2023 3rd Asian Conference on Innovation in Technology (ASIANCON)
Pune, India. Aug 25-27, 2023

Progressive Web Applications, a New Way for Faster Testing of Mobile Application Products

1st Aatmaj Mhatre
Dept. Computer Science
K.J. Somaiya College of Engineering
Vidyavihar, India
https://orcid.org/0009-0006-0795-0713

2nd Swati Mali
Dept. Computer Science
K.J. Somaiya College of Engineering
Vidyavihar, India
https://orcid.org/0009-0004-6055-4010

***Abstract*—Progressive Web application (PWA) is a rapidly emerging technology that aims to provide an intermediate between native applications and web applications. PWAs provide certain advantages over traditional web applications, making them easier and faster to build compared to native applications. This paper discusses the PWA methodology, its advantages, and applications in detail. It also compares PWA with native and website application developments and presents PWA as an effective method for rapid prototyping. This study uses an app developed for security guards at a university for identification and verification purposes to support the claims and demonstrate the strengths of PWA.**

***Index Terms*—Progressive Web applications, product development, user testing, mobile application development, entrepreneurship, software development lifecycle**

I. Introduction

The traditional method of launching a mobile-based application into the market uses native applications. The native applications are available on the App Store or Play Store and are installed on devices for development and testing purposes. However, application development using any native application requires a considerable amount of cost and time. Many startups do not possess that many resources and thus face delays in product delivery processes. The Progressive App Development (PWA) approach is a great boon in such scenarios, as the time and effort required to build a PWA are less when compared to a native application. In essence, a PWA is a web interface with an appearance and behavior similar to a native mobile app that can be used for application development. The PWAs also greatly support rapid prototyping. A Rapid prototype is a small-scale working prototype of the project in hand that can be used for user testing and concept validation. Such a prototype must be cost-effective, fast and easy to modify. Such prototyping allows startups to fail early. Rather than investing time and effort into building a native application and then failing, making a PWA rapid prototype is very cost and time-efficient. Moreover, PWAs are easier to change than native apps and hence can be used for A/B testing in the early stages of product development. Thus the most costly process of building a the final product can be shielded from requirements change by using PWAs for rapid prototyping and testing.

As PWA is still an emerging domain of research, the paper aims at providing an overview of the methodology, compare it with traditional development approaches and give a experimental analysis of advantages with application development with PWA. The claims in this study are supported and validated using an application developed using PWA that aims at identification and verifications of students at the security gates of a university campus.

The paper is organized as follows: Section I introduces the paper. Section II describes the literature survey of progressive web applications. Section III offers a new methodology that reduces the time required for testing of mobile applications using PWAs. Section IV describes the advantages of PWAs over Native applications in the context of rapid prototyping, while section V describes its advantages over web applications. Section VI presents the experimental results of PWA testing testing and finally, section VII concludes the paper.

II. Literature Survey

Researchers have been studying and practising more effective ways of software development. PWA is one of the strong candidates of such studies. The PWA concept was first introduced by Google in 2015. Since then, PWA development has been rapidly gaining popularity [5] [1]. It is considered an emerging technology with tremendous potential in the future. PWAs can be cached easily and increase user engagement [21]. Some popular PWAs include Uber, Spotify and Flipcart [9]. PWAs are especially beneficial for startup enterprises. Many studies have been carried out on PWAs that compare them against traditional native or web applications. Biørn-Hansen and others [11] [12] have scrutinized and compared PWAs against cross-platform app development approaches. They have tested various types of applications for performance with the help of two case studies. Tandel and Jamadar [22] have discussed the impact of PWA on web development. They have compared the features provided by Naive applications, general applications and PWAs. Mole [19] reviewed the different issues and challenges in cross-platform development and its applications, discussed the approaches that may resolve these issues and discussed how the PWA addressed the issues. Lee and others [18] have studied the security and privacy of PWAs. They have identified security and design flaws in PWA development. Fournier [14] has compared the performance of PWA and native applications in terms of smoothness, memory and CPU usage. The results of the study implied that

TABLE I
COMPARISON AMONG NATIVE, WEBSITE AND PWA

	Native	**Website**	**PWA**
Offline	yes	No	yes
Push notification	yes	No	yes
Mobile specific	yes	No	yes
Home screen access	yes	No	yes
No download	No	yes	yes
No marketplace	No	yes	yes
Linkeble and Sharable	No	yes	yes
Low data	No	yes	yes
No updates	No	yes	yes

PWA, though they have great benefits, were not as smooth as Mobile applications on Android. Frankston [15] explored various aspects of PWAs. Recently, Fauzan and others [13] have provided a literature review on PWAs.

PWA development has been discussed in various project reviews and theses. Rensema [20] has explained various aspects of PWA development like compatibility, performance, security, privacy, and user/business impact along with use cases. Lama [17] has discussed the technical details of PWA development including core tech, design, tools, and technologies. Kerssens [16] has studied the difference in the performance of PWAs with native and traditional web applications. They have shown that the performance and energy consumption of a PWA is at least similar or even slightly better when compared to the traditional approaches of Native and Web in mobile development. A summary of all the observations of the studies can be found in table **??**.

Various sources on the internet provide useful information on Progressive web applications. Web.dev [9] provides various case studies on PWA development. Google's PWA website [4] also provides various resources for PWA development. In this study, we aim to utilize PWA for rapid testing of ideas for new products. To our knowledge, there has been no previous study on this specific topic.

III. PROPOSED METHODOLOGY

In this section, we propose a methodology for the testing of a mobile application product.

The most expensive and time-consuming stage of mobile app development is the production stage. Building any feature in the native mobile application requires time. The feature has to go through stages like development, testing and deployment. However, in the early stages of product development, many ideas fail. This means that if a feature that has been built does not become a success, all time and resources are wasted. PWAs can help to minimize this risk of wastage by providing low-cost testable prototypes of the features. The development of a feature in PWA is cheaper when compared to native applications. If new features are built first using PWAs, they can be tested on real users. Moreover, multiple variants of the same idea can be A/B tested on PWA applications in a very easy manner. This testing can be done across all platforms on real users. If the feature idea fails, only a small amount of effort is lost. If the idea is a success, then the same feature can be built on native applications. The time taken to build this feature on the native application can be used to test new features in PWA in parallel. This way ensures the minimization of failure risk. The flowchart above shows the process using PWA. The same methodology can be applied to the initial prototype development and ideation process. The process without PWA is shown in figure III. The methodology proposed is shown in figure III.

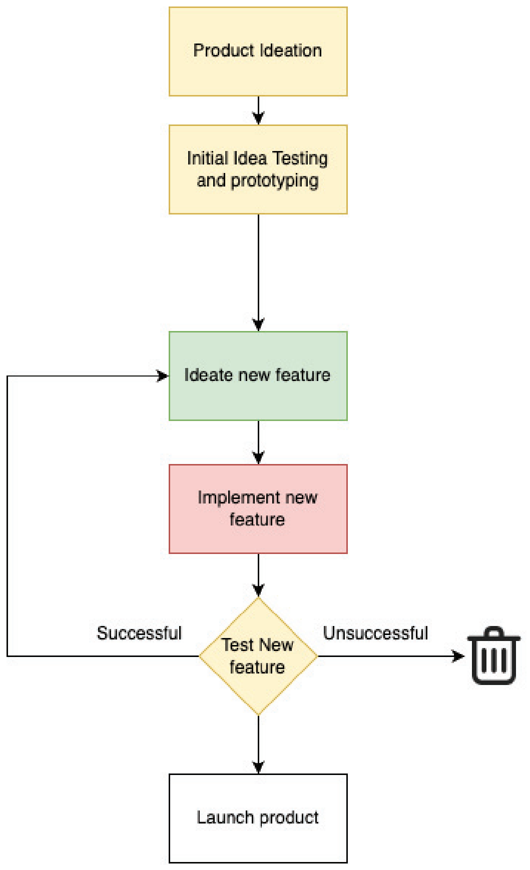

Fig. 1. Methodology without PWA

IV. ADVANTAGES OF PWA OVER NATIVE APPS

PWAs have numerous advantages. Many of them have been discussed in detail by many references [11] [17] [19] [5]. In this section, only the advantages of PWA that are relevant to the context of Rapid prototyping are discussed.

A. Ease of development

PWAs are seen as a unifier between native and web applications [11]. Since a progressive web application is built using the same tools as websites, thus providing compatibility with its web app versions [5]. Almost every web application can be easily converted into a PWA. Recent conversions of web apps to PWA for desktops include YouTube, Google Drive, Twitter, and Facebook.

PWAs can be built using the exact same tools as any other web application. This provides the following advantages in the context of startups -

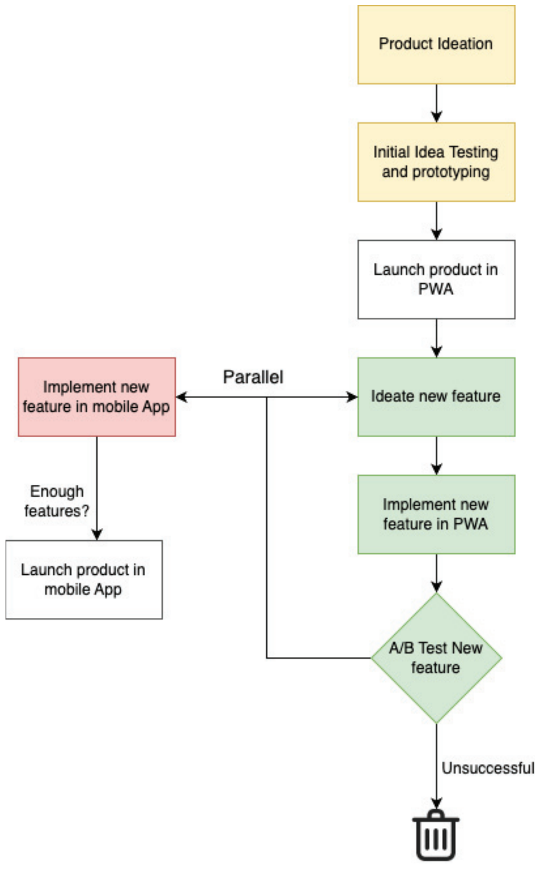
Product Ideation
Initial Idea Testing and prototyping
Launch product in PWA
Parallel
Implement new feature in mobile App
Ideate new feature
Implement new feature in PWA
Enough features?
Launch product in mobile App
A/B Test New feature
Unsuccessful

Fig. 2. Proposed methodology with PWA

1) This makes the development process faster.
2) Less cost of development.
3) Less talent is required than native application builders.
4) Ease of development provides lesser training time.

B. *Multiplatform*

One of the most important advantages of PWA for startup enterprises is the lesser overhead of development across various platforms. The write-once-use-anywhere benefit of PWAs drastically reduces the cost of development, testing and maintenance.

1) Operating systems compatibility: PWAs can be configured to be installed on any mobile device as well as desktop. Both IOS [6] and Android systems support PWAs in varying capabilities. PWA can be installed on Macintosh systems (OSX), Windows OS, and Linux OS variants as well as can be run through all supported browsers. This power of PWA makes it a popular choice for various applications that need to be run on mobile as well as desktop [5]. PWAs can run on almost all major operating systems and devices including Android and IOS mobiles, Ipads, and Chrome Books.

The table IV-B1 summarizes the features provided by PWAs on different platforms. This multiplatform feature makes testing across users with various devices easier. Currently, only Flutter is the most popular framework with the multiplatform feature.

TABLE II
COMPATIBILITY OF PWA ACROSS MOBILE FRAMEWORKS.

Feature	Android - Google	IOS - Safari
Web Push Notifications	Available	Not Available
Offline Browsing	Available	Available
Background Synchronization	Available	Not Available
App launching screen	Available	Not Available
Add to the home screen	Available	Available
Bluetooth	Available	Not Available
Geolocation	Available	Available
Geofencing	Not Available	Not Available
Image Capture	Available	Available
Video Capture	Available	Available
Host on Store	Available	Available

2) Browser compatibility: Chromium-based browsers provide support for PWA on all devices except iOS and iPadOS. As of now, iOS and iPadOS users have to use Safari for installing web applications. Firefox does not support PWA except for partial support for Android devices.

C. *No Update is required*

In the case of native applications, users have to be bothered again and again to update the application. This prevents the application from making small and numerous test cycles. Except for Over The Air (OTA) updates, other updates need to be verified by Google or Apple before they can be implemented. This further restricts the time to market for rapid test cycles. All this can be reduced by using PWAs which do not need frequent explicit updates. PWAs can synchronize and update in the background. More details on the background synchronization of PWA can be found in the study done by Behl and Raj [10]. This feature is very useful for the following cases -
1) Testing can be done as frequently as possible.
2) Features per test can be narrowed down to 1.
3) No requirement for testing updates from third-party stores.

D. *Easier A/B testing*

A/B testing is an important method for the development of any product. A/B testing helps reduce errors, find the best ideas and reduce guesswork early in the product development cycle. Such experiments are done by many large companies on a routine basis. PWAs are far easier to test than native applications. For example, if a certain feature needs to be A/B tested, then only that webpage can be modified into two versions, instead of making two versions of the entire application. This modular approach makes the addition of new features as well as A/B testing cost and time-effective. Building a PWA is similar to building a web application and changes can be easily performed. This makes it easier for A/B testing

E. *Low size*

In the case of prototypes, the brand image is not fully set up. Due to the uncertain feature set and large scope of errors in the testing phase, it is likely that users have not fully subscribed to the product. In such cases, if the application is consuming

TABLE III
SIZE COMPARISON OF PWA AND NATIVE APPLICATIONS

Name	PWA	Native
Pinterest	1.6 MB	25 MB
Twitter	252 KB	32 MB
Facebook	33 KB	69 MB
Flipcart	370 KB	18 MB
Amazon	325 KB	57 MB
Telegram	325 KB	29 MB

a large amount of space in their devices, users may uninstall the applications. However, a PWA takes a very less amount of storage space in the application. This is ideal for users who do not have data or storage to download the application. A comparison between the download size of PWAs and their native versions can be seen in table IV-E.

F. Lesser loading time

PWAs are faster than native applications [2]. Due to the transition from server-side to client-side as well as caching information, the loading speed for the application is drastically reduced, thus providing a seamless experience to the users. In a startup scenario where there are multiple competitors and no brand name, reducing the time latency in users is critical. Lesser loading time leads to more user retention and more revenue. According to a case study, every additional second of page load time resulted in 15-20% lower revenue and sales. By reducing this load time, Orange Polska S.A. was able to convert 52% better by optimizing its PWA [9] This is true even for the prototyping of the product. Consider the scenario where users are installing a new application which they have never heard of before. The reduction of friction between users and the application in such cases will increase the usage of the application. According to a case study, Tinder PWA increased the usage of the application when compared to a native application [9].

G. No third-party stores required

PWAs can be directly installed from the browser. This is advantageous as the app can be tested on real users without having to launch the application on the Apple app store or Google Play store. Currently, the App Store requires an annual fee of $99, accompanied by a long waiting time till the application gets accepted.

However, the most advantageous benefit of this feature is that the users are not required to be redirected to the stores to install an application. This makes the cost of attracting consumers to their mobile website becomes substantially lower than the cost of driving app downloads. Convincing users to install a PWA directly from a browser is simpler for the users than forcing them to go through the app store. App stores can distract users from the flow due to network download issues and large download times. App stores also provide alternative options to users which may include competitors. These disadvantages are prominent for startup applications where getting the user to use the application is of utmost importance. On the other hand, by using the PWA application, the time of the user does not get wasted. The PWA runs on the browser directly before installation, thus enabling the user to use the application before installation. Installation of the application and usage can occur simultaneously unlike native applications which need to be installed first before usage. According to a case study, the number of transactions for MishiPay increased 10 times by cutting the installation step required for native applications [9]. The process of native application installation is explained in the flowchart IV-G and the PWA installation process is explained in flowchart IV-G.

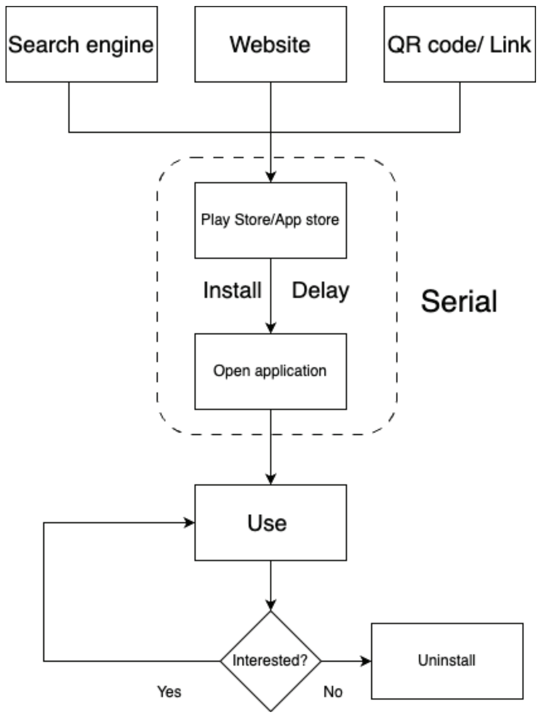

Fig. 3. Steps to install Native application

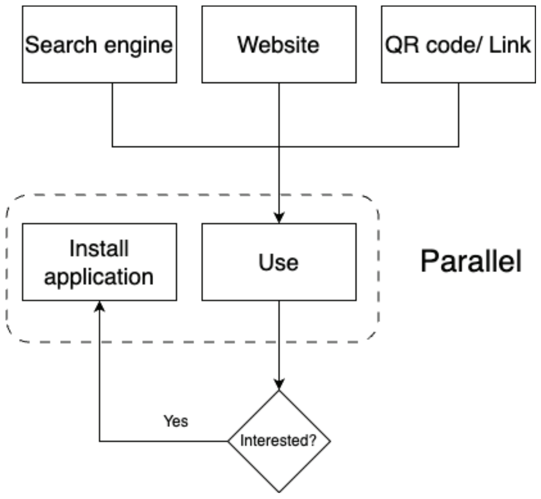

Fig. 4. Steps to install PWA

H. Other Advantages

1) PWAs are Discoverable: PWAs are easier to discover than native applications through mobile browsers. This is useful for scaling users after testing of idea.

2) Lesser Skill set required: PWAs require the same set as web development. PWAs can be built using HTML, CSS and Javascript. This reduces the cost of hiring. The 2022 survey for commonly used language among professional developers by stack exchange [8] shows 67.9% responses for Javascript 54.93% HTML and 40.08% for typescript. This is very high when compared to 33.4% for Java 9.92% for Kotlin, 6.67% for Dart and 5.18% for Swift.

3) Lesser Mobile Fragmentation: Native mobile applications running on different versions of operating systems makes native application development costly. This can be avoided in the testing phase by using progressive web applications. PWAs are affected by fragmentation to a lesser extent when compared to native applications [3].

V. Advantages of PWA over Web Apps

There are many case studies that describe how companies benefitted by converting their web applications to PWas. Goibibo improved its conversion rate by 60% by converting its mobile application to a PWA. Blibli improved its revenue by 10x by providing PWA for its' returning users. Nikkei saw 2.3x more organic traffic accompanied by 58% more conversions. Apart from these, many other enterprises like Infobae, voot have benefited from the conversion. In this section, we describe the advantages of PWA over web applications for new product testing. All these case studies are available at the Web.dev website [9].

A. Instability

PWAs look like Native apps on the app screen. This has the following advantages- Provides access point - The Icon in the app screen is an easy way to access the application users do not have to go to the webpage.
1) Reminds users - Acts as a constant reminder to use the app. This is not present in Web applications.
2) Notification remainder in the icon - The number of notifications of the application is clearly visible on the app icon attracting users' attention in case of any notification.
3) Application-like feel - there is very little difference between the feel of native applications and PWAs for the users. This is not present for mobile applications.
Currently, Android devices support the 'add to home screen' feature for any website. However, PWAs provide advantages like custom icons on the home screen, default names, notification icons and splash screens. One example of this is Rakuten 24. As a new service, Rakuten 24 increased its performance by 450% through the use of PWA. One of the key factors involved in this was its installability [9].

B. Visibility on Google Play Store

PWAs can also be visible on Google Play Store. This is useful in the later stages of testing when the application has been opened to public usage. Users can search for the application in the Play Store in a similar manner as any native application. Moreover, applications are indexed and can also be advertised in the Play Store in a similar manner as native applications. Currently, this feature is available only on the Microsoft Store and Google Play Store.

C. Offline performance

PWAs can Cache data and operate offline in a method similar to Native applications. This is useful for low bandwidth usage. JD.ID improved its mobile conversion rate by 53% with caching strategies, installation, and push notifications [9]. Frequently used data like images or home screen assets can be cached to make the application usable in offline mode.

D. Service workers.

Service workers are proxy servers that are intermediate between web applications, the browser, and the network [7]. It is a JavaScript file which runs separately from the browser. Service workers used in PWAs are capable of background processing. This ensures an instant and reliable experience for users, just like native applications. In Twitter PWA, fast user experience on first load and subsequent views enables users to view and create Tweets as quickly as possible [9].

VI. Experimental results

To validate the research findings the experimentation involved development of A Progressive Web application prototype for a university campus application. It involved a digital Identity Card Portal for security verification at a University that has over 40,000 students. In order to test the working of the same, a Progressive Web Application was developed. The following observations were found during the testing phase.
1) **Ease of Development** - The PWA prototype was easily developed by one single student with minimum skill requirements. The only skill required was HTML and Vanilla JS. The prototype was completed in 20 man-hours
2) **Multiplatform** - Single PWA was sufficient for both iPhone and Android users. Traditional applications could not have been tested on real iPhone users. It could also be run from any computer system in the campus.
3) **No Update required** - Iterative testing could be easily performed due to these features.
4) **Low size** and 5) **Lesser Loading time** enabled testing to be hassle-free on limited bandwidth areas of the campus. Students could use the application in 5-10 seconds only. On the other hand, installing a similar mobile application requires 2-3 minutes to install. The size of the PWA was only 2Mb while a similar mobile application takes 30 MB.
6) **No APK requirement** enabled students to easily access the application through QR codes. This enabled easier testing of the application. QR code leading to the website made the process of usage and downloading much easier than the traditional process of APK installation followed by usage. This benefit of likability and sharability was one of the most

profound in the entire process.
7) **Instability** Makes it easier for the students to use than a website as it can be easily opened directly from the home screen without much effort.
8) **Offline performance** enabled students to use the application without the requirement of connectivity.

In the above case study, progressive web applications played a vital role in the development of the digital Identity Card Portal. The PWA prototype has enabled the testing of the idea into a small sector of the university. This can be scaled up gradually and the results of its performance can be incorporated in the future versions of the product. A/B testing can be performed easily on various students to find out what they exactly need and improve the product. All of these activities would have taken more resources in terms on time and computations had it been a native or a web-only application.

Based on these experimental experiences, it can be concluded that PWA provides benefits of both speed of development similar to web applications with interface and features of native applications; it can be used for prototyping of native applications before they are built. PWAs can be used to create a working model of the final product in a smaller time and at a lower cost. It can be used to test and improve ideas at varying scales. The PWA approaches are advantageous over native applications especially in the scenario of new brand inceptions and resources are limited. They can be tremendously useful like the scenario discussed in the case study, where students/more users were responsible for the product development. Thus it can be concluded that PWAs are a very useful tool for bringing any new app-based product into the market.

VII. Conclusion and future scope

This paper proposes a new methodology that can speed up mobile application-based product testing. It also elaborates advantages of PWAs over traditional applications as well as web applications in the context of rapid prototyping. The proposed methodology reduces risks and time for finding product market fit. The progressive web applications for testing and prototyping of new products can be effectively used to lower the risk and failure costs of features in a product at its early stages. With PWA, the product development, product release and testing new features can take place concurrently, thus reducing the initial development costs. This has been made possible due to the novel feature of PWAs being easier and faster to develop, cross-platform and easy to deploy. Thus PWAs can prove a powerful tool for the rapid development of prototypes for especially for start-ups. Instead of investing time and resources in developing native applications, startups can now focus on iterating through their ideas at a minimal cost.

The future scope of this study consists of the study of scaling of PWA products. Various methodologies related to the scaling and native app transition phase of PWA prototypes can be investigated. Various aspects of non-LEAN methodology can be studied in the context of PWA applications. Awareness of this methodology needs to be done in startups. Lastly, process metrics need to be established for this methodology.

References

[1] The history of pwa development — the pwa book. https://www.divante.com/pwabook/chapter/02-the-history-of-pwas. (Accessed on 05/10/2023).
[2] How fast is pwa? check this pwa demo to figure it out - tigren. https://www.tigren.com/blog/pwa-demo/. (Accessed on 05/10/2023).
[3] Hybrid, native, and pwas: Testing your mobile apps for compatibility. https://www.browserstack.com/blog/hybrid-native-pwas-testing-your-mobile-apps-for-compatibility/. (Accessed on 05/10/2023).
[4] Progressive web apps. https://sites.google.com/view/progressivewebapps/home. (Accessed on 05/10/2023).
[5] Progressive web apps: Escaping tabs without losing our soul — by alex russell — medium. https://medium.com/@slightlylate/progressive-apps-escaping-tabs-without-losing-our-soul-3b93a8561955. (Accessed on 05/10/2023).
[6] Progressive web apps on ios are here — by maximiliano firtman (firt.dev) — medium. https://medium.com/@firt/progressive-web-apps-on-ios-are-here-d00430dee3a7. (Accessed on 05/10/2023).
[7] Service worker api - web apis — mdn. https://developer.mozilla.org/en-US/docs/Web/API/Service$_Worker_API.(Accessedon05/10/2023)$.
[8] Stack overflow developer survey 2022. https://survey.stackoverflow.co/2022/most-popular-technologies-language-prof. (Accessed on 05/10/2023).
[9] web.dev. https://web.dev/. (Accessed on 05/10/2023).
[10] Kashish Behl and Gaurav Raj. Architectural pattern of progressive web and background synchronization. In *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, pages 366–371. IEEE, 2018. doi:10.1109/ICACCE.2018.8441701.
[11] Andreas Biørn-Hansen, Tim A Majchrzak, and Tor-Morten Grønli. Progressive web apps: The possible web-native unifier for mobile development. In *WEBIST*, pages 344–351, 2017. doi:10.5220/0006353703440351.
[12] Andreas Biørn-Hansen, Tim A Majchrzak, and Tor-Morten Grønli. Progressive web apps for the unified development of mobile applications. In *Web Information Systems and Technologies: 13th International Conference, WEBIST 2017, Porto, Portugal, April 25–27, 2017, Revised Selected Papers 13*, pages 64–86. Springer, 2018. doi:10.1007/978-3-319-93527-0_4.
[13] Reza Fauzan, Ice Krisnahati, Bima Dinda Nurwibowo, and Della Aulia Wibowo. A systematic literature review on progressive web application practice and challenges. *IPTEK The Journal for Technology and Science*, 33(1):43–58, 2022. doi:10.12962/j20882033.v33i1.13904.
[14] Camille Fournier. Comparison of smoothness in progressive web apps and mobile applications on android, 2020.
[15] Bob Frankston. Progressive web apps [bits versus electrons]. *IEEE Consumer Electronics Magazine*, 7(2):106–117, 2018. doi:10.1109/MCE.2017.2776463.
[16] Tjarco Kerssens. Applicability of progressive web apps in mobile development, 2019.
[17] Narendra Lama. Providing native experiences in mobile with pwa. 2019.
[18] Jiyeon Lee, Hayeon Kim, Junghwan Park, Insik Shin, and Sooel Son. Pride and prejudice in progressive web apps: Abusing native app-like features in web applications. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1731–1746, 2018. doi:10.1145/3243734.3243867.
[19] Patrick V Mole and PV Mole. Progressive web apps: A novel way for cross-platform development. *no. September*, 2020.
[20] Dirk-Jan Rensema. The current state of progressive web apps: A study on the performance, compatibility, consistency, security and privacy, and user and business impact of progressive web apps, 2020.
[21] Dennis Sheppard and Dennis Sheppard. Beginning progressive web app development. 2017. doi:10.1007/978-1-4842-3090-9.
[22] Sayali Tandel and Abhishek Jamadar. Impact of progressive web apps on web app development. *International Journal of Innovative Research in Science, Engineering and Technology*, 7(9):9439–9444, 2018. doi:10.15680/IJIRSET.2018.0709021.