Review of Benefits and Challenges of Progressive Web Apps

Mayuri Rangari; Tushar Sudhir Raghatate; Yash Gajanan Pal
Department of MCA, G H Raisoni College of Engineering and Management, Nagpur, Maharashtra, India.

Abstract:
The authors have conducted an in-depth study of Progressive Web Applications (PWAs). Progressive Web Applications (PWAs) are a major web technology innovation that aims to fill the gap between native mobile apps and conventional web experiences. The present paper is an intensive examination of PWAs, touching upon their strengths and weaknesses in the implementation process. The strongest benefits of PWAs include stronger performance, offline ability, and cross-platform availability, which jointly bring more accessibility and engagement with the user base. Though there are challenges in deploying PWAs, including inconsistent browser support, security risks, and the issue of incorporating features of policies into current web infrastructures. Based on a synthesized review of available literature and case studies, this research seeks to make an informed contribution towards the real-world implication of deploying PWAs in modern web development practice.

Keywords: Progressive Web Applications, Web Development, Native Applications, Cross-Platform Compatibility, Progressive Web App

1. Introduction

As PWAs increasingly become popular, their place in contemporary web development is a matter of debate. This paper discusses the advantages and disadvantages of PWAs, assessing their influence on the web and mobile environment. By considering their technology base, uptake patterns, and limitations, we seek to determine whether PWAs have the potential to be a viable substitute for native applications in the long term.

Rapid development in cell phone technology has had a vast impact on the development of websites, forcing businesses to optimize end-user experiences in various platforms. Nevertheless, many traditional mobile websites lack functionalities including offline access, push notifications, and home screen installation, preventing them from competing with native apps [1]. To fill this void, Progressive Web Applications (PWAs) were launched in 2015 by Alex Russell and Frances Berriman that introduced a blended method combining the best of both web and native apps [2].

PWAs use innovative web technology such as service workers, web app manifests, and secure HTTPS to create an unobstructive user experience. Unlike traditional mobile web apps, PWAs have the capability of running offline, loading at super speed, and providing app-like interaction, resulting in greater engagement and availability [3]. Major brands have been using PWAs to boost performance, conserve data, and open up access to areas where networks are inconsistent [4].

DOI: https://doi.org/10.5281/zenodo.15605514

Though they provide their advantages, PWAs also have limitations. One of the largest drawbacks is limited native device feature support, since

PWAs are constructed upon web APIs that may not necessarily support advanced features like Bluetooth, NFC, or fingerprint scanning. Consistent browser support also does not exist, affecting cross-platform consistency. Security is also a concern, since PWAs are founded on service workers that must be well maintained in order not to introduce vulnerabilities [5].

2. Literature Review

The User interaction has been heavily affected by the shift away from classical websites and toward web applications compatible with mobile phones. Flexibility of architecture and fluid layout were the central components of the initial mobile web solutions. Progressive Web Applications (PWAs), closing the gap between the web and native mobile apps, were made mainstream by the threshold of speed and offline limits.[3]. Several empirical studies have assessed PWAs' impact on performance metrics such as page load time, time-to-interactive, and conversion rates. Research findings suggest that businesses adopting PWAs observe higher user retention and improved conversion rates due to their fast and engaging nature. For instance, an e-commerce platform implementing a PWA saw a 70% increase in engagement and a 20% rise in conversions [10].

PWAs offer enhanced performance, offline functionality, and seamless user experience, leveraging service workers and web app manifests. Research highlights that PWAs can reduce loading times by up to 80% compared to traditional mobile websites. Additionally, they improve re-engagement through push notifications and home screen

installation features [7]. Despite their advantages, PWAs face challenges such as limited support on iOS devices, restricted access to certain native APIs, and varying levels of browser compatibility. Studies indicate that while Android provides robust PWA support, Apple's ecosystem restricts background processes, impacting offline performance and notifications [5].

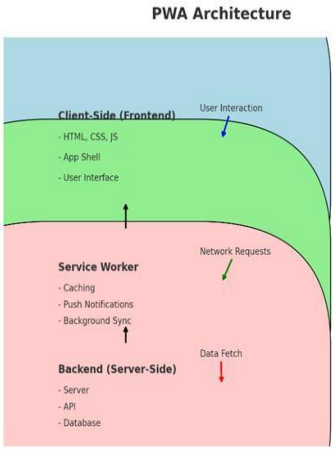
The academic discourse on PWAs focuses on their technological advancements and adoption rates across industries. Studies discuss the role of JavaScript frameworks like React and Angular in simplifying PWA development, along with advancements in caching strategies and service worker optimizations. Furthermore, PWAs have been recognized as a cost-effective alternative to native app development for small businesses [2].

Case studies from leading tech companies demonstrate the real-world impact of PWAs. For example, Twitter Lite, a widely cited PWA implementation, resulted in a 65% increase in pages per session and a 75% reduction in data consumption. Similarly, Starbucks' PWA allowed users to browse the menu and customize orders even when offline, enhancing customer satisfaction [6].

3. Methodology

The objective of a Progressive Web Application (PWA) is to deliver a user experience similar to native mobile apps but with the inherent nature of web applications. PWAs are designed to overcome the shortcomings of earlier web development practices by delivering an integrated solution that is, without fail, cross-platform compatible. The PWA development approach is designed to provide native app-like experiences with instant on-load, offline, and performance enhancement [3][10].

DOI: https://doi.org/10.5281/zenodo.15605514



[16]

Technologies and Components

1. User Interface (App Shell): The static part of **a** PWA is built using HTML5, CSS3, and JavaScript. The user interface, or app shell, is cached locally by the user's device in order to offer fast loading even where there is minimal network connectivity [3].

2. Service Worker: An integral part of PWA architecture, the service worker is an application that executes in the background and handles caching, background sync, and push messages. It enables offline availability and improved performance by utilizing local storage [3].

3. Web App Manifest: This JSON file communicates metadata, which allows PWAs to be installed on the home screen and act like native apps.

4. Frameworks and Libraries: The Angular and React frameworks are commonly used to create PWAs with the added advantage of having service worker integration out-of-the-box for rich features [3].

5. Secure HTTPS Connection: PWAs are loaded via HTTPS for secure data exchange, as against standard web apps [10].

Approach and Data Collection

Qualitative and quantitative methods were employed in order to learn about the drawbacks and advantages of PWAs. Developer interviews and reviews of effective PWA uses, such as Twitter Lite, were used to learn about the improvements in performance and user interactions. Data were also gathered by online questionnaires and web metrics tools to compare key performance indicators (KPIs) such as load time, bounce rates, and retention of users [15].

Case Study: Twitter Lite

Twitter Lite, being a PWA, was built to provide instant and stable service in low connectivity areas. The app size was minimized to under 1 MB, leveraging caching via service workers to enable offline capability. Push messages and background sync additionally increased user interaction [15].

Custom Addition

With the ongoing evolution of web standards and browser APIs, PWAs will be able to fill the gap between native and web apps and become a possible choice for companies wanting to offer an app-like experience on the web.

1. Benefits

1. GlobalandCross-Browser Compatibility: By way of progressive enhancement, PWAs run wonderfully well across a whole host of browsers (e.g., Chrome, Opera, Brave, etc.) and geographies (e.g., India, UK, North Korea). Wherever, on whatever browser, that means everyone should be able to use it.

2. Responsive design: Because PWAs are responsive, they can be used on a variety of devices, including desktops, tablets, smartphones, and possibly even new ones in the future. This ensures a consistent experience across platforms and screen sizes. Also, it improved user interaction because of responsive design.[1]

3. Accessibility of Web pages through URL: Because PWAs can be accessed using a URL, just like any other website, they are easy to reach and share. This offers an app-like experience without the need to install apps from the app store.[3]

DOI: https://doi.org/10.5281/zenodo.15605514

4. **Setting Up a Home Screen:** PWAs offer easy access when they are positioned on the home screen of the user's device, much like native apps do. Customers can now interact with the app directly from their home screen without using a browser or going to a website.[6]
5. **Performance:** In addition to caching and preloading, service workers can be used for background processing to provide users with a consistent and instantaneous experience.
6. **Offline Capability:** PWAs allow offline usage by locally storing essential static assets (such as HTML, CSS, JS, images, and fonts) on the user's computer. This makes it possible to access the application offline after the initial visit and guarantees a better experience in scenarios with poor internet connectivity.[8]
7. **PWA Compatibility on Desktops:** The ability of applications created with the PWA development methodology to function and display on desktop and laptop computers without distortion or issues is known as desktop compatibility. However, this is not the case for the Native and Hybrid approaches, as applications are only usable on mobile devices that support the relevant and particular platform.
8. **Push Notifications:** A PWA is a web application created to offer a native-like user experience on mobile devices, complete with push notifications. Users can view re-engaging content with the help of the push notification tool. [9].

2. Challenges

In spite of the many advantages of Progressive Web Apps (PWAs), their implementation and adoption are confronted with a series of challenges. These challenges stem from the technical complexities in PWA development and deployment, along with constraints in user perception as well as industry adoption.

1. Service Worker Complexities

One of the main difficulties in developing PWAs is the intricacy of handling Service Workers (SW), which are essential for supporting offline functionality, push notifications, and background synchronization. Deploying and maintaining SWs demand careful attention to caching policies, network interception, and event management. Additionally, because service workers run in parallel with the main browser thread, developers face challenges with debugging, error management, and version control (Paragraph 2). Also, processing PWA requests via service workers can be problematic since developers must determine whether to return cached responses, perform network requests, or create new responses. This becomes a complex issue that creates questions and issues, especially when updates are involved, as developers must effectively handle cached data, service worker files, and other resources [11].

2. Performance Optimization Issues

PWAs aim to provide responsive and rapid experiences on a large variety of devices and network situations. Nevertheless, it can prove hard to sustain performance when adding rich features. Complex frameworks and heavy resource functionality can lead to responsiveness problems, particularly on lower-end devices. Optimization techniques are needed to balance immersive features and low-resource usage, and they might be challenging to realize due to device diversity and browser support [12].

3.ChallengesofAdoptionand Implementtion

Although PWAs have proved successful in sectors such as e-commerce, news, and social media, their adoption is vastly different across industries. Certain sectors are slow to adopt because of the learning curve involved with PWA-specific technologies, including service workers and web app manifests. Moreover, providing uniform cross-browser compatibility is challenging

DOI: https://doi.org/10.5281/zenodo.15605514

because PWA support levels are different across browsers [14].

4. Technical Constraints and Browser Compatibility

Not all browsers support PWA features fully. While major browsers such as Chrome and Firefox work very well with PWAs, others, including Safari, have reduced functionality, especially in terms of push notifications and background sync. Additionally, PWAs do not have access to some device-specific hardware capabilities such as NFC, advanced camera capabilities, or Bluetooth, as natively installed apps have access to, reducing their use in certain situations. [13][15].

5. User Awareness and Perception

End-user perception continues to be a key impediment to the adoption of PWA. Most users are not aware of how to use PWAs or consider them less secure than native apps from official stores. This perception impacts the wide-scale adoption of PWAs since users prefer the familiarity and perceived security of native applications.

6. Monetization and Business Challenges

PWAs are confronted with monetization difficulties because they do not normally get listed on app stores, leading to less visibility and fewer opportunities for subscriptions or in-app payments. Developers frequently have trouble including secure and reliable payment mechanisms, which may translate into revenue losses. Furthermore, because PWAs do not benefit from the visibility advantages of app marketplaces, they can encounter limited visibility, particularly for new apps with no existing user base [15].

7. Security Issues

Although PWAs must be served via HTTPS, they continue to depend on web technologies susceptible to typical web security vulnerabilities. Strong security features must be employed to safeguard personal user information, but poor practices leave PWAs open to cyber threats [15].

4.Conclusion

Progressive Web Apps (PWAs) have great potential to become part of the standard web development tools, even though their relatively modest uptake by companies is so far. While PWAs have many benefits, it is not clear if they will ever fully replace native apps or responsive websites. For certain business models, particularly gaming, messaging, social media, and entertainment, native apps may remain the best choice. But as people become used to the PWA experience surfing the web, they will eventually embrace these apps, providing a native-feeling experience without installation. Hence, PWAs will probably coexist with native apps instead of fully supplanting them.

The developer community is slowly putting resources into PWAs, but an evident lack of academic work implies scope for additional research. The present situation of PWA technology indicates continuous enhancements, especially in browser support and device compatibility. Although Chrome remains ahead in PWA support, other browsers, such as iOS Safari, remain incomplete in terms of compatibility with core features like Service Workers. This discrepancy highlights the importance of combined efforts to drive the web-native development ecosystem forward.

PWAs offer a potential solution to overcome challenges related to conventional mobile development, including shortening development time, maintenance expenses, and handling platform fragmentation. The use of capabilities such as offline loading, background synchronization, and push notifications fills the gap between web applications and native experiences, offering PWAs as a feasible alternative for mobile application development. More investigation is required to solve memory management and efficiency on different devices.

It is important to understand the challenges developers experience when

DOI: https://doi.org/10.5281/zenodo.15605514

developing PWAs in order to enhance innovation in this area. Through examining discourse in online forums of developers, researchers have identified hot topics, including the Service Worker lifecycle and technical issues. Resolving these developer issues and offering complete solutions will accelerate the use of PWA technology.

In the future, PWAs hold vast potential for innovation and usability. By solving current issues, embracing innovative strategies, and promoting cooperation between developers and researchers, the future of PWAs can be designed to provide stable, immersive, and user-friendly web experiences. Ongoing developments and a continued focus on improving PWAs will make them competitive with native apps, ultimately transforming the digital world.

References

[1] Sayali Sunil Tandel, Abhishek Jamadar, "Impact of Progressive Web Apps on Web App Development", IJIRSET, Vol. 7, Issue 9, 2018.

[2] Amit Mhaske, Aditya Bhattad, Priyanka Khamkar, Radhika More, "Progressive Web App for Educational System", IJIRSET, Volume: 05 Issue: 01, 2018.

[3] V S Magomadov, "Exploring the role of progressive web applications in modern web development", J. Phys.: Conf. Ser. 1679 022043, 2020.

[4] Rachma Verina Rochim, Alam Rahmatulloh, R Reza El Akbar, Randi Rizal, "Performance Comparison of Response Time Native, Mobile and Progressive Web Application Technology", INNOVATICS-VOL. 5 NO. 1 (2023) 36-43.

[5] Andreas Biørn-Hansen, Tim A. Majchrzak, "Progressive Web Apps: The Possible Web-native Unifier for Mobile Development".

[6] Tim A. Majchrzak, Andreas Biørn-Hansen, Tor-Morten Grønli," ProgressiveWeb Apps: The Definite Approach to Cross-Platform Development?", Proceedings of the 51st Hawaii International Conference on System Sciences j 2018.

[7] Svitlana O. Leshchuk, Yurii S. Ramskyi, Anastasia V. Kotyk and Serhiy V. Kutsiy, "Design a progressive web application to support student learning", Vol-3077, Paper-16 CEUR Workshop Proceedings, 2021.

[8] Ivano Malavolta, Giuseppe Procaccianti, Paul Noorland, Petar Vukmirovi, "Assessing the Impact of Service Workers on the Energy Efficiency of Progressive Web Apps", 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft).

[9] Oluwatofunmi Adetunji, Chigozirim Ajaegbub, Nzechukwu Otunemec, Olawale J. Omotoshod, "Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development", American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS) (2020).

[8] Ivano Malavolta, Giuseppe Procaccianti, Paul Noorland, Petar Vukmirovi , "Assessing the Impact of Service Workers on the Energy Efficiency of Progressive Web Apps", 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft).

[9] Oluwatofunmi Adetunji, Chigozirim Ajaegbub, Nzechukwu Otunemec, Olawale J. Omotoshod, "Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development", American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS) (2020).

[10] Elias Yasar Akyol, "Barriers of Adopting Progressive Web Applications – A Qualitative Study Focusing on the Swedish Context", Spring 2023: MAGI02.

[11] Mohammadreza Abbasnezhad, Amir Jahangard Rafsanjani and Amin Milani Fard, "Challenges in Developing Progressive Web Applications: An Empirical Study Using Stack

DOI: https://doi.org/10.5281/zenodo.15605514

Overflow", Journal of Soft Computing and Information Technology (JSCIT), Spring 2024.

[12] Chakradhar Avinash Devarapalli, "Progressive Web App (PWA): Optimal Strategies & Challenges", International Journal of Research in Engineering and Science (IJRES), March 2024.

[13] Mani Shankar Srinivas Lingolu, Manoj Kumar Dobbala, A Comprehensive Review of Progressive Web Apps: Bridging the Gap Between Web and Native Experiences, International Journal of Science and Research (IJSR) · February 2022.

[14] Sourab John Jacob, Rajesh N (Assistant Professor), "Progressive Web Apps: A lighter alternative", International Research Journal of Engineering and Technology (IRJET) e ISSN: 2395-0056 Volume: 07 Issue: 04 | Apr 2020.

[15] Bangar Raju Cherukuri, "Progressive Web Apps (PWAs): Enhancing User Experience through Modern Web Development", International Journal of Science and Research (IJSR) · October 2024.

[16] pollo.ai/ai-image-generator

DOI: https://doi.org/10.5281/zenodo.15605514