LIVE COVID-19 TRACKER PWA APPLICATION

[1] Bhagyashri Satish Jorvekar ,PREC Loni, India
[2] Ankita Sunil Bhagwat, PREC Loni, India
[3] Shital Sanjay Magar, PREC Loni, India
[4] Pooja Dattatry Pulate, PREC Loni, India

ABSTRACT

Everyone is aware of COVID-19. One of the most and important times for people and countries who are fighting with this corona virus. So, this is our small initiative to bring awareness among people by showing the confirmed/active/recover/death cases of all the states and districts of India through our PWA (Angular support) application where data is updated every 15-30 minutes. So this is Our Small Application COVID-19 TRACKER.

.

Keyword : - *acquisition; pwa; pwa-app ;virus; fighting; death cases; updated; Live Covid_19 Tracker*

1. INTRODUCTION

Today, the pandemic made everyone's life painful. Everyone is trying to get subsistence and stay alive. In this regard, we are trying to develop a tracker using technology where Angular 10 becomes the useful one.

There is an application used for tracking the patient details on country-wise and another application is for tracking Indian state-wise and district-wise patient details. Using this application, we will track the state-wise patient details for a particular date with particular time. Moreover, we will try to showing the Indian map, where user can click on any of the states and get the patient details of that particular state.

We will also provide a list with all state names as a dropdown here. Moreover, any user can choose any particular area and get the details of that particular area i.e. state wise and district wise easily. Here we are using a free public API derived from the Covid-19India website for data and Angular 10 as a tracker builder.

Applications can also be mobile and available where needed rather than on a particular device. We can introduce place as its own attribute. Touching a surface in the living room would turn on the light in that room. When we place a phone call, we may be trying to reach a person, or perhaps we are trying to contact anyone who happens to be home at the time. There are many interaction modes, with touch being just one. Cameras can enable rich gesturing or even the use of facial expressions. Voice is another interactive mode, and there are so many more. PWAs now give us a chance to explore new possibilities. While a PWA can be treated like a standard application on a device, the ability for it to be run from a uniform resource locator (URL) makes it easy to use the application on any device with a browser. Today, airlines are removing screens from planes and expecting travelers to carry their own devices. Some airlines hand out tablets. Perhaps it will make sense to bring back the larger screens to connected devices with browsers.

2. LITERATURE SURVEY

Initial searches for academic involvement in progressive web apps returned a limited amount of results per January 2017. A search for Progressive Web Apps on Google Scholar resulted in two theses, a keynote paper for the Mobile! conference and an IEEE paper on web app launch times. Using the same search query, the Taylor & Francis Online article database returned zero results; as did Science Direct. The ACM Digital Library returned the aforementioned keynote paper. IEEE X explore contained the a fore mentioned launch time paper, which merely mentions PWA (Gudla et al., 2016). A Research Gate search covers both publications and questions on their website, but it rendered no relevant results. Academic contributions are, per January 2017, virtually non-existent or not visible nor findable in the search engines explored. An unknown base of research was identified for Service Workers. Searches for Service Workers among the aforementioned databases returned various results not constrained to computer science, but also within child care an the public sector. Other combined phrases, such as {Service Workers, web, API} did manage to filter the outcome to a certain degree, still polluted with irrelevant results. 2.2 Design Implementation To gain better understanding of the possibilities of progressive web apps, three technical artifacts were developed for comparison. An hybrid app was developed using the Ionic Framework. An interpreted app was developed using React Native. Lastly, a progressive Web App was developed using React.js. As our main aim is to introduce PWA as a concept together with some technical and higher-level comparisons, we excluded the aforementioned cross-compilation approach from our study. The applications employ a master-detail navigation pattern, where the master view presents a list of clickable items fetched from an API. Upon list item click, the app navigates to the detail view, displaying the item image together with its author and title. The data is fetched from www.reddit.com/r/Art/.json. All three technical artifacts have been opensourced1 to allow verification of the results. The artifacts are depicted in Figures 1, 2 and 3 (see p. 3).

1.2 Technologies and Concept

1.2.1. Framework Agnostic

A plethora of frameworks for web development exists (Smeets and Aerts, 2016). The Web Fundamentals group demonstrated framework agnostic by implementing PWAs in three different frameworks (Osmani, 2015).

1.2.2 Service Workers

The Service Worker 3 is responsible for most of the core features associated with progressive web apps (Gaunt, 2016). A PWA cannot correctly work in browsers without Service Worker support. The worker is enlistment on a user’s first page visit. It contains the JavaScript file embodying life cycle hooks for business logic and cache control. It can be used to handle tasks such as background synchronization (Archibald, 2016), caching mechanisms for data and application shell, as well as interception of network requests (Osmani and Gaunt, 2017).

1.2.3 Application Shell

The application shell is defined by the Google Web Fundamentals group as “[...] the minimal HTML, CSS, and JavaScript powering a user interface”. Osmani and Gaunt (2017). They list three criteria for the shell: fast loading time, cached, and displaying dynamic content. Data is pulled from external APIs.

1.2.4 Web App Manifest

The purpose of the manifest file is to expose certain modifiable settings to app developers. These settings contains such as logo image path, app name, splash screen and more. In short, the manifest can be used to modify behaviour and style of PWA applications.

4.2.5 Security through HTTPS

For security reasons, HTTPS is required for a Service Worker to register in the browser and accordingly act on events (Gaunt, 2016). The reason for enforced security is described by Gaunt , as using the "[...] service worker you can abduct connections, fabricate, and filter responses".

2.1 System

It includes client's side as well as server side Database. User works on following items of client's side
1. Home module
2. Graph Module
3. Vaccination Module
4. Help module

2.2 Project Design

To develop the problem under consideration and vindicate feasibility using concepts of knowledge canvas and IDEA Matrix for IDEA Matrix and Knowledge canvas canvas. IDEA Matrix is represented in the frontal form. Knowledge canvas represents about identification of opportunity for product. Feasibility is represented w.r.t. business perspective.

I	**D**	**E**	**A**
Increase	Drive	Educate	Accelerate
Improve	Deliver	Evaluate	Associate
Ignore	Decrease	Eliminate	Avoid

Where,

Improve=Result Satisfaction.

Decrease=Processing File.

Evaluate=Image interaction.

Associate=Analysis Image using machine learning.

3. Result

This section seeks to provide insights into differences between interpreted apps, progressive web apps, hybrid apps and native apps by comparing a set of features and concepts. It also presents insights on such as technical frameworks and experience-unification for end-users. Table 1 provides a non-exhaustive list of feature available in PWAs as of January 2017, along with their compatibility. Remarks follow subsequently

(a) The Enable improved add to Home screen developer flag in Chrome Canary for Android can be enabled in order for PWAs to be installed like normal apps (Joreteg, 2016).
(b) (b) PWAs will be made searchable from the Windows 10 app marketplace, thus becoming "first-class citizens" of their app ecosystem (Rossi, 2016).
(c) Push notification support through the Push API2 is available, but limited to certain browsers.
(d) As Apple's Safari browser does not yet support the Service Workers API, the IOS platform is yet to fully realized and leverage the potential of the technological advancement.
(e) A PWA can use HTML5-based APIs for hard ware and platform access in addition to features and functionality made possible by Service Workers.
(f) Hardware and platform access for Hybrid apps is usually provided by Cordova, a library for handling the bridging between a native app's web-view component and the device's APIs.

4. CONCLUSIONS

Thus, the above details explain how to design a simple Covid-19 tracker using Angular 10. It makes the sense that how the disease has spreads all over the world. The details give a summary of the data derived from the parts of the world. To learn more about technological developments in tracking the disease, one can opt for Angular Online Training. This Learning will help you more to understand it.

The current state of progressive web apps involves a lack of certain hardware and platform APIs and features that only (certain) cross-platform and native apps can access. Recent browser advancements have been forces of unification for the end-user app experience, including, but not limited to, installable and native looking web apps through PWA'S. While Chrome is leading the way for PWA browser support, Apple's IOS Safari is yet to support the necessary Service Worker API. We find that there is much potential for PWAs to become a unifier for web-native development without the use of cross-platform frameworks. As an end-user the PWA installation process becomes more similar to regular apps through new advancements in user experience aspects. Web apps can look, feel and act similar to native, hybrid and interpreted apps. While there are hardware and platform API limitations PWAs not found in the other approaches, product requirements and specification will in the end be the deciding factor for choice of approach. We would like to conclude with an encouraging note, a quote by Archibald (2016) form the 2016 Google I/O Conference". We can everything that ends up on the home screen to be competitive with native apps. We want to make the web a First-Cass part of the operating system in the operating system in the user's mind".

5. REFERENCES

[1] Archibald, J. (2016). Instant loading: Building offline-first progressive web apps.

[2] Ater, T. (2017). Building Progressive Web Apps: Bringing the Power of Native to the Browser. O'Reilly.

[3] Ciman, M. and Gaggi, O. (2016). An empirical analysis of energy consumption of cross-platform frameworks for mobile development. Pervasive and Mobile Computing.

[4] Corral, L., Janes, A., and Remencius, T. (2012). Potential advantages and disadvantages of multiplatform development Frameworks–A vision on mobile environments. In Procedia Computer Science, volume 10, pages 1202– 1207. SciVerseScienceDirect.

[5] Dalmasso, I., Datta, S. K., Bonnet, C., and Nikaein, N. (2013). Survey, comparison and evaluation of cross platform mobile application development tools. In Proc. 9th (IWCMC), pages 323–328.

[6] Edwards, A. R. (2016). The building blocks of progressive web apps – smashing magazine.

[7] Gaouar, L., Benamar, A., and Bendimerad, F. T. (2016). Desirable requirements of cross platform mobile development tools. Electronic Devices, 5:14–22.