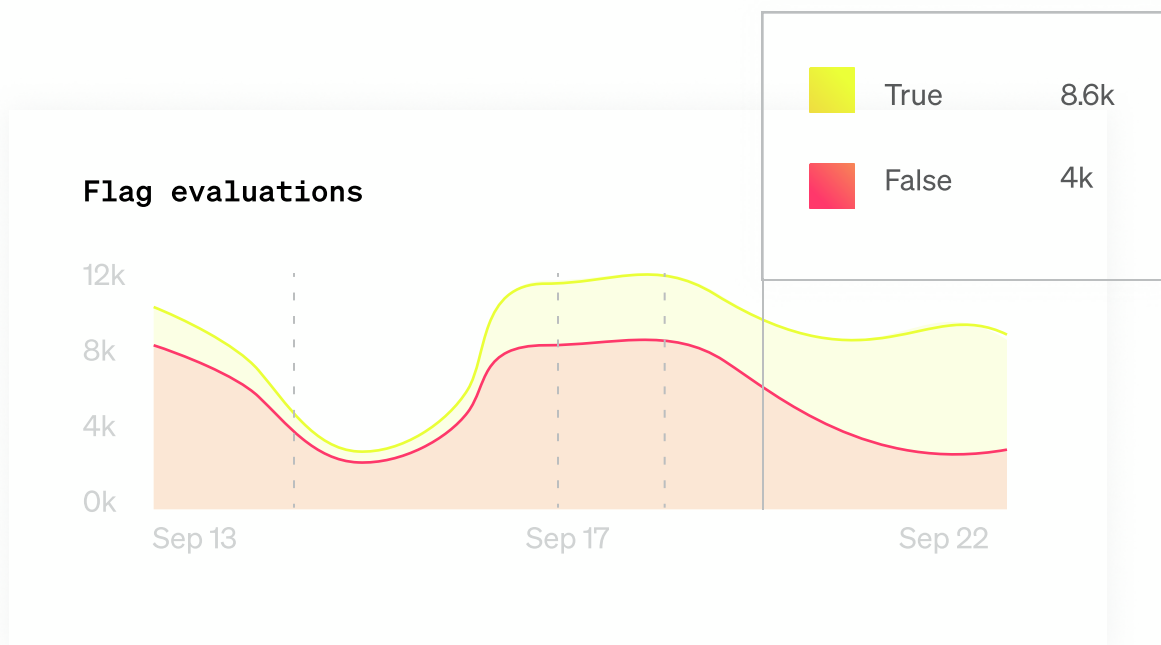


Monitor feature performance

Track flag evaluations to make data-driven decisions with LaunchDarkly. Quickly visualize the amount of flag evaluations with [flag insights](#) and understand how changes to targeting rules or increased traffic impact overall feature consumption. It's now easy to quickly validate what percentage of mobile users are experiencing your latest feature!

Additionally, with LaunchDarkly, it's simple to audit changes to feature flags. Perhaps you see a sudden decrease in evaluations and you want to know why. Simply review each flag's complete audit log to clearly understand when the flag was changed, who made the change, and how the change impacted the end-user experience. And with LaunchDarkly [live events](#), quickly validate that flags are being evaluated as intended based on targeting and incoming device data.

Together, with flag insights, the audit log, and live events, ensure with confidence that a rollout is going as planned. And quickly surface unexpected user behavior for real-time remediation to optimize app reliability and user satisfaction.



03

Improve in-app experiences

Delivering personalized user experiences across a variety of app versions, devices, and screen sizes is complex. With LaunchDarkly, easily target based on any parameter, sync features across mobile and web platforms, and ensure you are always delivering the best features to your end-users.

Deliver advanced personalization

In a time when mobile user expectations are at an all-time high, it's critical to have the tools needed to provide targeted, personalized experiences. With LaunchDarkly's [robust targeting capabilities](#), mobile development teams can boost user engagement by providing personalized experiences across your userbase.

In fact, you can deliver targeted experiences based on anything you know about your end-user audience—and yes, we mean *anything*. Target features by device type, operating system, application version, location, user tier, etc. to get the right experiences to the right audiences, faster. LaunchDarkly's targeting capabilities are both robust and easy to use. Even non-developers can build and manage targeting rules in LaunchDarkly (with the right user permissions and approvals in place, of course).

71%

of app users will churn
90 days after download⁴

Sync experiences across platforms

Deliver consistent user experiences across different web and mobile platforms (iOS, Android, and Fire OS) with coordinated feature releases.

Let's assume within your LaunchDarkly instance you have multiple teams using multiple environments. By implementing [hierarchy rules](#) across your feature flags, it's possible to now manage a set of associated flags together. This not only ensures that your app's code doesn't break due to features being toggled on prematurely, but it also allows organizations to launch new features or experiences simultaneously across each of their environments.

For example, let's say you are launching a new UI for both your mobile app and web portal with multiple sub-features. Without LaunchDarkly, it's difficult to simultaneously release your new UI across both mobile and web. This is mainly due to the lack of control over the timing of a mobile release due to app store review cycles. But, on top of that, without associated flags and hierarchy rules, both the web and mobile teams would need to go through duplicative, simultaneous release motions to ensure everything is deployed at the same time and in sync.

LaunchDarkly removes the duplicative work. Once all the new features are tested and deployed for both mobile and web, you can toggle on a single overarching UI launch prerequisite flag to release the new features to all users across all platforms at the same time.

Prerequisite flags

Each time a new flag is created, there's an option to make it dependent on another flag. This ensures that flags that depend on each other to function properly won't accidentally get turned on separately and cause code to break. For example, if Flag A is a prerequisite for Flag B, Flag A must be toggled "on" for Flag B to serve evaluations.

Optimize the user experience

Run experiments with real end-users in production to make data-driven decisions. Experiments can help validate new ideas, determine interest in a new feature, increase product adoption, and most importantly, drive revenue and customer engagement.

With [LaunchDarkly Experimentation](#), seamlessly measure the impact of product changes, understand which feature variations boost user engagement, and enable your mobile teams to deliver the right experiences to the right users. This data-driven approach shortens user feedback loops and ensures winning feature variations are being served. On top of that, LaunchDarkly Experimentation is built directly within the development workflow, meaning once an experiment has ended, mobile teams can simply toggle on the winning variation for all users in real-time—no having to go back and code a winning variation based on an experiment run in a visual editor tool.

Furthermore, ensure users on old legacy app versions receive your latest features and experiences by ensuring they are on the latest app version. With LaunchDarkly, you can gracefully degrade the user experience however you wish. For example, you could disable functionality dependent on APIs containing the breaking changes. Or you could be more forceful and present an in-app notification telling customers they must update to a newer app version. Either way, this ensures when you launch your latest winning features, they will be available to all app users.

⁴ [“40 Fascinating Mobile App Industry Statistics \[2023\],” Zippia, March 2023.](#)



Customer success stories

LaunchDarkly empowers mobile teams to take back control over their apps, decouple releases from the app store, and deliver the right in-app experiences to customers exactly when they need them.

Here are two examples of how our platform helps mobile developers at leading organizations release mobile features and fixes on their schedule.



Climate LLC reaps the rewards of real-time app updates

[Climate LLC](#), the digital farming arm of Bayer Crop Science, and their app platform FieldView™ help farmers across the world collect, store, and monitor agricultural data. It's critical for FieldView™ to maintain consistent and reliable uptime during harvest season.

Challenge

Climate LLC was leveraging a homegrown feature flagging tool to toggle features all the way on and off. As they scaled to over 220 million subscribed acres worldwide, they needed a tool that could scale with them and conduct progressive rollouts, push real-time flag updates, and target features to enable personalized experiences.

Solution

With LaunchDarkly, Climate LLC has full control over the timing of their mobile releases, executes gradual rollouts, and instantly rolls mobile features back in real-time if something goes wrong. They leverage targeting to not only create unique experiences based on farmers' locations around the world, but also to test features in production with internal users before rolling out to everyone.



Results

Climate LLC deploys app updates with confidence and has significantly reduced the burden on developers. With LaunchDarkly, they can now deploy updates in real-time. And if something does go awry, they can use LaunchDarkly to simply roll features back instantly without having to push an update through the app store. This mitigates the impact on their end-users and ensures FieldView™ is always up and running when farmers need it most.



Read: [Bayer's digital farming arm improves reliability](#)



//

During the planting and harvest seasons, we sometimes need to ship changes to the Cab App [mobile app] at a faster pace than is ideal. In the past, we'd have to run these changes through a rigorous approval process before going live. It was both slow and stressful. But now our attitude is: We trust what we ship, but we always have a backup plan with LaunchDarkly.

Artie Lee

Director of Engineering, Climate LLC





General Motors deploys at turbo speed

As the demand and competition for technologically-advanced cars continue to accelerate, [General Motors](#) (GM) has sought ways to differentiate itself. The company currently offers a handful of customer-facing mobile applications that include features like remote start, vehicle monitoring, personalized owner resources, vehicle safety, and emergency services.

Challenge

Prior to LaunchDarkly, GM conducted “big-bang” releases. These mobile releases were challenging as they consisted of long development cycles and large code changes all at once. This caused unstable test environments and long triage sessions when something went wrong. On top of that, GM struggled with the complexities of managing multiple apps across various device types, OS and app versions, vehicle makes and models, and subscription tiers.

Solution

GM now utilizes LaunchDarkly to deliver value to customers quicker than ever. They utilize feature flags both defensively, to roll back negative-impacting mobile features in-real time, and offensively, to understand flag evaluations,



conduct testing, and learn from their audience. LaunchDarkly also enables GM to more adequately test across their many app permutations by gradually rolling out features to a subset of users before releasing them to a larger audience.

Results

With LaunchDarkly, GM has reduced its development cycle time and strives for continuous deployment by separating deployments from releases. They are minimizing unplanned downtime, reducing user-impacting bugs, and delivering excellent in-app user experiences.



Read: [How General Motors Leverages Feature Flags](#)



//

We found a number of issues that, in the past, we would've never found until our customers got their hands on it. So, instead of using my customer to find issues, I'm able to do it internally.

Jim DeMercurio

Director of Mobile Solutions, GM



Get started

With LaunchDarkly, organizations can take back control over their mobile apps, enhance the user experience, and successfully deliver an engaging, reliable app. By decoupling feature releases from the app store, development teams can validate features in-app, roll back features in real-time, deliver personalized experiences, and more.

If you're ready to take control over your mobile app by releasing new features on your schedule, mitigating risk through real-time rollbacks that don't require shipping a fix to the app store, and improving your overall user experience and engagement, then request a customized, personally-guided demo.

[Get a demo](#)

