Progressive Web App for Educational System

Amit Mhaske[1], Aditya Bhattad[2], Priyanka Khamkar[3], Radhika More[4]

[1,2,3,4] *Student, Dept. of Computer Engineering ,NBN Sinhgad School of Engineering , Maharashtra, Pune*

---***---

Abstract: *With the invent of Smartphone especially after the launch of Android OS which is free the popularity of using smartphone is very huge. Close to 60% user across the globe are using smartphones and the transaction are increasing day by day. Most of the users use native mobile application to browse the contents of the particular industry. Another way is to browse the contents is through web browser. But both have the limitation. The first one which is the native app, user need to download the app first and then they have to use it as per their requirement. This has major two disadvantages, one is it takes some space on local smartphone device and to operate it smoothly the network connection has to be strong. It is a very slow process to access this native app where only 2G or lesser bandwidth 3G network is available. The second approach which is through the web browsing has disadvantages as the user experience is not that great compared to native app. To overcome above limitations, Google had provided a solution through a term called as Progressive Web App (PWA) which gives you a rich experience just like native apps. You don't require installing this PWA and can be operated through a splash screen. Our Paper majorly focuses on the building a progressive app for an educational system.*

Keywords: Progressive web app, Splash Screen, service work, network, native app experience, web library.

Introduction:

As we had discussed in beginning, the world is moving towards smartphone applications and total activity on mobile phone through web browsing or through app is almost close to 67% which the recent survey had identified. Most of the time for the user had been spend in browsing the site through the web browser of smartphone. But as discussed earlier because of the interface of most of website are not auto responsive in mobile web browser, the user doesn't feel good about the contents in browsing the application. Even though the mobile browser are trying to become full fleshed software platform but still as of today the mobile web application struggles to provide eye pleasing and satisfactory experience to the user mainly because of lack of strong network connection across the various area. That's why PWA (Progressive Web Apps) is a new technology designed and developed by Google to overcome the limitation of mobile browsing and native applications. PWA can be launched by clicking on an icon on the home screen of the device just like how one goes with native apps. PWA's get instantly loaded on your screen regardless what kind of network connectivity is available in your area. They support the splash screen through push notifications.

In background of PWA the service worker comes in picture which is a set of API which allows developer to programmatically cache and preloaded assets and manages the data through a concept called push notifications. Service Worker is a module which runs its own thread and it is responsible to provide generalized entry points by which PWA can process the background task.

The main characteristics and features of progressive web apps are as follows

- **Progressive -** Work for every user, regardless of browser choice because they're built with progressive enhancement as a core tenet.
- **Responsive -** Fit any form factor, desktop, mobile, tablet, or whatever is next.
- **Connectivity independent -** Enhanced with service workers to work offline or on low quality networks.
- **App-like -** Use the app-shell model to provide app-style navigations and interactions.
- **Fresh -** Always up-to-date thanks to the service worker update process.
- **Safe -** Served via TLS to prevent snooping and ensure content hasn't been tampered with.
- **Discoverable -** Are identifiable as "applications" thanks to W3C manifests and service worker registration scope allowing search engines to find them.
- **Re-engageable -** Make re-engagement easy through features like push notifications.
- **Installable -** Allow users to "keep" apps they find most useful on their home screen without the hassle of an app store.
- **Linkable -** Easily share via URL and not require complex installation.

In this paper we are putting focus on creating progressive web app for an Educational System.

Literature Review:

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.

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.

Following table summarizes the comparison between progressive web app, native app and standard web apps.

	Native App	**PWA**	**Standard web App**
Submitting the app	Need two developer accounts. One for the Play Store and one for the Apple Store	No developer account needed	No developer account needed
How to install	Need to go to the App store or Play Store, click download, enter password	Just click a button to add them to their phone home screen (only on Android)	No installation required
Size	Sometimes heavy. They can take a while to download on your users' phones	Very lightweight and fast	Very lightweight and fast
Updates	Need to be submitted to the store, then downloaded by the user	Instant updates	Instant updates
Offline access	Available	You need to use the app once online, then should be able to access the cached content offline	Not required
Launch in Full Screen	Yes	Yes	No
User experience	Excellent (when the app is designed well)	At times confusing because of the double menus (app menu and browser menu)	Same as progressive web apps
Engagement	Very high. People spend a lot more time in native apps as there are no distractions	Harder to keep people engaged. The app is like an extra open tab in the browser – it's very easy for people to switch	Same as progressive web apps
Discoverability	Not great – your users need to know the name of the app to find it, or you need to work hard on your app store optimization	Good – your app can appear in search results if you optimize it for SEO	Not required
Push notification	Yes	Yes(Android Only)	Yes(Possible with third party services)

Architecture:

There are various ways by which this all in approach of progressive web app model goes, but one of the most common ways is using Application Shell. This is not a hard requirement, but does come with several benefits.

The application shell architecture uses the user interface so that it can work offline and generates its own contents by using technology called JavaScript. When user goes on multiple repeat visits to same app then it gives u the meaningful pixel positions so that screen can be loaded fastly without the network. This is how we can increase the performance gains.

Service Workers:

Service workers are the main part of PWA which runs in background separately from the web pages. All the response to events, network request made from server and client is managed by service workers. The lifetime of Service workers are generally kept for a short time. It wakes up when it gets an event and runs only as long as it needs to process it.

The API which we are going to use in Service Workers is generally limited when we compare with the full functional JavaScript. This is standard for workers on the web. The DOM Stucture of a web page is not accessible by a Service worker can't access the DOM but can access things like the network request, fetch API and Cache API, The Indexed DB API and postMessage() are also available to use for data persistence and messaging between the service worker and pages it controls. Push events sent from your server can invoke the Notification API to increase user engagement. A service worker can intercept network requests made from a page (which triggers a fetch event on the service worker) and return a response retrieved from the network, or retrieved from a local cache, or even constructed programmatically. Effectively, it's a programmable proxy in the browser. The neat part is that, regardless of where the response comes from, it looks to the web page as though there were no service worker involvement. Service Workers are a way to increase Web app performance by helping to cache and deliver content and background functionality (like push notifications). Service workers can make sites work offline or help speed up the content by, "intercepting network requests to deliver programmatic or cached responses."

System Architecture:

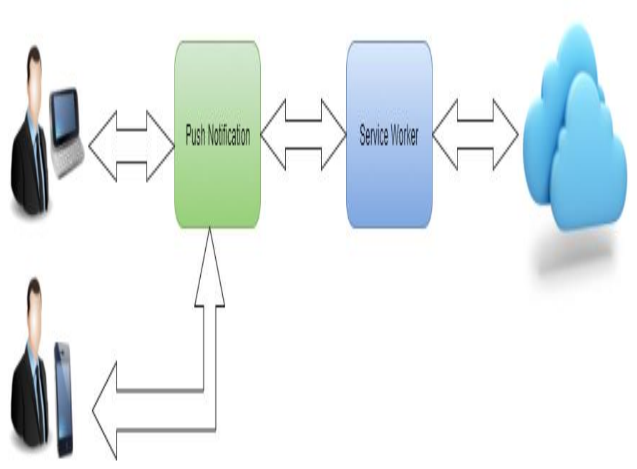

Figure 1: System Architecture of PWA for an Educational System

Conclusion:

Progressive web app is a mid-way approach for native app and web application. It reduces lot of burden of user about poor network connectivity and rich interface just like native 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. Progressive web apps are an interesting forward look into the future of mobile apps. It will become a key factor in the world of apps.

References:

[1]http://www.ness-ses.com/progressive-web-apps-the-new-future-of-mobile-apps/

[2]http://blog.cloudfour.com/android-instant-apps-progressive-web-apps-and-the-future-of-the-web

[3]https://arc.applause.com/2015/11/30/application-shell-architecture

[4]http://digiday.com/platforms/wtf-progressive-web-apps/

[5]https://developers.google.com/web/updates/2015/11/app-shell?hl=en [6]https://addyosmani.com/blog/getting-started-with-progressive-web-apps