IJFEAT

INTERNATIONAL JOURNAL FOR ENGINEERING APPLICATIONS AND TECHNOLOGY

IMPACT OF PROGRESSIVE WEB APPS ON WEB APP DEVELOPMENT

Ms. Rakhi Bathiya[1], Mr. Pratik Ajmire[2], Ass. Prof. Mohit Popat[3]

[1]*Student, Department of Computer Science and Engineering, Jawaharlal Darda Institute of Engineering and Technology, Maharashtra, India,* ***rakhibathiya26@gmail.com***

[2]*Student, Department of Computer Science and Engineering, Jawaharlal Darda Institute of Engineering and Technology, Maharashtra, India,* ***pratikajmire9@gmail.com***

[3]*Assistant Professor, Department of Computer Science and Engineering, Jawaharlal Darda Institute of Engineering and Technology, Maharashtra, India,* ***mohit.popat@jdiet.ac.in***

Abstract

Innovation in technology is making an impact on how product and services are being designed. After the launching of Android OS especially smartphones were invented and now there is huge increase in use of smartphones. For browsing the contents most of the users use native mobile applications. The other way is to browse the contents is through a web browser. But both the ways have limitations. The first way which is the native app, user has to download the app firstly and then they must use it as per their requirements. This has two major disadvantages; one is it takes space on local storage of smartphone device and for operating it smoothly the network connection must be strong enough. The areas where 2G or lesser bandwidth or 3G network is available, it becomes a slow process when user access this native app. The second way which is through the web browser has disadvantages since the user experience is not as great as native app. Thus, order to overcome limitations, this paper is solution of above problem. Google has proposed a way to have one App be equal on both web and on mobile devices – Progressive Web Apps (PWA). Enabled for the most part by the Service Workers, APIs Progressive Web Apps (PWA) are a new class of Web applications. PWA are linkable with an URL which is fully responsive and secure. PWA is not required to be installed like a native App. where it is a website built using web technologies that acts like an App.

Index Terms: *Progressive, Offline, Service Worker, App Shell, App Manifest.*

---------- *** ----------

1. INTRODUCTION

In 2015, Google Chrome engineer Alex Russell and Designer Frances Berriman coined the term "progressive web apps" to describe apps taking advantage of new features supported by modern browsers, including service workers and web app manifests, that let users upgrade and update web apps to progressive web applications in their native operating system. Native mobile applications can also send push notifications, load on the home screen, work offline. Mobile Web Apps accessed in a mobile browser, by comparison, historically haven't done those things. Progressive Web Apps compatible with new Web APIs, new design concepts, and new buzzwords. Progressive Web Apps support features we expect from native apps to the mobile browser. Such that it uses standards-based technologies and run in a secure container which is accessible to anyone on the web. Even with the availability of fully developed mobile web application, it still struggles to provide an eye pleasing and satisfactory experience to the users mainly because of slow network connection across various areas. Therefore, PWA (Progressive Web Apps) is a new technology designed and developed by Google to overcome the limitation of native applications and mobile browsing. PWA is launched by tapping on an icon on the home screen of the device just like how one goes with native apps. PWA's are instantly loaded on your screen regardless of the network connectivity is available. They support the splash screen through push notifications. In the background, the service worker (set of APIs) allows developer to programmatically cache and preloaded assets and manages the data through a concept called push notifications. Service Worker is a module that runs its own thread and it is responsible to provide generalized entry points by which background task can be processed by PWA. PWA are linkable with an URL which is fully secure and responsive. Progressive Web Apps start out as tabs in Chrome and become progressively more "app" like; the more people use them, to the point where they can be pinned on the home screen of device or in the app drawer and have access to app-like properties such as push notifications and offline use.

2.LITERATURE REVIEW

Salma Charkaoui, Zakaria Adraoui and El Habib Benlahmar provided an insight into cross-platform development of mobile applications in their paper Cross-platform mobile development approache, where JavaScript frameworks were one of the approaches discussed [16]. Since this paper was published, the web has moved forward. For example, it states that a web application cannot access the Device API of a mobile device, which is not true anymore [17]. Some features are available to a web application today, such as camera, recording media and file access. This makes this research interesting today. According to Andre Charland and Brian Lerous, "Web apps are cheaper to develop and deploy than native apps". In their paper Mobile Application Development: Web vs. Native from 2011, they state the not so controversial fact that native applications are faster and have better user experience than web applications on mobile devices. They further talk about PhoneGap, a framework for building native mobile applications with web technologies, and how it has bridged the native and web environments, allowing web applications to live in a native environment. The rundown is that the web has not achieved the level of performance that native code provides, but it is getting close, giving the example of Quake 3 running in the browser [18]. In 2016 Jan Steczko wrote a thesis about companies, experiences with cross platform development compared to native development for mobile devices. Steczko interviewed 13 businesses in order to answer his problem. The thesis concludes that the companies preferred native development and that the advantages are stronger for native development. The companies thought that native applications were faster and could provide better user experience, and this compensates for the fact that creating two different applications for the mobile and the web are more expensive in both time and money. But the companies did also know that they could create cross-platform applications with lower cost and the development of these applications would be faster and simpler. According to Steczko's research, the choice between these two application strategies depended on several factors. For example, complexity of the application, budget or quality [19].

In 2014, the number of global users accessing the web on mobile devices surpassed those accessing it on a desktop [20]. This shows that making your web applications mobile-friendly is more important now than ever. Companies often see the need to develop native applications or hybrid applications to overcome the limitations that the web as a platform imposes on mobile devices. In many cases, they must develop their application for both the web, IOS and Android. A native application is generally coded in a device specific programming language and integrated development environment (IDE). While native Android application is usually coded in Java with the IDE Android Studio. These applications are generally installed through app stores on mobile phones and have rich access to device hardware through platform specific APIs. These applications are installable from the respective operating systems application store, and run inside a native environment, with all features available to a native application. Taken out of this native environment, these applications fail to deliver this experience due to browser constraints. Progressive Web Applications (PWA) could solve this problem. A hybrid application denotes an application built with web-based technologies but that can with the help of hybrid development frameworks (e.g., Apaches Cordova), appear and act as a native application. Therefore, when developing an application targeting mobiles, three common alternatives have traditionally been to build i.e. native, hybrid or mobile web applications; However, progressive web apps (PWAs), introduced by Google in 2015, can be considered a fourth alternative to these. A PWA is a web application that aims to deliver a native-like user experience on a mobile device, such as offline support and push notifications.

By taking this topic for research and building the progressive web app for an educational system we come across lot of background work created and conducted by various researchers worldwide [18]. As we are more familiar with the invent of using mobile web which exist for years as a subset of WWW which is really a slow and not so good interface on mobile phones. It has a support of WAP protocol and we generally load m.website.com pages on limited browser supported smartphone and tables which could not handle full web support. For a few years it looked like the old, dirty mobile Web was going to die. Adaptive and responsive design came to make full websites look good on mobile with rich and immersive experiences. The "mobile" bit was going to be stripped out and all we were left with was the Web, in all its glory, from any device we decide to access it. But it now looks like the mobile Web is making a comeback. Instead of breaking down barriers between the mobile Web and the full Web, a group of technology companies is working to try and make the mobile version of the Web faster. Native Apps on mobiles are fast whereas the mobile websites are comparatively slow. In 2016, this particular problem of Web and native App was prime conversation during all the discussion and conference. Researcher around the world was planning to launch a new way of programming which will help fill this gap of Web and Native Apps. Putting by a summary, PWA launches as a new tab in browser and progress similar to like "app" where most of the people are used for native app. We can have various pin points so that one can go to home screen or an application from the app drawer by using notifications and also by using offline access. Progressive Web apps (PWA) are just like native app in terms of security and full touch responses.

3. DETECTING PWA FEATURES

What exactly makes a Web app a Progressive Web App is not being defined clearly. One of the most open definitions comes from Samsung [1], maker of the Samsung Internet browser (emphasis ours): "Progressive Web Apps (PWAs) are regular mobile and desktop web applications that are accessible in any web browser. In browsers that support new open web standards they also provide additional capabilities including offline support and push notifications". Just like with Ajax[3], the term PWA became a catch-all umbrella brand for Web apps that in some way or the other use Service Worker APIs , feel (native) "app-like," use latest browser features if they are available(Progressive Enhancement[2]),or that can be installed

to the home screen. Russell [4] lists a number of requirements for what he calls "baseline appyness": "A Progressive Web App is functionally defined by the technical properties that allow the browser to detect that the site meets certain criteria and is worthy of being added to the home screen.

3.1 DETECTING SERVICE WORKER SUPPORT

A Service Worker is installed by calling the register method on the navigator object, whose first parameter is obligatory and contains a URL that points to a JavaScript file with the Service Worker code. The result of this promise-based API in the success case is then a Service Worker Registration object, which is either newly created if there was no previous Service Worker, or updated in the alternative case where a previous Service Worker existed [5]. In order to detect if a given Web View supports PWA features a tall, we can thus make a simple existence check for the API, and then try to register a Service Worker, as out lined in Listing.

3.2 CONSIDERED PROGRESSIVE WEBAPP FEATURE

Offline Capabilities: The ability to still load and work at least to some extent, even when the device is offline [5], for example, when the so-called air plane mode is activated or when the device temporarily has no network coverage.

Push Notifications: The capability to display push notifications as defined in the Push API [6], for example, to point users to fresh content, even when the app is not running.

Add to Home Screen: The capability to be installed (added) to a device's home screen for easy access as out lined in [7].

Background Sync: The capability to synchronize data in the background [5], for example, to send messages in a deferred way after a temporary offline situation in a chat app.

Navigation Preload: The capability to start network navigation request seven while the Service Worker has not booted yet [8], which would else be a blocking operation.

Storage Estimation: The capability to estimate the available storage that an application already uses and to know the available quota enforced by the browser [9].

Device Memory: The capability to read the amount of available Random-Access Memory (RAM) in Gigabyte of a device in order to allow servers to customize the app experience based on the built-in memory [10].

Safe: Progressive Web Apps are served via HTTPS, which ensures that without authentication it cannot be tampered by anyone.

4. CORE TENETS OF PWA

4.1 SERVICE WORKERS

Service Workers, are the powerful tool behind Progressive Web App. The features provided by service workers are as below:

- Offline Access.
- Push Notifications
- Content caching
- Background content updating

Following are the functionalities performed by Service workers:

1. Caches the App Shell.
2. Invalidates the cache when needed.
3. Gets the push notification id from the user to send the notification.
4. Updates the background content.

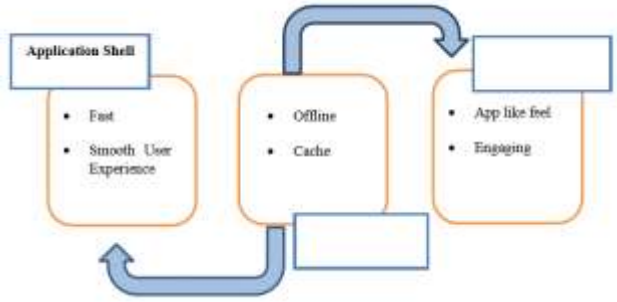


Fig-1: CORE TENETS OF PWA

4.2 APP SHELL

Application Shell Architecture is served up by the Service Workers and then the content is delivered. These are often cached by the service workers from its source through API requests. The sites that people visit more often will be able to hold the last content the person visited while waiting for the network to dynamically load the latest refresh. With the App Shell model, the focus is on keeping the shell of app UI and the content inside of it separate from each other, and they are cached separately. Ideally, App Shell is cached such that it loads as quickly as possible when a user visits the app and returns later. Having the shell and the content load separately, it theoretically improves the user's perception of the performance and usability of the app.

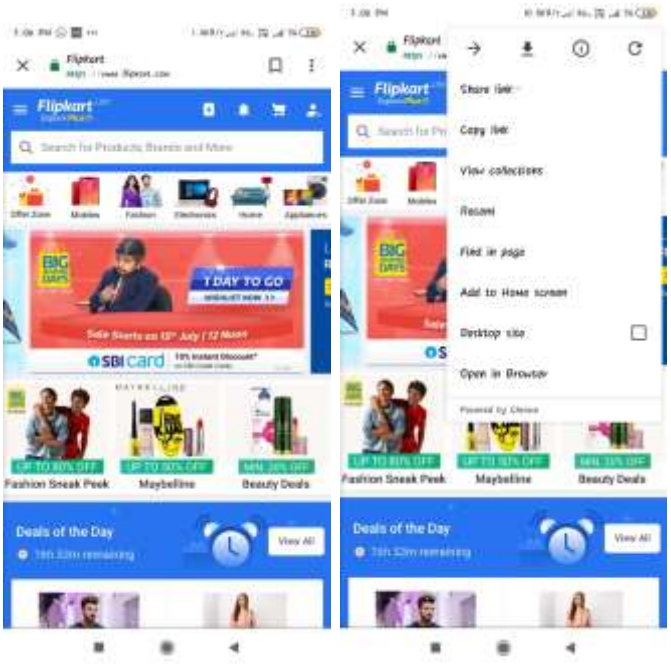
4.3 WEB APP MANIFEST

The web app manifest has the role to provide information about an app (such as its name, author, description and icon) in a JSON text file. The manifest must inform details for websites installed on the device's home screen, providing users with fast access and a richer experience. The collection of web technologies called progressive web apps includes Web app manifests as its part through which websites that can be installed to a home screen of the device without an app store, along with other capabilities like working offline and receiving push notifications.

5.APPLICATION OF PWA

Flipkart Lite, Progressive Web App (PWA) combines the best of the web and Flipkart native app. It leverages new, open web

APIs to offer a mobile web experience that loads fast, uses less data than before, and reengages users in multiple ways. Users visit via their browser and find a fast app-like user experience [14].



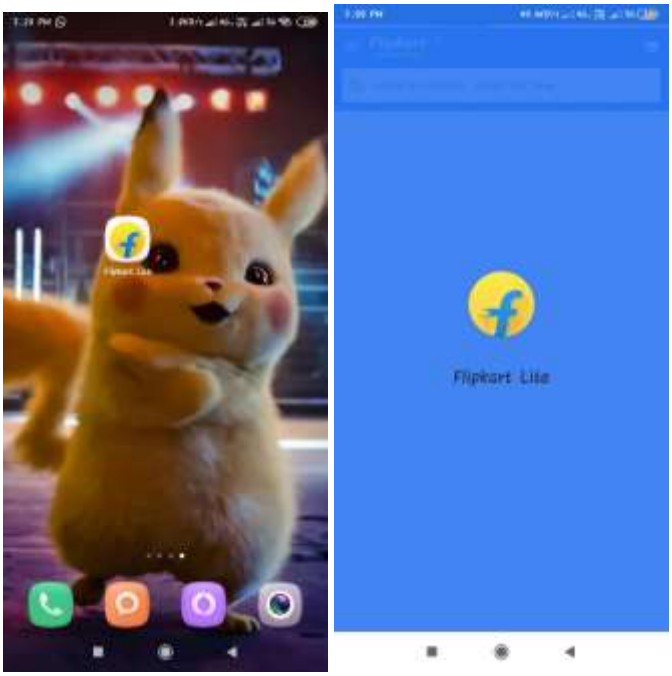


Figure 2: Screenshots showing some PWA features in action at the example of Flipkart (https://www.flipkart.com):(i)open with URL (ii) add to home screen prompt, (iii) icon on the home screen (iv) splash screen while launching, (v) launched in full screen mode without URL bar

5. CONCLUSION

Progressive web apps have benefits for everyone involved. The user will be able to instantly install the "app" without visit to the app store and a large download, which can be an unpleasant experience on a slow connection. Organizations can go back to developing web apps without separate Android and iOS teams. They can "update" and "release" their apps without going through the app store approval process. Releases and defect fixes can be deployed immediately in web app. Web design elements are immediately picked up by progressive web app. A progressive web app is a website that combines the best experiences of the web and an app. They don't require any installation. The app loads quickly, even when the user is on bad networks. It can send relevant push notifications to the user and has an icon on the home screen and loads as top level, full screen experience. Application shell architectures comes with several benefits but only makes sense for some classes of applications. The model is still young and it will be worth evaluating the effort and overall performance benefits of this architecture. Progressive web apps are an interesting forward look into the future of mobile apps. This will become a key factor in the world of apps.

REFERENCES

[1]Samsung. 2017. Progressive Web Apps. https://samsunginter.net/docs/ progressive-web-apps. (2017).

[2] Steve Champeon. 2003. Progressive Enhancement and the Future of Web Design. http://hesketh.com/publications/progressive_enhancement_and_the_future_of_web_design.html.(2003).

[3] Jesse James Garrett. 2005. Ajax: A New Approach to Web Applications. http://adaptivepath.org/ideas/ajax-new-approach-web-applications/.(2005).

[4] Alex Russell. 2016. What, Exactly, Makes Something A Progressive Web App? https://infrequently.org/2016/09/what-exactly-makes-somethinga-progressive-web-app/.(2016).

[5] Alex Russell, Jungkee Song, Jake Archibald, and Marijn Kruisselbrink. 2017. Service Workers 1. Editor's Draft, 22December2017.w3c.

[6] Peter Beverloo, MartinThomson, Michaëlvan Ouwerkerk, Bryan Sullivan, and Eduardo Fullea.2017. Push api. w3cEditor'sDraft15December2017.w3c.

[7] Paul Kinlan. 2017. The New and Improved Add to Home Screen. https: //developers.google.com/web/updates/2017/02/improved-add-to-home-screen. (2017).

[8] JakeArchibald.2017. Speed up Service Worker with Navigation Preloads. https://developers.google.com/web/updates/2017/02/navigation-preload.(2017).

[9] AnnevanKesteren.2018. Storage. Living Standard—LastUpdated9January2018. Whatwg

[10] Shubhie Panicker. 2017. Device Memory 1. Editor's Draft, 11 December 2017. wicg.

[11]Apple Developer Documentation. 2018. SFSafariViewController. https: //developer.apple.com/documentation/safariservices/sfsafariviewcontroller. (2018).

[12] Paul Kinlan. 2016. Chrome Custom Tabs. https://developer.chrome.com/ multidevice/android/customtabs.(2016).

[13] Niels Leenheer. 2017. About Chrome, ios and Payment Request. https: //nielsleenheer.com/articles/2017/about-chrome-ios-and-payment-request/. (2017).

[14]https://mobiforge.com/news-comment/progressiveweb-apps-are-future

[15] https://developers.google.com/web/showcase/2016/flipkart#tldr.

[16] Charkaoui, S., Adraoui, Z. and Benlahmar, E. (2014). "Cross-platform mobile development approaches". 2014 Third IEEE International Colloquium in Information Science and Technology (CIST).

[17] "What Web Can Do Today," What Web Can Do. [Online]. Available: https://whatwebcando.today/ Accessed: Mars 13, 2017

[18] Charland, A. and LeRoux, B. (2011). "Mobile Application Development: Web vs. Native". Queue, 9(4), p.20.

[19] Jan Steczko. "Analysis of companies' experience with cross-platform development compared to native development for mobile devices" in Lnu.diva-portal.org, Lnu Diva portal, 2016.