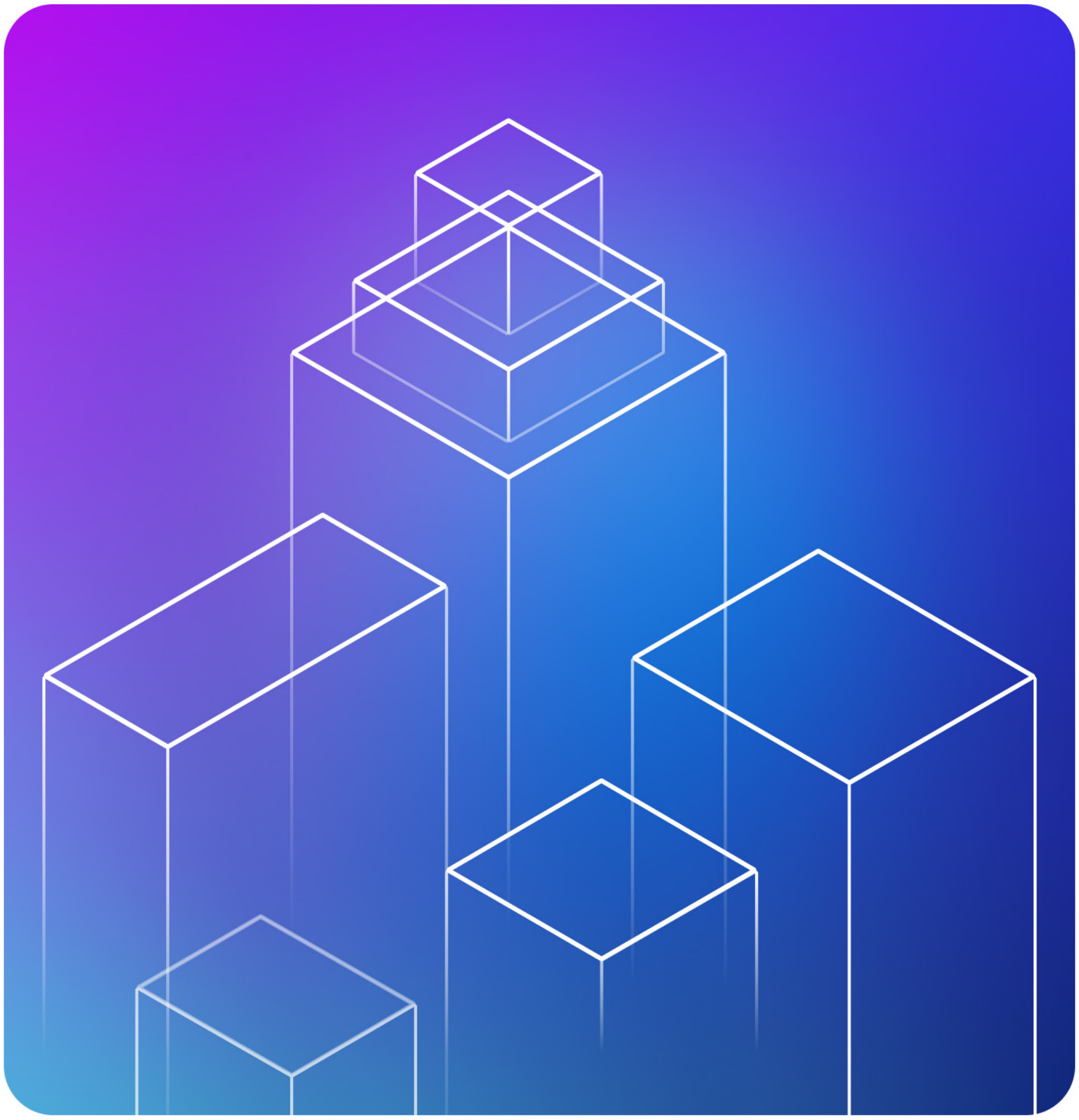
ionic

eBook



Micro Frontends for Mobile: The Benefits for Enterprise Development

ionic

eBook

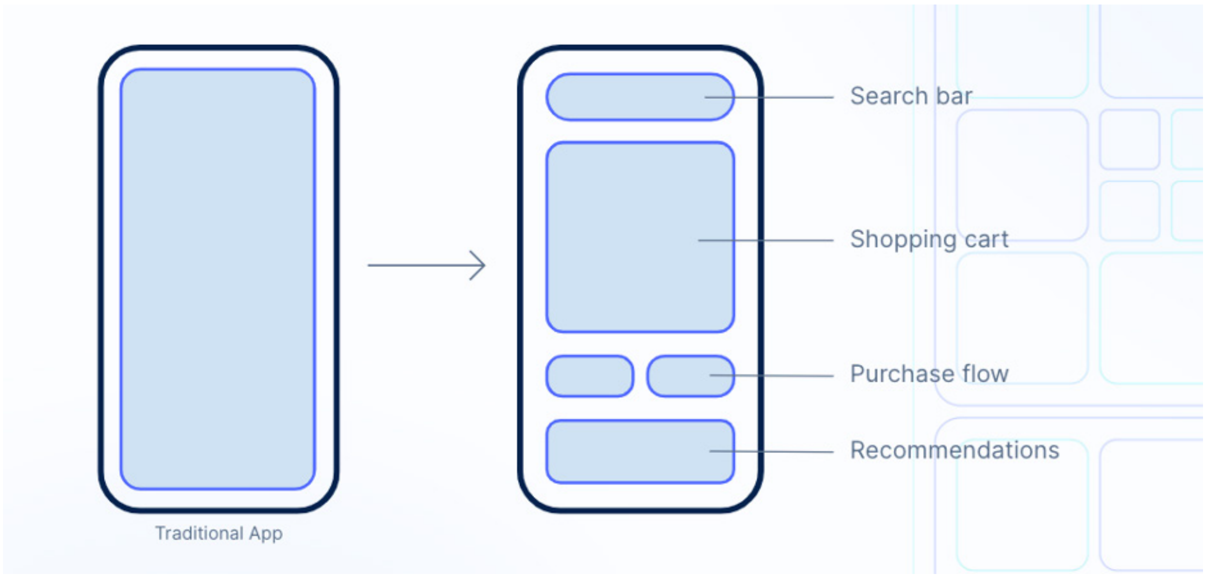
CONTENTS

INTRODUCTION

The Benefits of Enterprise Development

It’s been a little more than a decade since microservices were introduced as a modular, flexible alternative to the bulky monoliths used to support enterprise software architecture. Today, at least 81 percent of businesses now use microservices within their organization.

Microservices or a microservice architecture breaks down the backend of applications into independent functions. This enables various development teams to build the app simultaneously, update the app functions without disrupting the other app functions, and ultimately scale production by having multiple teams build, test, and ship separate services in parallel.



The proliferation of microservices is bringing this concept to the frontend experience in the form of micro frontends. When implemented through the right development platform, micro frontend architecture can reduce the time and resource cost of web and mobile app development for your organization—all while supporting a more flexible and engaging user experience. In this guide, we’ll offer a deep dive into the business benefits and development opportunities created when enterprise organizations embrace micro frontend development.

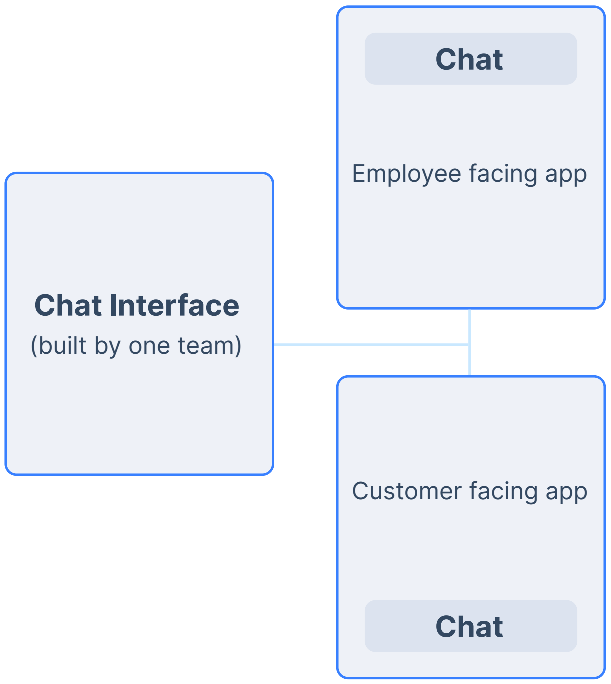
You’ll also get a first-hand look at how Ionic Portals can turn the promise of micro frontends into reality for your enterprise native mobile applications.

Micro Frontends, Defined

Excited about the potential value of micro frontends, but confused about what they actually are? Don't worry—we're here to help.

Micro frontends take the monolithic application architecture delivering your entire frontend experience from a single codebase, and divide that monolith into many different features. These features can then be built, updated, and deployed in parallel—and reused across multiple applications.

For example, a chat interface might be built and deployed as a micro frontend that lives in a dozen or more separate applications. This chat interface would be built and maintained by a single team whose single focus is on the chat experience. When you build a new application that needs a chat feature, you simply incorporate this micro frontend experience—no need to build an entirely new chat solution for that individual app.



This approach to frontend development offers a number of benefits, including:

- Faster development of new applications, by reusing existing functionality.
- Easier scaling of large projects by allowing multiple teams to work in parallel.
- Improved UX by having each team focus on doing one thing well.

The current micro frontend model borrows concepts from past innovations on the enterprise frontend—including Self-Contained Systems and Frontend Integration for Verticalized Systems—and brings together these approaches while unifying frontend and backend applications through a mediated API layer.

Historically, micro frontend architectures have been very popular with modern web applications, where it's relatively straightforward and easy to enable this kind of approach.

However, doing this in a native mobile app is not so easy. We'll get to that shortly.

Key Characteristics of a Micro Frontend

Don’t let your business get fooled into creating a new frontend that only looks like a micro frontend.

As you research development platforms that deliver micro frontend capabilities, you need to be certain that each developer’s resulting applications will be able to deliver a true micro frontend experience. You don’t want to invest time and resources into this transformation project to then discover you’ve built a clunky low-code knockoff that walks and talks like a micro frontend—but lacks the true flexibility and composability your business needs.

By definition, any micro frontend or microapp should deliver the following capabilities:

1. **Reusability and portability.** The microapps within a micro frontend should offer plug-and-play capabilities, making it easy for those components to be utilized across multiple applications, or to be seamlessly integrated with new applications without disrupting the larger app ecosystem.

2. **The ability to communicate with other components.** Microapp tooling provides a container that enables communication with multiple components at once—enabling clear and separate lines of communication.

3. **Seamless integration that makes these connections virtually invisible to users.** An app built with micro frontends may be composed of many smaller, independent parts, but it shouldn’t appear that way to the user. Any good micro frontend will be able to deliver a seamless user interface that can seamlessly accommodate many different user journeys while providing a consistent user experience.

4. **Team-based ownership of micro applications.** One of the biggest architectural shifts enabled by micro frontends is the ability to empower development teams to take control of the app or apps within their domain. This team-based approach makes it easier to isolate code between applications, supporting portability, and it enables those teams to support individual experiences through optimized application technology and design. From the example above, having a single team own the chat experience allows them to focus and improve by doing one thing well.

The Business Benefits of Using Micro Frontends for Mobile

While certain benefits of micro frontends are universal to any enterprise embracing this development approach, it's important to align any micro frontend development project with specific outcomes your business is seeking.

When Luca Mezzalira, then a VP of Architecture at DAZN, first sought out a micro frontend approach for his organization, his transformation project was focused on addressing a specific area of need.

When I started this journey, the problem I had is, how can I scale a single-page application with hundreds of developers distributed in Europe?" Mezzalira explained in a [2020 presentation at QCon.]() "Unfortunately, I didn't have an answer. What I thought is, let's start to look at a successful architecture pattern that allows me to do that."

- Luca Mezzalira, VP of Architecture at DAZN

The alignment of development goals with your micro frontend project will help your organization accelerate and optimize the ROI of this new approach. That said, micro frontend development for mobile can deliver the following business benefits:

- **Ability to scale development by working in parallel.** Large frontends make it difficult—if not impossible—to manage multiple development projects over time. This creates a bottleneck of development projects that must be prioritized over one another. By working in parallel, development teams are able to work simultaneously across multiple projects contained within apps, resulting in faster app updates and development cycles.

- **Reduced time and resource cost for mobile app development.** Remember the scenario we laid out regarding the frontend chat interface? By repeating the use of this single interface multiple times across multiple applications, you're able to implement new capabilities faster, and with less development labor.

- **Greater consistency and continuity across application touchpoints.** Although your micro frontend is composed of many different moving parts, the shared interfaces and capabilities—along with a seamless frontend design—will actually give your end users greater consistency as they move across digital touch points and modalities than a monolithic frontend. Micro frontend architecture incorporates new changes faster—and keeps your frontend experience responsive to your users' evolving needs.

Modular apps and responsive features make it easier to customize each user's experience to their specific needs or interests. As new micro journeys are discovered and prioritized, this modular design makes it easy to build out new features and pathways to serve those divergent paths.

Building Apps With Micro Frontends: A Cost-Benefit Analysis

Before your organization can define the business use case for micro frontend development, a cost-benefit analysis is necessary to identify and quantify the costs and benefits relevant to any conversation around development ROI.

While the proportions of this cost-benefit analysis will be case-specific to every organization, the top drivers of, and barriers to, micro frontend development will be familiar to most companies. Emerging research and enterprise surveys regarding micro frontend adoption have identified the following categories as being most influential on enterprise cost-benefit analyses:

Cost Drivers

- **Frontend Growth.** The larger and more complex frontends become, the greater their associated maintenance costs. Businesses can also incur increased costs through the inflexibility and scaling difficulties these monoliths create, which can slow down or disrupt revenue-generating activities—and which may require the adoption of additional solutions to supplement these limited capabilities.

- **A Large, Inflexible Codebase.** As monolithic frontends grow, their dependence on a single codebase becomes a point of friction for developers. These architectures inevitably feature a large and expanding number of dependencies, and working around these dependencies requires more development, testing, and integration time.

  With smaller codebases offered by a micro frontend, developers can be more agile and efficient when building out new capabilities and integrating applications. The resulting frontend is much easier to scale and requires less maintenance, reducing ongoing costs while reducing barriers to revenue generation.

- **Application Interdependence.** In a large, single codebase, monolithic frontends become bloated with applications that are dependent on one another, for better or worse—all too often, worse. A single application malfunction can create a ripple effect of downtime and disruption across all applications within that codebase. Even if you're fortunate to avoid these disruptions, replacing outdated applications with newer, more valuable solutions requires the equivalent of codebase surgery—and a successful outcome is never guaranteed.

  While this interdependence stifles the adoption of newer, better solutions, it can also leave your frontend hitched to costly, inefficient applications that are delivering little value to your organization. This creates an opportunity cost, and an organizational tech debt, that will only grow over time.

- **Speed of Delivery.** Monolithic frontends inevitably require slower, more time- and resource-intensive updates and integration projects. With a micro frontend, the smaller codebase and application independence accelerates the speed and reliability of application delivery. This streamlined, modular approach eliminates one of the most unpredictable and risky cost considerations of updating frontend monoliths.

Benefits

- **Long-Term Flexibility and Support for New Technologies.** Both your business technology needs, and the enterprise technologies available on the market, are certain to change over time. This change can occur rapidly, and in unpredictable ways—and for monolithic frontends, keeping up with this change can be complex and riddled with speed bumps.

  Thanks to the smaller codebases and modular microapps making up your applications, adapting to these changes and adopting new technologies is both faster and less labor-intensive for your developers. Changes can be made without any impact to other applications within your frontend, helping your business and its frontend architecture keep pace with up-and-coming innovations.

- **Autonomous, Team-Based Development.** Large codebases in monolithic frontends often leave development teams with their hands tied. This is because the individual applications within a traditional frontend can’t be worked on or updated without impacting the rest of the codebase.

  Instead of putting those development teams through multiple layers of approvals and tedious development projects, these smaller, self-contained microapps give individual teams full ownership of all changes and updates made within their development domain.

- **Reduced Maintenance Demands for Frontend Applications.** With self-contained microapps and smaller codebases, developers spend less time maintaining your frontend—which frees up time and resources to enhance the capabilities and overall experience your frontend offers your end users.

- **Faster, More Reliable Application Testing.** Testing is an endless headache in monolithic headframes, which require a test to occur across the entire application—even when the changes being tested are relatively small and simple. While this testing is essential to avoid bigger problems and bugs from being introduced, it creates friction between developers implementing changes and engineering leaders eager to avoid disruptions.

  Micro frontend applications alleviate this problem by making it easy to isolate testing to a single self-contained microapp. Even when testing turns up problems, the impact of this issue is limited to that microapp alone, preserving the rest of your frontend.

- **Unrivaled Scalability.** Development teams can move faster on new projects and capabilities. Modular application design makes it easy to add new features or grow operations by having multiple teams and potentially hundreds of developers working in parallel. Small codebases eliminate the hassle of integrating new capabilities and complexities into an already massive frontend architecture. All of these alleviated pain points pave the way for a fully scalable micro frontend supporting the needs of your organization and your end users.

Who Benefits from Adopting Micro Frontends?

Micro frontends offer long-term cost benefits and performance capabilities that serve many aspects of both your backend operations and your end-user experience. By aligning micro frontend migration with desired business outcomes for the full spectrum of stakeholders directly impacted by this approach, your business can build an even stronger use case and improve ROI forecasts for this new application architecture.

Here's a look at the central stakeholders and perspectives served by a micro frontend application.

App Developers

Micro frontend applications are a game-changer for app development teams, which directly benefit from reduced maintenance demands, faster and more efficient development cycles, and reduced complexity when implementing changes.

By creating containers to confine codebases and isolate micro applications from one another, app developers can take a more iterative approach. This enables continuous delivery from developer teams, as well as an emphasis on optimized design and micro journey experiences.

Engineering Leaders

Software engineers are responsible for coordinating the integration of individual microapps within a single user experience. In a monolithic front-end, engineering leaders are tasked with overseeing development teams to make sure development, testing and other processes aren't causing disruptions across the application—a time-consuming task that takes them away from more value-added responsibilities.

With a micro frontend, engineering leaders can shift their activities away from development team oversight and focus on collaborative processes such as communication across teams and applications, as well as reuse of microapps across the frontend itself. Greater priority and attention can be given to maximizing efficiencies of the micro frontend itself, and to delivering the best user experience possible.

Enterprise Executives

C-suite and other enterprise leaders will directly benefit from the long-term cost savings and efficiencies that micro frontends can create for the IT department. Optimized spending and expense reduction—including maintenance costs and app development costs—will increase ROI over the life of the micro frontend.

Meanwhile, the agile and scalable design of the micro frontend means that frontend design and capabilities won't become a constraint as the business pursues its growth goals. As leaders plan out and execute their business strategy, they can be confident that their frontend application will be an asset in reaching those goals.

End Users & Customers

Consistent, user-friendly design is a crucial element of successful end user engagement. While the micro frontend breaks this application into many different parts, this modular design enables a composable architecture to deliver a consistent, seamless user experience.

Microapps are capable of delivering a multi-experience user journey that defines and serves the micro journeys taking place within that larger experience. This benefit requires development teams to have a deep understanding of customer journeys and the personas and paths being served by the micro frontend. However, when executed properly, the end result is a far more responsive, engaging, and individualized customer experience.

Examples of Enterprise Apps Built With Micro Frontends

Although you probably can't tell based on the end-user experience alone—and that's a good thing, by the way—a number of popular enterprise applications are currently built with micro frontends.

By adopting a micro frontend, these enterprise apps are able to seamlessly integrate web and native content, streamline and simplify the user experience, and make it easier to manage and update these applications over time.

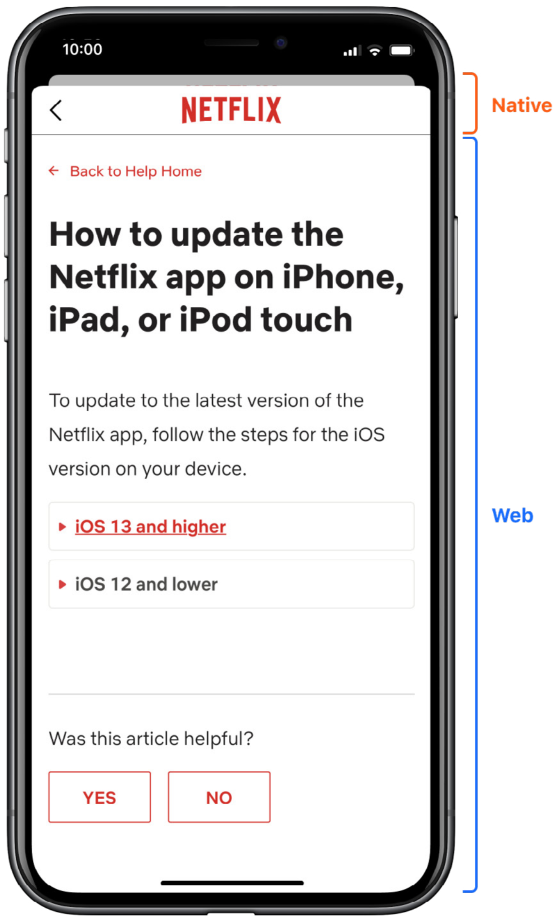
Here's a look at how micro frontends have laid the foundation for modular, responsive and scalable app experiences for Netflix, Apple Music, and Amazon.

Netflix

The Netflix player and navigation tools are all native to the app on each user's device—which is important to deliver a consistent experience across so many application versions spanning mobile, desktop and smart TV devices. But the Netflix library of offerings can change based on, most notably, the location where you're watching, which is why access to a web-based library is important to ensure accurate content availability for each viewer.

Similarly, all content supplemental to the app-based experience is web-based and constantly changing as the app itself grows and evolves. User guides, help content, FAQs and other important information is hosted on the Netflix website and delivered to app users via WebViews.

This makes it easy to make a single update that proliferates across all instances of the app: if new help content is needed for Netflix users, for example, developers only need to add or update a single website page, and can then make that new content immediately available across all apps without forcing an app update.



← Back to Help Home

How to update the Netflix app on iPhone, iPad, or iPod touch

To update to the latest version of the Netflix app, follow the steps for the iOS version on your device.

▸ iOS 13 and higher

▸ iOS 12 and lower

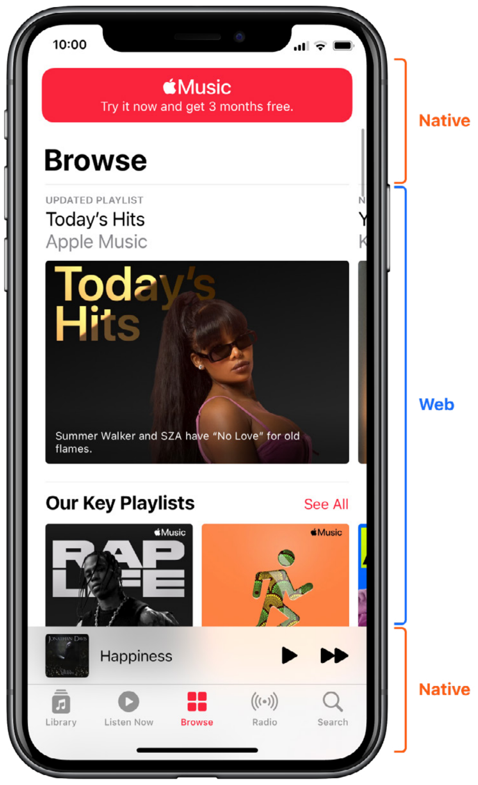
Was this article helpful?

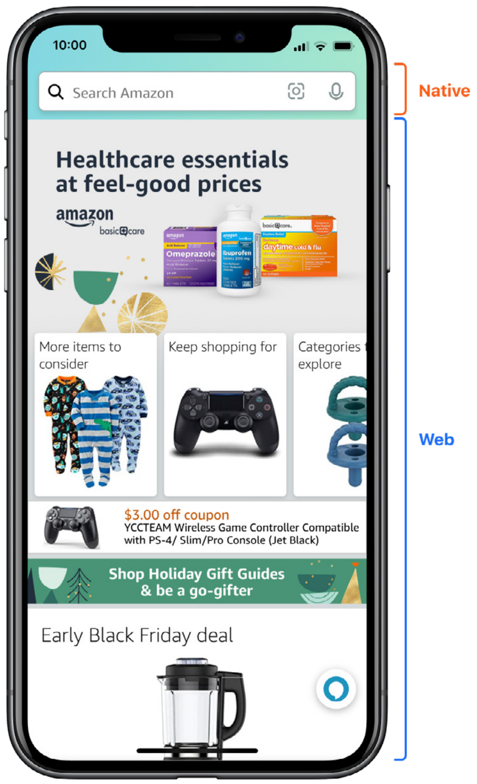
YES NO

Apple Music

Where Netflix's user experience is primarily powered by native app design, Apple Music is more balanced in its integration of native and web elements. Its native app navigation features easy-to-use tabs for listening, browsing, searching, playing and discovering all audio content available through the app.

Meanwhile, all of Apple Music's playlist cards, web ads and other visual content are web-based. As with Netflix, this gives developers the control they need to quickly change the content being promoted across all applications. If a new album is released or a new song rises to the top of the charts, the web-based ad spots can be changed to highlight these new songs and artists, and/or related music categories app users may be eager to discover.




10:00
Search Amazon
Healthcare essentials at feel-good prices
amazon basic care
More items to consider
Keep shopping for
Categories to explore
$3.00 off coupon
YCCTEAM Wireless Game Controller Compatible with PS-4/ Slim/Pro Console (Jet Black)
Shop Holiday Gift Guides & be a go-gifter
Early Black Friday deal
Native
Web

Amazon

Amazon's app-based experience features native tabs to access information related to each user's account, as well as a search bar to search for items available within its massive product catalog. The product catalog itself, though, is web-based to keep up with the many different changes in product listings and product availability taking place at any given time.

Like Netflix, Amazon also uses WebViews to complement its app-based experience with easy access to web-based information, including account information, order histories, and Amazon help center resources. This helps the retailer deliver personalized service across many different applications with minimal developer updates to the app itself.

How are enterprises embedding web content into native apps today?

Large enterprises are building on top of stock WebViews—which are native web browser components—that Apple and Google provide in their standard SDKs. They are used primarily to display internal and external web content.

The problem with using stock WebViews is that only low-level APIs are exposed. Meaning, these companies are investing a lot of time and resources to build good user experiences on top of these stock building blocks. The following are just a few solutions engineering teams need to build to create advanced experiences for users.

- **Communication between web and native layer.** Engineers using stock WebViews need to create and implement a pathway to send data back and forth. They would also need to develop the process of handling authentication in cookies.
- **Updating web experience dynamically.** How are the applications kept updated in a dynamic fashion? Traditionally, applications would have to go through the normal app store submission and review process, which can take several days. So engineers would have to build mechanisms in their stock WebViews that would point to a server or load the updates remotely.
- **Managing native elements.** Permission dialog boxes, navigation, and load all have to be implemented and managed with stock WebViews. Engineers would also need to develop handling errors—what happens when something doesn't load correctly on the web, native, or both.

The difficulty of building solutions on top of stock WebViews

While stock WebViews, on the surface, should save engineering teams development time, they could often lead to bigger problems if not developed correctly.

- **Time consuming and costly to maintain.** We get at least one major iOS and Android release every year, which means there are bugs and new features to integrate. They would need to make sure these custom WebViews are kept up to date and don't break with new operating system versions or patches.

  Building solutions on top of stock WebViews takes time away from core competencies. Companies like Amazon want to focus on creating the best store experiences for their customers. They don't want to take the time to build complicated features on top of WebViews.

- **Security risks and vulnerabilities.** Engineering teams need to make sure they are building solutions on top of WebViews that pass data securely and correctly. They have to make sure user data can't be compromised—a very complicated task.

- **Lack of granular controls for collaboration between the native and web team.** Native teams want to be sure the web teams are building in a sandboxed environment which minimizes the risk of breaking the native features of the app. Native teams also want to be sure that the web teams' contributions are working properly with the rest of the app. This functionality doesn't come standard with stock WebViews and needs to be built from scratch.

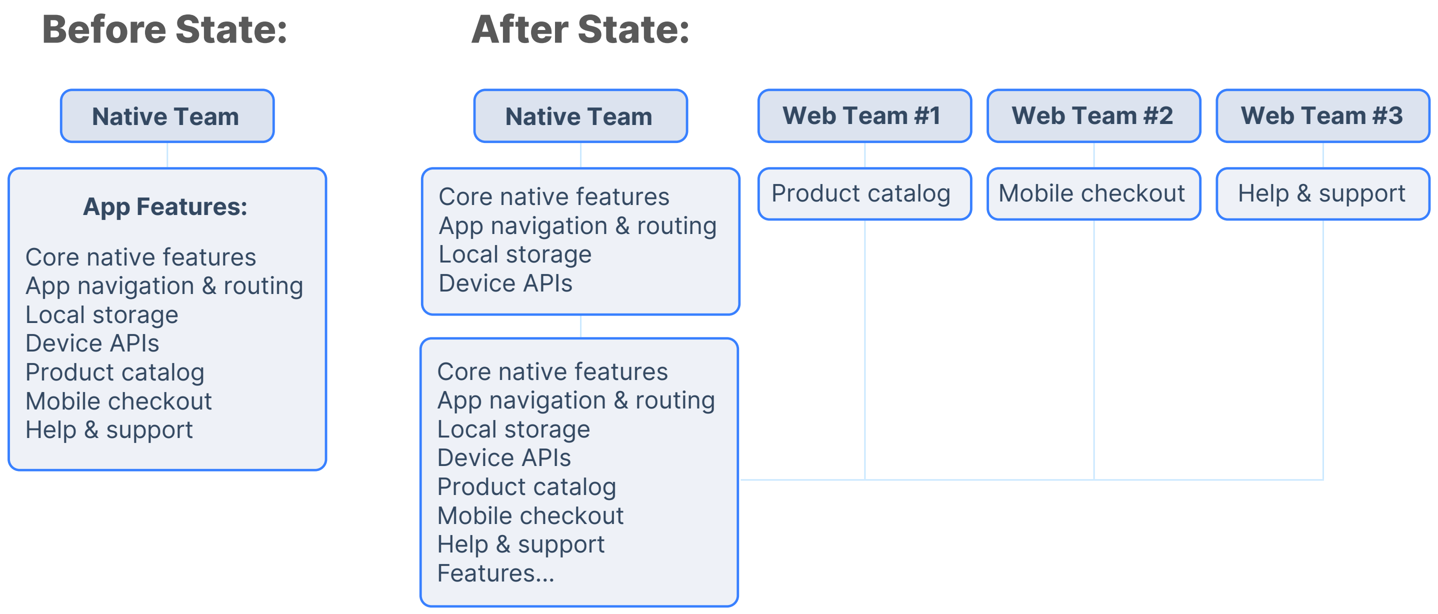
- **Lack of native feature access from the web.** While engineers can use the standard web APIs like the camera integrated in the stock WebViews, there are some user experience pieces that need to be built in order to provide a really great experience. Some of the building blocks available are low-level, basic, and not up to par with what customers expect.

Introducing Ionic Portals

While the micro frontends implemented by Netflix, Apple and Amazon have upgraded those native apps through improved functionality and reduced developer demands, the approach employed by those enterprise brands also creates certain constraints and limitations.

Although the WebViews approach is popular because it can be utilized for both Apple and Android applications, it isn't always intuitive, easy, or cost-effective for businesses eager to unlock the performance and value of their micro frontend.

Fortunately, there's a better way to embed web content into native applications. Ionic Portals addresses the limitations of WebViews by making this component fully secure and extensible, while also reducing resourcing needs to make this development approach more scalable.



With Ionic Portals, your micro frontend is able to reap the following benefits:

- **Expanded WebView capabilities,** including easier and more secure options for adding web-based content into native app experiences.
- **iOS and Android** development compatibility.
- **Full access to native device components and device API,** such as smartphone cameras, geolocation services, and more.
- **Security and data privacy compliance support** from our team of trusted security experts.
- **Granular control over which native sections and features are accessible to web developers,** mitigating issues and bugs that bleed outside of the intended web development container, and simplifying QA.
- **Empower existing web developer talent** to contribute to native mobile apps, allowing each team to focus on doing one thing well.
- **Scale development with multiple teams** working in parallel.
- **Live Update capabilities that bypass the typical app store review process**—which lets your brand push out new updates and capabilities on a more responsive, user-friendly timeline.
- **Advisory support, developer training and other services** to help you seamlessly integrate new web-based capabilities into your native app.

Once you integrate your native application with the Portals development library, you can set web developer permissions for specific parts of your native app and then explore our pre-built native plugins designed to empower your micro frontend with new web-based experiences and capabilities.

Choosing your services is often the hardest part. By the time you've picked out the native plugins you want to add to your native app, all that's left to do is integrate the pre-built plugin and ship the application off to your end-users!

CONCLUSION

Build Native Applications That Do More With Less

Whether you're building a house or building an app, the right tools make all the difference. Micro frontend development opens up a whole new range of possibilities when it comes to controlling app resourcing and development costs—while also making sure the app is set up to deliver long-term value.

Micro frontends offer the flexibility, ease of use and scalability every enterprise needs to deliver great user experiences both now and in the years ahead. As your frontend needs change and evolve, you'll have a cost-effective framework in place to quickly adapt and onboard new services into your native application.

Don't waste any more time and energy on maintaining your inflexible, monolithic frontend. Take the first step toward building a micro frontend of your own—talk to an Ionic App Specialist today.

About Ionic

Ionic is a leader in enterprise mobile app development, with 5 million developers worldwide and thousands of enterprise customers who use Ionic to build mission-critical apps for their customers, both external and internal.

It powers 15% of apps in the app store, not including thousands of apps built internally at enterprises for every line-of-business need. Ionic is unique in that it takes a web-first approach, leveraging HTML, CSS, and Javascript to build high-quality iOS, Android, desktop, and Progressive Web Apps.