From Native to Progressive Web App
A Proof of Concept

Gunharth Randolf
1810738273
Entwicklung & Betrieb Mobiler Informationssysteme
Web Communication & Information Systems
29. July 2019

Abstract

Lorem ipsum

Contents

1. Introduction

Outline: Why was the study undertaken? What was the research question, the tested hypothesis or the purpose of the research? "What?" and "So what?" What is the paper about, and why should the reader care?

In 2007 announces Steve Jobs iphone ...

Latest advances in web technologies due to the support by Google pushing the support.

Recent applications like the one from Twitter

What are progressive web apps.

Motivation Liebel (2019)

The ACM Digital Library only shows 6 search results since 2017 A search on Google scholar revealed All articles related to technology side of things. Surprisingly, I was not able to find any entries that focussed on the business side of things. Just looking at the costs of developing and maintaining native or hybrid apps

Developing a native application is an expensive solution. Programmers spend years learning to code heavy native apps and app owners usually invest a huge budget finding a development team. Therefore, an answer to this persistent problem is enhancing traditional web app to a Progressive Web App (PWA). (Nguyen, 2019)

2. Methods

When, where, and how was the study done? What materials were used or who was included in the study groups (patients, etc.)?

This study implements three research methods to gain a broader understanding of the possibilities, state of research and technological practices regarding Progressive Web Apps.

2.1 Literature Search

A search for "Progressive Web Apps" in Google Scholar returns

Notably, there is a book by liebl in German

Interestingly, there seems to be no study available yet as Further reserach will have to

2.2 Related Works

The Google Web Fundamentals group acts as the driving force behind the documentation on PWA related topics and the creation of blog posts and tutorials. As the supporting technologies advance with every Chrome browser release, the Web Fundamentals website[1] is the foundation for upcoming research and studies.

[1] https://developers.google.com/web/fundamentals

Other than Google-created content, online platforms such as Medium blog posts[2] and Youtube tutorials[3] lay the foundation of technical research for this study.

2.3 Technical Implementation

To gain a better understanding of the possibilities of Progressive Web Apps a PWA was developed. The motivation for the PWA was to take an existing native application as a reference and re-create the application as a Progressive Web App. For this purpose I picked the *Beer With Me*[4] application, which is available on the Apple App Store as well as on Google Play. As stated by the Swedish developers Antonsson, Knutsson and Tidbeck "Beer With Me is a social application to notify your friends when you are drinking so that they can join".

[2] https://medium.com/search?q=Progressive%20Web%20Apps
[3] https://www.youtube.com/results?search_query=Progressive+Web+Apps
[4] https://beerwithme.se/

3. Concepts & Technical Implementation

This chapter provides a detailed look at the required concepts and the technical implementation of the PWA. The project has been open-sourced to allow verification of the results[1]. Further, the application is available online[2].

3.1 Requirements

The following section outlines the concepts and requirements used for the project by looking at the core functionality of the original *Beer with Me* app.

Progressive Web App. The functional clone of the *Beer with Me* app must implement all relevant techniques as outlined by the Progressive Web App specifications. Technologies include Service Workers, Application Shell (AppShell), Web App Manifest and serving the demo installation over HTTPS. Further, it is the intention to follow the 10 principal concepts, which lay the foundation for Progressive Web Applications, as closely as possible: Progressive, Responsive, Installable, Connectivity Independent, App-Like, Fresh, Safe, Discoverable, Re-engageable and Linkable (Osmani, 2015).

Authentication and Authorization. Users must be able to register for the service with a combination of E-Mail and Password. As an alternative an existing Google account may be used in order to login directly.

Realtime Database. Data is stored as JSON in a NoSQL database and synchronized in realtime to every connected client.

[1] https://github.com/gunharth/bwm
[2] https://bwm.gunicode.com

Realtime Notifications. If enabled by the user push-notifications shall be sent to the user in realtime, whenever a new user registers for the service or when a new post was published by a user.

Realtime Map. If enabled by the user his/her current geolocation is saved and displayed on an online map.

Single Page App (SPA). The app shall be implemented as a Single Page App with one of the main JavaScript frameworks (React, Angular or Vue.js).

Option to add a photo using the device's camera

Frontend, Backend framework/system straight forward deployment process

As part of this study the following points were defined as being not part of the requirements, as they do not have an impact on the used technologies of the PWA, nor do they support the output of this research. Design considerations of the application can be neglected, nor is the intention to copy the interface and design of the original *Beer with Me* app. The resulting PWA focusses on the implementation of the functionality, and not to offer a duplication of the original. Further, all functions are implemented as a proof-of-concept. Thus, to make it a production ready PWA certain functionality is not implemented, like the creation of groups. The PWA supports one group only for testing purposes.

3.2 Technical Implementation of Requirements

As per the actual implementation of the PWA the goal was to base the solution on the least different providers on the one hand, as well as a minimum in terms of coding and programming languages. Hence, the front-end of the application uses the basic building blocks of regular web sites being HTML, CSS and JavaScript. To satisfy the back-end requirements A cloud solution provider

3.2.1 Single Page App (SPA) & Progressive Web App

For the development of the frontend the JavaScript framework Vue.js[3] and its ecosystem was used. The Vue Cli[4] offers great tooling support for Vue.js developments - among other it comes with a preset for installing a Progressive Web App skeleton.

Listing 1: Installation and project creation commands with the Vue Cli

```
1 $ npm install -g @vue/cli
2 $ vue create beerwithme
```

On project creation Vue Cli offers you to manually select the features that will be installed. Next to Progressive Web Application support the Vue Router and Vuex was installed. The Router enables navigation services within a SPA and Vuex is responsible for the management of state within the application.

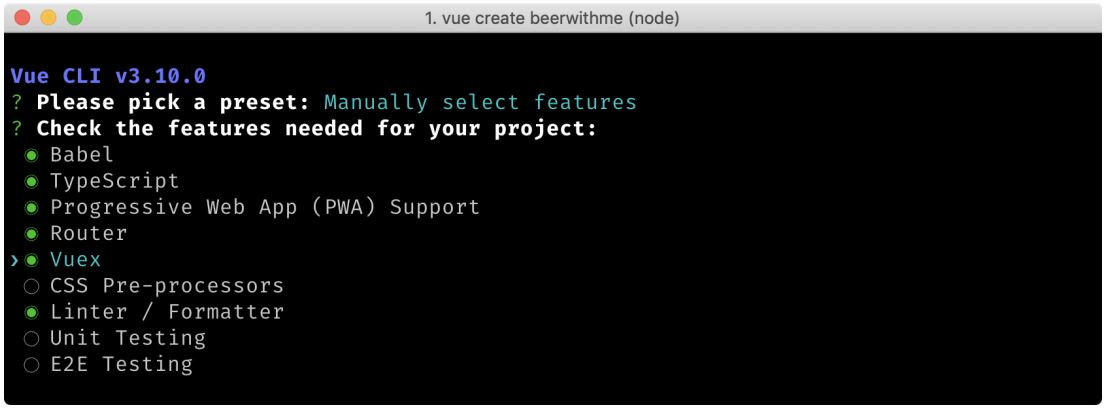


Figure 1: Installation options for a Vue.js project

The Vue Cli PWA support comes with Workbox[5] installed - a library developed by Google for adding offline support to web applications. Among other features it mainly helps with the automatic creation and management of application caches, service worker and the creation of the web manifest for a production environment.

Listing 2: registerServiceWorker.js: Workbox registering the service worker for production

[3] https://vuejs.org
[4] https://cli.vuejs.org
[5] https://developers.google.com/web/tools/workbox

```
import { register } from "register-service-worker";
if (process.env.NODE_ENV === "production") {
  register(`${process.env.BASE_URL}firebase-messaging-sw.
     js`, {
    ready() {
      console.log(
        "App is being served from cache by a service
           worker.\n" +
          "For more details, visit https://goo.gl/AFskqB"
      );
    },
    registered() {
      console.log("Service worker has been registered.");
    },
    cached() {
      console.log("Content has been cached for offline
         use.");
    },
    updatefound() {
      console.log("New content is downloading.");
    },
    updated() {
      console.log("New content is available; please
         refresh.");
    },
    offline() {
      console.log(
        "No internet connection found. App is running in
           offline mode."
      );
    },
    error(error) {
      console.error("Error during service worker
         registration:", error);
    }
  });
}
```

The final lighthouse report for the *Beer with Me PWA*:

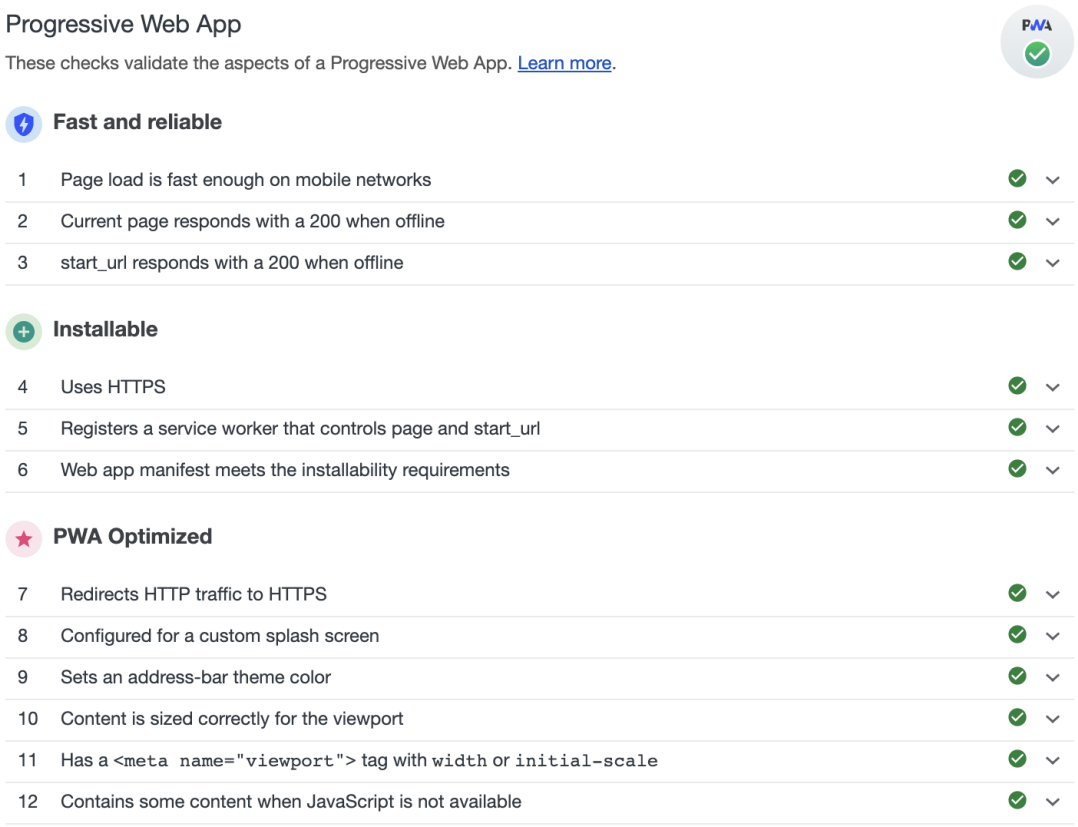


Figure 2: Lighthouse PWA report

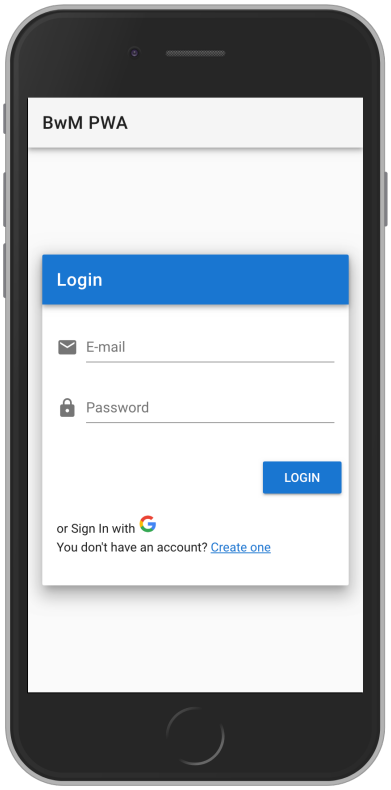
For the front-end design of the app Vuetify[6] was implemented. A design component framework for Vue.js based on the material design[7] guidelines developed by Google.

Listing 3: Command to add Vuetify to the Vue.js project

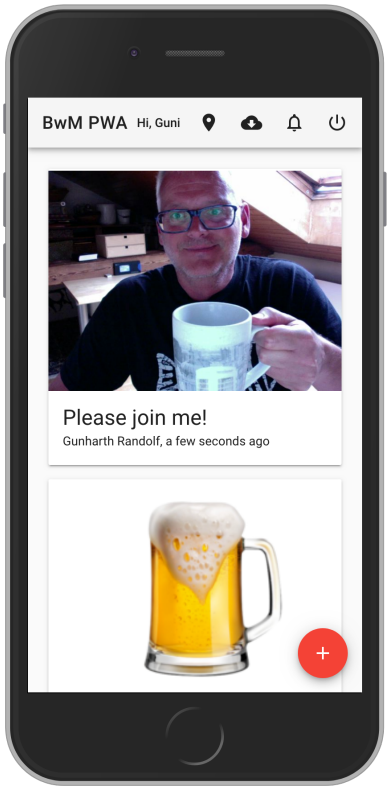
```
1 $ vue add vuetify
```

The map functionality was implemented by using the npm package vue2-leaflet[8], which is a Vue wrapper library for the open-source JavaScript library Leaflet[9]. Map tiles are served from OpenStreetMap[10].

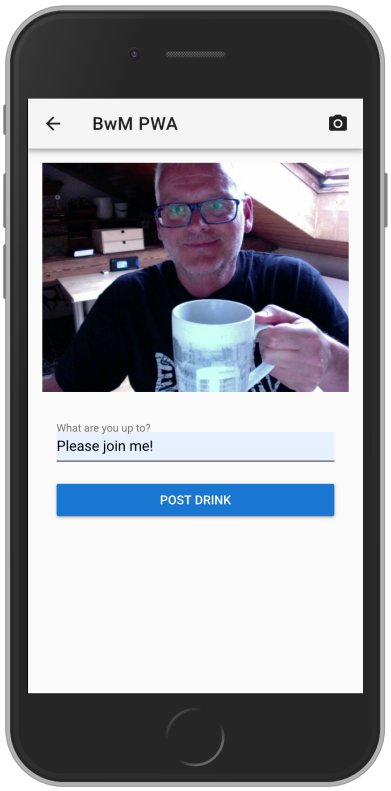
[6] https://vuetifyjs.com
[7] https://material.io
[8] https://www.npmjs.com/package/vue2-leaflet
[9] https://leafletjs.com
[10] https://www.openstreetmap.org



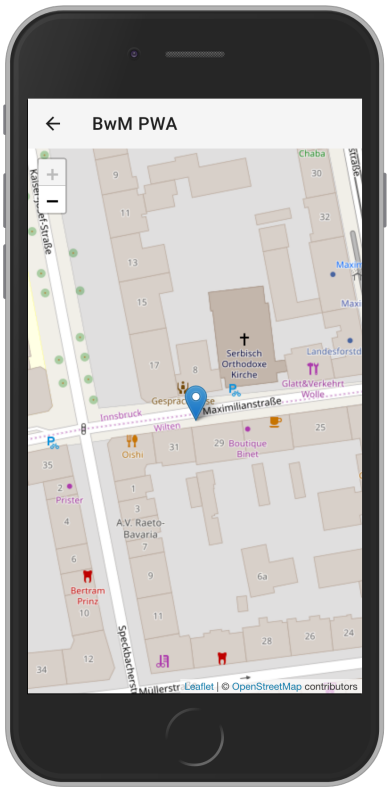
(a) Login screen



(b) Main app screen



(c) New post screen with photo option



(d) The map

Figure 3: Screenshots showing Vuetifys' material design components

3.2.2 Cloud Provider and Realtime Functionalities

For the purpose of the backend requirements Firebase was used. Google aquired it in 2015 ... The Firebase service offers the following services that are required for the Beer with me project: Authentication and Authorization, Firestore offers a real time database

Authentication and Authorization

Listing 4: main.js: Firebase Auth initiation using VueJS

```
1  // Init firebase auth before Vue inits the App
2  firebase.auth.onAuthStateChanged(firebaseUser => {
3    if (firebaseUser) {
4      store.dispatch("autoSignIn", firebaseUser);
5    }
6    if (!app) {
7      app = new Vue({
8        router,
9        store,
10       render: h => h(App)
11     }).$mount("#app");
12   }
13 });
```

Realtime Database feeding the feed of posts on the main application page. Map positions

Listing 5: Realtime query for new posts (Home.vue)

```
1  firebase.db
2    .collection("drinks")
3    .orderBy("created_at", "desc")
4    .onSnapshot(snapShot => {
5      this.drinks = [];
6      snapShot.forEach(drink => {
7        this.drinks.push({
8          id: drink.id,
9          url: drink.data().url,
10         comment: drink.data().comment,
11         author: drink.data().author,
12         created_at: drink.data().created_at,
```

```
13        });
14      });
15    });
```

Realtime Push Notifications

Firebase. Firebase is a service that Google aquired in 2015 ties in hand in hand with their goal on pushing on

4. Results

What answer was found to the research question; what did the study find? Was the tested hypothesis true?

5. Discussion

Limitations, next steps , ...

What might the answer imply and why does it matter? How does it fit in with what other researchers have found? What are the perspectives for future research?

6. Conclusion

Limitations, next steps , ...

What might the answer imply and why does it matter? How does it fit in with what other researchers have found? What are the perspectives for future research?

Bibliography

Liebel, C. (2019). *Progressive Web Apps: das Praxisbuch*. Rheinwerk Computing. Rheinwerk Verlag, Bonn, 1 edition.

Nguyen, P. H. (2019). Progressive Web App in Enhancing App Experience: Life Care Insurance Claim Application.

Osmani, A. (2015). Getting started with Progressive Web Apps. https://addyosmani.com/blog/getting-started-with-progressive-web-apps/. Retrieved 2019-06-08.