De interactie van een progressive web application met het besturingssysteem: een vergelijkende studie en proof-of-concept

Onderzoeksvoorstel Bachelorproef 2019-2020

Tijs Martens[1]

Samenvatting
Er zal onderzoek gedaan worden naar de huidige staat van PWA's (progressive web applications) en hoe progressive web apps gebruik kunnen maken van functies van een besturingssysteem. In een eerste deel zal gekeken worden naar welke functies besturingssystemen beschikbaar stellen voor progressive web applications en welke functies een native app wel kan gebruiken maar een PWA niet. Vervolgens zal er naar een oplossing gezocht worden voor de functies van een besturingssysteem waar een PWA geen toegang tot heeft. Voor het derde deel zal een vergelijkende studie gevoerd worden naar de state of the art van PWA's op de verschillende besturingssystemen ten opzichte van native apps.
Dit onderzoek is relevant omdat het ontwikkelen van PWA's een stuk sneller en goedkoper is dan het ontwikkelen van native applicaties. Door het in kaart brengen van wat mogelijk is en wat niet kan er beter beslist worden wanneer er een PWA kan gebruikt worden in plaats van een native app. Op deze manier kan er voor sommige bedrijven tijd en budget gespaard worden zonder kwaliteit in te leveren. De verwachting is dat voor veel use-cases een PWA volstaat en er geen ontwikkeling van native apps voor verschillende platformen nodig is.
Simon Floré van experience design studio Bothrs zal me bijstaan in dit onderzoek.
Sleutelwoorden
Webapplicatieontwikkeling. PWA — native apps — besturingssystemen

Co-promotor
Simon Floré[2] (Bothrs)

Contact: [1] tijs.martens@student.hogent.be; [2] simon@bothrs.com;

Inhoudsopgave

1. Introductie

Op een desktop is het gebruikelijk om taken uit te voeren in de browser. Voorbeelden hiervan zijn Gmail en Google drive. Dit is echter nog niet het geval op mobiele toestellen. Elke online service, waarbij mobiele gebruikers een doelpubliek zijn, heeft een native mobiele applicatie nodig. Vaak volstaat een traditionele responsive website niet omdat dit essentiële functies zoals push notificaties ontbreekt. Dit heeft als gevolg dat gebruikers minder snel terugkeren naar jouw website. (Hiltunen, 2018)

De ontwikkeling van native mobiele applicaties is echter geen goedkope of snelle oplossing. Er moet gelijkaardige code geschreven worden voor meerdere platformen. Digitale agentschappen en andere software ontwikkeling bedrijven willen vaak een zo snel mogelijke service bieden aan hun klanten, dit is momenteel moeilijk.

Progressive web apps kunnen voor deze problemen een oplossing bieden. Bij het ontwikkelen van een mobiele applicatie zijn vaak functies, die aangeboden worden door het besturingssysteem, nodig. Voorbeelden hiervan zijn zijn: locatievoorzieningen, offline gebruik, camera,push notificaties, …

In deze thesis zullen volgende onderzoeksvragen onderzocht worden

- Welke stappen zijn nodig om een een traditionele website om te vormen tot een progressive web application?

- Wat zijn de beperkingen van een progressive web app?
- Hoe kan een PWA alle functionaliteiten gebruiken die beschikbaar zijn voor native applicaties?
- Hoe staan de verschillende besturingssystemen ten opzichte van progressive web apps?
- Welke andere technologieën kunnen er gebruikt worden om applicaties te ontwikkelen voor meerdere platformen waarbij er maar 1 codebase is?

2. State-of-the-art

2.1 Wat is een PWA

Een PWA is een website gebouwd met web technologieën die zich gedraagt als een native app maar waarbij er niet door het installatie proces moet gegaan worden zoals bij native applicaties. (Tandel & Jamader, 2018)

Een PWA is een 'enhanced website', dit is een website met een paar extra bestanden die er voor zorgen dat de site extra functionaliteiten heeft. Volgende bestanden zijn nodig om een van een website een PWA te maken

- Manifest.json – Dit is een bestand waar je enkele eigenschappen van de applicatie instelt zoals: app icoon, startpagina, kleurschema, ...
- Service worker – Dit is een bestand waarbij je zelf caching kan doen. Hierdoor kan, eens een website geladen is geweest, de site offline gebruikt worden.

(Harris, 2017)

2.2 Welke functies van een besturingssysteem kan een PWA gebruiken

Niet alle besturingssystemen geven evenveel functies die gebruikt kunnen worden vanuit een PWA: IOS and Android stellen volgende functies ter beschikking

- Notificaties op het toestel
- Locatie voorziening
- Camera
- Gebruik van de sensoren van het toestel
- Audio output
- Betaling systemen (Android pay bij Android, Apple pay bij IOS)
- Spraakinput
- Bluetooth (enkel Android)

Andere functies waar native apps wel toegang tot hebben zijn niet beschikbaar voor PWA's

- Toegang tot contacten
- Toegang tot de kalender
- Toegang tot alarmen
- Toegang tot telefonie data
- Berichten
- Bel functie
- Toegang tot low level hardware sensoren
- Camera (video's)
- Maximum storage van 50Mb op IOS
- Geen widgets

(Ivano, 2016) (Destrebecq, 2019)

2.3 Waarom progressive web applications

De verwachtingen die een gebruiker heeft van een website of mobiele service zijn hoger dan ooit. Als je website niet binnen 3 seconden geladen is zal je al 53van de gebruikers verliezen. Dit kan voorkomen worden door progressive web apps. Als de website één maal geladen is geweest kan de site opgeslagen worden in het cache geheugen. Dit zorgt voor een snellere ervaring. (Google & awwwards, 2017)

Gebruikers raken gerustreerd als het gedrag van een service anders is op de verschillende platformen. Dit verschil komt er omdat dit totaal verschillende code-bases zijn, die vaak door verschillende teams ontwikkeld en onderhouden worden. Met progressive web applicaties wordt er 1 codebase geschreven die op alle platformen gebruikt wordt. Hierdoor zal het systeem op elk platform heel gelijkaardig werken. (Google e.a., 2019)

Volgens Google zijn er 3 grote redenen om over te schakelen naar PWA's

- Betrouwbaarheid – Door het cache geheugen zal een gebruiker nooit een pagina terechtkomen met een melding dat er geen internet connectie is.
- Snelheid – Een PWA kan sneller zijn dan een gewone website door het gebruik van cache storage. Een PWA kan sneller zijn dan een native app doordat het een veel kleiner bestand is.
- Aantrekkelijkheid – een PWA kan aanvoelen als een native applicatie.

(Google, 2019)

2.4 Beveiliging

Een Progressive web application moet altijd een HTTPS verbinding hebben. Dit wil zeggen dat de data die tussen de client en de server verstuurd wordt versleuteld is. (Zakir e.a., 2013) Dit is een stap in de juiste richting maar het zorgt er dus niet voor dat elke PWA een veilige applicatie is. PWA's worden vandaag gebruikt om gebruikers op te lichten. Een vaak gebruikte techniek is dat een bestaande native applicatie wordt nagemaakt. Op deze manier proberen ze wachtwoorden en betalingsdetails te verkrijgen. (Lee e.a., 2018)

2.5 Native containers

Niet alle problemen kunnen opgelost worden via het web de dag van vandaag. Voor sommige taken zijn nog steeds native applicaties nodig. Als er een PWA moet gemaakt worden met de functies van een native applicatie, kan deze PWA ontsloten worden door een native wrapper. Een nadeel hiervan is dat de grootte van een PWA drastisch verhoogt. Een ander nadeel is dat er twee of meer codebases onderhouden moeten worden, één van de site en minstens één van de wrappers. Deze websites zijn dan niet meer beschikbaar via de browser en moetn geïnstalleerd worden via een app-store. Dit wordt ook wel 'web based hybrid mobile app' genoemd. (Richard, 2019) (Ivano, 2016)

2.6 Apllication shell architecture

Om een snelle ervaring te bieden aan de eindgebruiker moet een applicatie opgedeeld worden in twee delen: de inhoud en de 'applicaion shell' De application shell is het deel van de interface die op elke pagina terugkomt, dit kan bestaan uit achtergronden, navigatie, ... Deze application shell moet volledig lokaal opgeslagen worden zodat deze niet steeds opnieuw gedownload moet worden. (Hiltunen, 2018)

3. Methodologie

Het onderzoek zal uit 3 delen bestaan.

Een eerste deel waar ik een PWA maak die gebruik maakt van de functionaliteiten van een besturingssysteem die aanspreekbaar zijn vanuit een progressive web app.

In het tweede luik van het onderzoek zal er gezocht worden naar hoe de functies die normaal enkel toegangkelijk zijn vanuit een native app toch gebruikt kunnen worden binenen een PWA. Voorbeelden hiervan zijn bluetooth, NFC, ... Er zal een PWA ontwikkeld worden waar deze functionaliteiten gebruikt worden door deze progressive web app te ontsluiten in een Native shell.

Na het voeren van deze twee praktische onderzoeken zullen er ook theoretische onderzoeken gevoerd worden. Er zal bekeken worden welke besturingssystemen de meeste functionaliteiten bieden waar progressive web apps gebruik kunnen van maken. Er zal ook in kaart gebracht worden wat de zwaktes en limiterende factoren zijn van een PWA ten opzichten van een native applicatie. In een laatste deel van het onderzoek zal er gezocht worden naar andere technologieën waarbij applicaties kunnen ontwikkeld worden voor meerdere platformen met één codebase. De ervaring die deze technologieën bieden voor de eindgebruiker en de ontwikkelaar zullen bekeken worden.

3.1 Technologieën

Progressive web apps kunnen gemaakt worden met de tools die gebruikt worden om traditionele webapplicaties te bouwen. Voor de onderzoeken zal voor de eerste fase HTML, CSS en JavaScript gebruikt worden. Eventueel kan er ook gebruik gemaakt worden van een javascript library zoals React. Er zijn verschillende tools beschikbaar voor het creëren van PWA's één van deze tools is Ionic. Voor het maken van een native shell zal er waarschijnlijk gebruik gemaakt worden van react native. Dit is een framework waarbij één codebase gebruikt wordt om apps te ontwikkelen voor IOS en Android. Hier moet er nog steeds vaak gelijkaardige code geschreven worden om de code te laten werken op de verschillende platformen.

4. Verwachte resultaten

Een website maken die voldoet aan de normen van een PWA zal niet veel tijd in beslag nemen. Er moeten slechts twee bestanden toegevoegd worden: het app-manifest bestand en een serviceworker. Het app-manifest bestand bepaald de look en feel van de applicatie eens deze geïnstalleerd is. De serviceworker zorgt voor het cachen van data afkomstig van een API. Echter, om aan de normen van een PWA te voldoen moet deze file gewoon aanwezig zijn. Als dit het geval is, kan de website geïnstalleerd worden op het toestel van de eindgebruiker en voldoet de website dus aan de normen van een PWA.

Het effectief gebruik maken van de functionaliteiten die een serviceworker kan bieden kent een steilere leercurve. De ontwikkelaar moet verschillende, complexe concepten begrijpen en kunnen toepassen. Een voorbeeld hiervan is de levenscyclus die een serviceworker doorloopt, dit is belangrijk voor het updaten van de inhoud van de website op basis van het cache geheugen of de backend. Om een goede offline ervaring te bieden zal het ontwerp van de webapplicatie ook herzien moeten worden. Er moet een application shell gedefinieerd worden. (Gaunt, 2019) (Osmani, 2016)

Er wordt verwacht dat er bij de besturingssystemen een duidelijk verschil zal zijn. Google, de ontwikkelaar van Android, heeft de revolutie die progressive web applications wel zijn gestart. Ze proberen zoveel mogelijk ontwikkelaars te overtuigen om PWA's te maken. Hierdoor wordt er verwacht dat Android meer functies van het besturingssysteem open zal stellen voor PWA's. Het is daarentegen bekend dat Apple toegang tot het besturingssysteem zoveel mogelijk wil beperken. Een duidelijk verschil tussen deze twee besturingssystemen wordt verwacht. (Biorn-Hansen e.a., 2017)

5. Verwachte conclusies

Native applicaties zullen waarschijnlijk niet snel vervangen worden. Ze bieden nog steeds de meeste flexibiliteit. Maar native applicaties bouwen is tijdrovend en duur. Voor projecten met een kleiner budget of een scherpe deadline zullen PWA's de markt domineren. Progressive web apps bieden vandaag al de snelste ervaring maar er ontbreken nog functionaliteiten. Er wordt verwacht dat dit de komende jaren zal verbeteren.

Referenties

Biorn-Hansen, A., Majchrzak, T. A. & Gronli, T.-M. (2017). Progressive Web Apps: The Possible Web-native Unifier for Mobile Development. *SCITEPRESS*. https://doi.org/10.5220/0006353703440351

Destrebecq, O. (2019, februari 12). *For your MVP: PWA or mobile application?* https://medium.com/swlh/pwa-or-mobile-application-d26e904e814e

Gaunt, M. (2019). *Service wokers: an introduction*. https://developers.google.com/web/fundamentals/primers/service-workers

Google. (2019). *Progressive Web Apps*. https://developers.google.com/web/progressive-web-apps

Google & awwwards. (2017). *speed matters*. https://www.awwwards.com/brainfood-mobile-performance-vol3.pdf

Google, Microsoft & Awwwards. (2019). *Progressive Web Apps: The future of the mobile web.* https://www.awwwards.com/PWA-ebook/en

Harris, M. (2017, oktober 10). *PWAs vs Native (aka There's A Progressive Web App For That)*. https://www.youtube.com/watch?v=vhg01Ml-8pI

Hiltunen, M. (2018). Creating multiplatform experiences with progressive web apps. https://www.theseus.fi/bitstream/handle/10024/157245/Hiltunen_Mira.pdf?sequence=1

Ivano, M. (2016). Beyond Native Apps: Web Technologies To The Rescue. *ResearchGate*. https://research.vu.nl/en/publications/beyond-native-apps-web-technologies-to-the-rescue-keynote

Lee, J., Kim, H. & Park, J. (2018). Pride and Prejudice in Progressive Web Apps : Abusing Native App-like Features in Web Applications. *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. https://cps.kaist.ac.kr/papers/18CCS-PPP.pdf

Osmani, A. (2016, oktober 5). *Progressive Web Apps with React.js: Part 3 — Offline support and network resilience*. https://medium.com/@addyosmani/progressive-web-apps-with-react-js-part-3-offline-support-and-network-resilience-c84db889162c

Richard, S. (2019, november 16). *Bridging the native app gap (Chrome Dev Summit 2019)*. https://www.youtube.com/watch?v=JKVZMqpiY7w&feature=youtu.be

Tandel, S. & Jamader, A. (2018). Impact of Progressive Web Apps on Web App Development. *Interanational journal of innovative research in Science, Engineering and Technology*, *7*(9), 9439–9444. https://doi.org/10.15680/IJIRSET.2018.0709021

Zakir, D., Kasten, J. & Halderman, A. j. (2013). Analysis of the HTTPS Certificate Ecosystem. *IMC '13 Proceedings of the 2013 conference on Internet measurement conference*. https://doi.org/10.1145/2504730.2504755