LANSA®
An Idera, Inc. Company

Progressive Web Apps

Create a Universal Experience Across All Devices

Introduction

The web has changed tremendously since its static beginnings. As the face of an online business, a website is built to invite new and internal customers into the world of an organization. However, as time went on, the web lacked the functionality, design, and native features that organizations desired the most, phones became ubiquitous, and mobile applications became a game-changer. Imagine downloading an application to your mobile device and being able to access local device storage, cameras, and Near Field Communications (NFC). So we moved to creating mobile apps, housed in app stores and sometimes limited to the mobile and desktop operating environments that we use on a daily basis. Fast forward to 2020, and a new game-changer has emerged, one whose technology has been around for quite some time, but the guidelines and realizations of possibilities were seldom fused together.

Welcome the progressive web application—a way to create a universal experience across mobile and desktop alike and to view and utilize the web in a completely new way while increasing your marketability.

What's the Goal of a Progressive Web Application?

The goals of progressive web applications are best described by the following three fundamentals:

Capable

Progressive web applications have a vast array of web APIs at their disposal to accomplish much of what a native application can provide. With technologies such as WebRTC, a progressive web application can be a chat client that connects via a peer-to-peer interface, stream video games down to the application, and enable live video chatting.

Reliable

Progressive web applications are fast, and the goal is to reach the next billion users who have limited network connectivity. End users need their applications to be a fast experience, have the most recent content, and be able to interact with their other applications with as little friction as possible.

Installable

A progressive web application can be installed to the desktop and mobile environment with its own unique icon. When you launch a progressive web Application, you are not viewing a web browser; you are viewing an application window that looks and acts like any other application you may have on your phone, tablet, or computer. Additionally, the user experience and interaction is seamless across environments—that is, end users will have the same experience across mobile and desktop environments.

Why Progressive Web Apps?

The progressive web app centers itself around core principles that enable the web to be more user friendly, performant, and interactive. The web has become a set of powerful standards that provide almost endless possibilities for your web applications. From interacting with native device features such as the camera all the way to being installable on desktop and mobile devices, progressive web apps provide a whole host of possibilities. However, what you should know is that many of the standards that make progressive web applications possible are not new, but with the emergence of key technologies such as the service worker, PWAs are more poised than ever to create scalable, reliable, and flexible experiences within the browser.

All of this sounds great, but what are the benefits of PWAs for the enterprise environment? That question can be answered readily by Pinterest, Twitter, Nikkei, and a host of other businesses with varying models that have found that providing a progressive web app increases retention and acquisition of new users, while decreasing the amount of data usage. These are only a few use cases for the utilization of progressive web apps. Let's take a look at how PWAs can be used for your business practice:

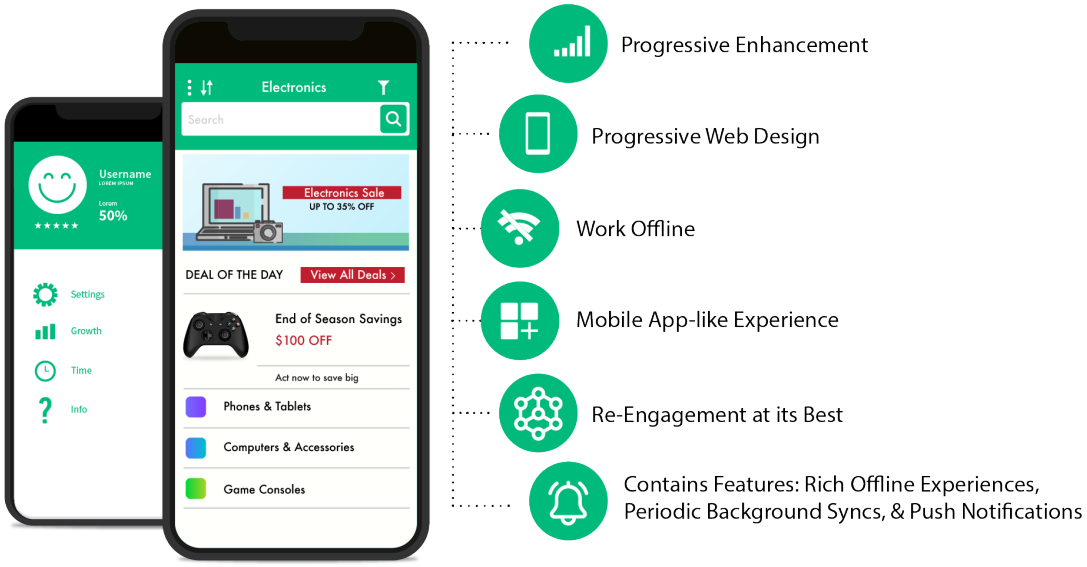
- Crafting new user experiences for employees to access and query the information they need on any platform
- Creating a web-based ticketing system for support agents, enabling work to be completed on a desktop or mobile environment
- Utilizing Web NFC to communicate with IoT devices and retrieve information about manufacturing equipment performance
- Creating a Content Management System (CMS) to deliver student content that can be available offline, enabling students with limited internet connectivity to view, edit, and complete assignments
- E-commerce with the Web Payment API and Paypal integrations
- Mobile accident assessment applications with the ability to take photos and videos of an accident, use audio recording to record testimonies, and more.

Twitter

- 65% increase in pages per session
- 75% increase in tweets sent
- 20% decrease in bounce rate
- Requires less than 3% of device storage compared to the native twitter for android.

Nikkei--publishing for more than 140 years, one of the most influential media businesses in Japan:

- 2x better speed index
- 14 seconds faster time-to-interactive
- 75% faster loading with prefetch
- 49% more daily active users
- 58% more conversions (subscriptions)



How Do They Work?

Service Workers

At the core of a progressive web application is a wonderful piece of technology called the service worker. The service worker acts as a network proxy that greatly speeds up the performance of a PWA by interrupting a request for resources to the network to determine if the requested items have been cached first. If they have been cached, then the service worker will in most cases use the cached response first, decreasing the need for the overabundance of network requests we typically see. Remember, you have the flexibility to customize these actions of the service worker, enabling your application to mold to the best use case of the business.

The use of caching by the service worker allows your PWA to work offline and load reliably regardless of the end user's network connection. For example, the service worker can be programmed to always serve content from the cache first and fall to the network. The added benefit of this strategy is that, if the content is not available in the cache, then what is served from the network can be cloned by the service worker and placed into the cache for future use.

In a way, service workers are akin to dark magic for the web. One of the magical parts of the service worker is that it creates a semblance of multithreading in JavaScript, meaning that while it fetches network requests, caches resources that you nominate to be available, and helps route requests to the right places, it is not blocking the UI of the application, allowing the end user to continue to interact with the application rather than seeing a page spinner and waiting for more content.

Web Storage

For the business environment, storage is an important part of the PWA experience. Previously, web applications have been limited to utilizing a small subset of storage within the browser. Frustratingly, the subset of storage had difficulty in persisting, required multiple page refreshes when updates were available, and sacrificed in the area of performance. With PWAs, we have been provided guidelines and a new means of interacting with storage that allows for dynamic persistence of data. Rather than use the application cache, we can use cache storage and create multiple stores for the necessary resources to load our applications and file-based content. In conjunction with cache storage, we are able to utilize IndexedDB to store other data, query the stored data, and much more. As you read this, you are probably asking, "How much data can I store?" The answer is:

LOTS!

For example, within the Chrome browser an application can use up to 60% of total disk space, Firefox allows up to 2 GB of storage, and Safari allows for up to 1 GB and will increase the limit in 200MB increments with permission from the end user.

Web App Manifests

The web app manifest is another part of the PWA that enables it to be installable on desktop and mobile devices. Basically, the manifest wraps relevant information about the application in JSON. Within the web app manifest, you can provide the resources for icons, versions, descriptions, related applications, splash screens, and much more. At the end of the day, having the web app manifest enables your application to install an icon on the device and launch as if it is a native application.

More Information:

- Service workers: An introduction
- Service worker concepts and usage
- Add a web app manifest

Building a Strategy around PWAs

Think offline first

When developing a PWA it is best to think in terms of the application being developed offline first. What this means is that the application should be developed with remote and limited network connectivity in mind. In considering how the application will work offline as you build out technical requirements, your team will find solutions that impact the vast majority of end users.

Examine your user base

Examine the data available about your user base. Question the types of actions that are necessary to them. What are the jobs that need to be completed? What are the pain points in your current application? How do your end users use the application today? Asking these questions will help formulate a foundation to determine how a PWA fits into the overall strategy for your organization.

Create the ideal workflow

When determining your organization's PWA strategy, start from the beginning of the user experience. Ask questions such as, What are the goals and jobs that need to be accomplished? From there sketch out what the workflow would look like. How does an end user get from point A to point B? What type of data should be accessible offline for the end user? How do they retrieve that data? Should the end user be able to save information for use later? Thinking along these lines aids in considering both a design- and a development-driven approach to your organization's PWA strategy and helps to clarify how and why the organization should utilize a PWA.

Whiteboard technical requirements

Following the design of the ideal user workflow, it is necessary to consider the technical requirements necessary for developing a PWA. Engaging in a whiteboard session with the necessary stakeholders such as business analysts and IT departments will provide the necessary conversations, feedback, and requirements for laying out the flow of information, architecture requirements, and user requirements for the PWA.

Equally important, you must acknowledge that while the web is becoming more powerful every day, some areas are still in experimental stages, such as interacting with a device's native file system through the web.Therefore when writing the technical requirements you may find that some functionality is only available through a native application. Even in those scenarios it may be beneficial to still create a PWA that links to the native application in the app store to download if users need to access more advanced functionality.

Understand the business impact

There are a number of ways to view the business impact of developing a PWA, and they are all positive. For businesses, utilizing a PWA provides an easier way of marketing solutions to end users. Since a PWA is linkable, it is easy to email, text, or instant message the link to users, enabling them to visit the site and install the application if they so choose. An added benefit is removing the necessity of entering an app store to access your application's features, resulting in more traffic and interactivity with your web application.

B2E app for company employees

A PWA for the enterprise application could begin with a portal for other partners or employees to enter and gain access to systems for ordering supplies or parts or updating manufacturing data across their desktop and mobile devices. Having this type of fluid installability removes barriers to entry. Additionally, since a PWA is browser-based, whitelisting and Mobile Device Management (MDM) solutions do not come into play. In other words, there is no need to deploy the application; employees can simply visit the website and download the application directly to their mobile or desktop environment, allowing data to be stored off the device and at the browser level.

Sample Web APIs Available for Use in PWAs

The APIs listed below are a drop in the bucket compared to the functionality that is available in the web today. From streaming media and video games to saving data offline and syncing back to the server when connectivity is regained, the web has grown tremendously. As an added bonus, there are more and more APIs being developed each day by Google, Microsoft, Mozilla, and more to become standards across the web. Such APIs provide the ability to exchange data through NFC tags, share links and information through the browser, and connect two different devices to share data—think Airdrop but for the web.

- Web Bluetooth API enables secure connections to bluetooth devices via the web application. An example of this would be connecting one's phone to a PWA that is being used on a desktop computer.
- Web Authentication API provides the ability for web applications to provide 2FA, fingerprint authentication, and more.
- Geolocation API enables a web application to request the location of the device viewing the application.
- Media Streams API provides web application support for streaming audio and video data. Visit https://music.youtube.com for an example of installing a streaming service to your desktop or mobile device.
- Image Capture API enables the web application to capture images or videos, retrieve metadata, and more.
- Payment Request API makes it easier for end users to manage their payment credentials and submit payment for goods and services to merchants without utilizing a form.
- Push API enables a web application to receive messages pushed from the server regardless of whether the web application is actively in use.
- Beacon API enables a web application to send critical, one-way information to the server asynchronously without blocking the UI and minimizing resources used. A beacon could be device analytics, measurement data, error codes, etc.
- Web Crypto API provides web applications with cryptographic and key management functions to implement security features such as privacy and authentication within the application. An application can create and verify digital signatures, encrypt and decrypt data, and generate and derive keys.

How Do You Build PWAs?

PWAs consist of HTML, CSS, JSON, and JavaScript and can be built using a variety of different methods. They are built nearly identical to regular web apps but with the addition of code needed to handle the service worker and manifest. Various libraries and frameworks are available to help accelerate the development process. However, even using these libraries and frameworks, a substantial amount of hand coding will still be needed to finish the app. Using these traditional methods will require extensive knowledge of service workers and how they need to interact with your app. Along with service worker knowledge, creating the tedious aspects of the user interface will also be a manual task. This approach really isn't all that difficult for small applications with limited functionality but can quickly become cumbersome and inefficient for enterprise PWAs.

Low-code PWA development

Low-code is the perfect use case for PWA development. The user interface is created through drag-and-drop design, often with standards like Google Material Design built into the designer. In addition to saving time creating the visuals of your app, low-code will automatically handle the service worker and manifest, meaning the developer doesn't need to worry about the details of how the app will interface with the device. Much like other low-code principles, the developer focuses on creating solutions instead of spending time on coding functionality before coding to the actual business solution. Without having to know the intricacies of PWAs. As a result, low-code developers can quickly create enterprise-level PWAs and businesses will see high ROIs from their low-code investments.

Are PWAs Right for Your App?

PWAs are not the answer to every app being developed. There are still some cases in which creating a native app might be the best option. If you intend to solely focus on mobile, with in-app purchases, then you will want to go with developing a native mobile app. However, if you want your app to be used across all devices and be discoverable by SEO, then PWA is the way to go.

You also might see some misinformation surrounding PWAs. Don't let the myths deter you from seriously looking into developing them. Progresive web apps do, in fact, have access to nearly all aspects of a mobile device like native apps, including NFC. Also, PWAs do not use more battery power than a comparable native app, and they can be run on older hardware and browsers.

Whether you need a customer-facing or internal app, a PWA should be given serious consideration for your development approach.

Be on the lookout for more information on how Visual LANSA is creating the next steps for low-code enterprise PWA.

Edgar Wharton
Senior Product Manager

About the Author

As a Product Manager, Edgar believes that the single most important thing is connecting the dots between all the complex systems that we have to maneuver through on a daily basis. Drawing from his experience within cultural sociology, UX Research, EdTech, and mPOS application development, Edgar thrives on hypothesizing and testing new insights that benefit end-users of all backgrounds. He firmly believes that Visual LANSA is not only a low-code development platform, but a toolkit capable of teaching and simplifying some of the most complex tasks and concepts in computer programming.

LANSA's mission is to make advanced software simple. We do this by taking care of the underlying and constantly changing technologies, enabling software developers to focus on the business problems that need solving and to rapidly produce high quality software.

When businesses can effectively capitalize on IT to innovate and differentiate, they gain a competitive edge. LANSA helps to make it happen.

LANSA®
An Idera, Inc. Company

THE AMERICAS
HQ Austin, TX
Tel +1 630 874 7000
Email info@lansa.com

EUROPE
HQ London, UK
Tel +44 1727 790300
Email info@lansa.co.uk

ASIA PACIFIC
HQ Sydney, Australia
Tel +61 2 8907 0200
Email info@lansa.com.au