Literatuurstudie

Inhoudstafel

[Wat is een PWA 3](#_Toc34396172)

[Service workers 4](#_Toc34396173)

[Wat is een service worker 4](#_Toc34396174)

[Functionaliteiten die een serivce worker mogelijk maakt 5](#_Toc34396175)

[Service worker lifecycle 9](#_Toc34396176)

[A2HS 10](#_Toc34396177)

[Application shell 11](#_Toc34396178)

[Progressive enhancement 11](#_Toc34396179)

[App manifests 12](#_Toc34396180)

[Besturingssystemen en PWA’s 14](#_Toc34396181)

[Inleiding 14](#_Toc34396182)

[Onderzoek 15](#_Toc34396183)

[Conclusie 27](#_Toc34396184)

[Browsers 27](#_Toc34396185)

[Consistentie 27](#_Toc34396186)

[Mobiele besturingssystemen 28](#_Toc34396187)

[Waarom een PWA 30](#_Toc34396188)

[Platform onafhankelijk en bereik 30](#_Toc34396189)

[Omzet 30](#_Toc34396190)

[bundle size 31](#_Toc34396191)

[Offline gebruik 32](#_Toc34396192)

[Betrokkenheid (engagement) 32](#_Toc34396193)

[Kost 33](#_Toc34396194)

[deployment 33](#_Toc34396195)

[Beperkingen van een PWA 34](#_Toc34396196)

[Tools voor het ontwikkelen van een PWA en alternatieven 36](#_Toc34396197)

[Inleiding 36](#_Toc34396198)

[Hybrid applicaitons 37](#_Toc34396199)

[Cross platform development – native 40](#_Toc34396200)

[Tools voor het ontwikkelen van een PWA 45](#_Toc34396201)

[Lighthouse 45](#_Toc34396202)

[Workbox 46](#_Toc34396203)

[Pwabuilder 46](#_Toc34396204)

[Chrome developer tools 46](#_Toc34396205)

[Ontsluiten van een PWA? 47](#_Toc34396206)

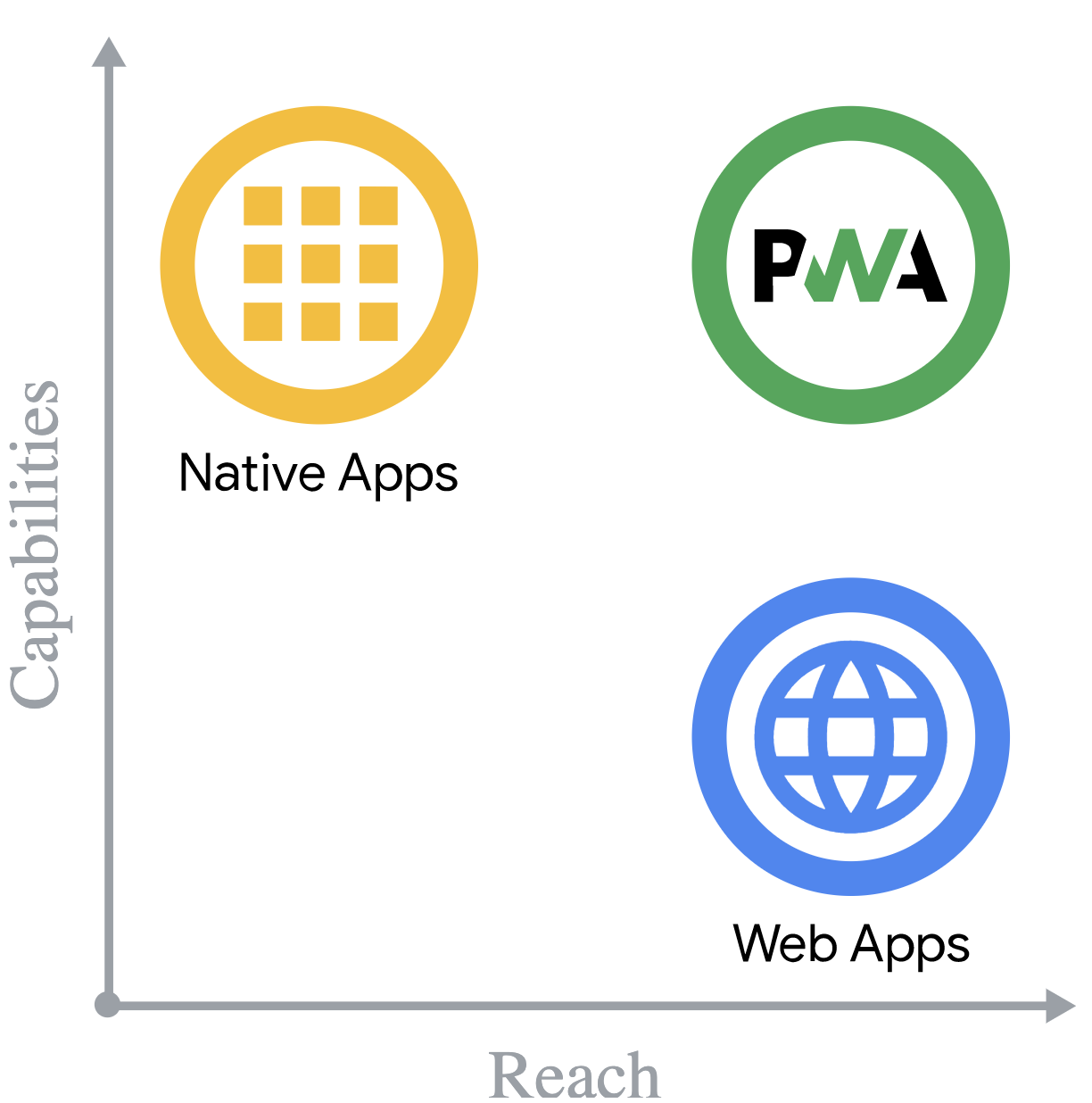
# Wat is een PWA

Het web is gegroeid tot een platform waar applicaties kunnen gepubliceerd worden zonder afhankelijk te zijn van een overkoepelend bedrijf. Voor een website is er slecht 1 codebase en telkens en de laatste versie is steeds beschikbaar voor de gebruiker.   
Dit allemaal zorgt ervoor dat een webapplicatie iedereen, overal kan bereiken en dit op elk mogelijk toestel.

Native applicaties zijn gekend om betrouwbaar te zijn en een heel goede gebruikers ervaring te kunnen beiden. Ze starten op als een alleenstaande ervaring en ze kunnen uitgebreid gebruik maken van het besturingssysteem: ze kunnen bestanden lezen en schrijven, gebruik maken van usb-connecties en bluetooth, ze hebben toegang tot de contacten en de kalender en nog veel meer. Native applicaties voelen aan alsof ze deel uitmaken van het toestel waarop ze werken.

We kunnen dus concluderen dat webapplicatie de bovenhand hebben in bereik maar dat native applicaties de bovenhand hebben als op functies aankomt.

Een Progressive web application (PWA) is een webapplicatie die gebruik maakt van moderne web API’s om een functies aan te bieden die voordien enkel beschikbaar waren voor native applicaties. De bedoeling van PWA’s is om de sterktes van webapplicaties (het bereik) en native applicaties te combineren (de functionaliteit).

­­ 

Volgens Google Developers zijn de drie pijlers van PWA’s dan ook

* Functionaliteit (capable)
* Betrouwbaarheid
* Installeerbaarheid

Deze aspecten worden in dit hoofdstuk verder uitgelegd.

<https://web.dev/what-are-pwas/> – 6 maart - 2020

## Service workers

### Wat is een service worker

De service worker is een script dat veel functionaliteiten beschikbaar maakt die voordien enkel beschikbaar waren voor native applicatie. In dit hoofdstuk wordt er bekeken naar welke functionaliteiten de service worker juist beschikbaar maakt en hoe dit gebeurt.

Een service worker is een web worker die tussen het netwerk en de applicatie wordt geplaatst. Deze service worker wordt tussen de backend en de frontend geplaatst. Dit zorgt ervoor dat de service worker inkomende en uitgaande netwerkverzoeken kan controleren en eventueel manipuleren.

<https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Offline_Service_workers> - 6 Maart - 2020

Een web worker is een script dat in de achtergrond van een applicatie werkt en die onafhankelijk is van de andere scripts. Web workers hebben dus geen impact op de prestaties van op de webapplicatie die er gebruik van maakt. Web workers hebben geen toegang tot de DOM van een webapplicatie, ze kunnen de inhoud van een website dus niet rechtsreeks manipuleren.

<https://upcommons.upc.edu/bitstream/handle/2117/99263/Web_Workers-Camera_Ready-1.pdf>

(hiltunen, 2018)

//nog iets schrijven over volgend artikel

<https://www.researchgate.net/profile/Ivano_Malavolta/publication/318326808_Assessing_the_Impact_of_Service_Workers_on_the_Energy_Efficiency_of_Progressive_Web_Apps/links/59d7404a0f7e9b42a6b069fb/Assessing-the-Impact-of-Service-Workers-on-the-Energy-Efficiency-of-Progressive-Web-Apps.pdf>

### Functionaliteiten die een serivce worker mogelijk maakt

De service worker werkt onafhankelijk van de applicatie. Dit houdt in dat een service worker wel nog kan werken terwijl de applicatie afgesloten is, hierdoor zijn volgende functies mogelijk binnen een webapplicatie:

**Offline gebruik**

De service worker kan netwerk verzoeken onderscheppen, als er een netwerk verzoek wordt verstuurd en er is geen internetverbinding zal de service worker de client antwoorden met een gepaste boodschap. Zonder service worker zou het systeem crashen.

Met service workers kunnen bestanden ook gecacht worden. Als een pagina geladen wordt kunnen alle elementen opgeslagen worden op het toestel. Als deze pagina later opnieuw bezocht wordt hoeft deze niet meer aan de server gevraagd te worden. Hierdoor wordt de applicatie sneller en minder afhankelijk van de netwerkverbinding.

De twee mechanismes die gebruikt worden om data offline beschikbaar te maken zijn ‘indexedDB’ en de ‘cache API’.

<https://developers.google.com/web/fundamentals/instant-and-offline/web-storage/offline-for-pwa> - 6 Maart - 2020

<https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API/Using_Service_Workers> - 6 Maart - 2020

De cache API wordt gebruikt om data die verkregen werd van netwerkverzoeken op te slaan. Zowel de ‘request’ als de ‘response’ van een netwerkverzoek kunnen in de cache API opgeslage worden

<https://developers.google.com/web/fundamentals/instant-and-offline/web-storage/cache-api> - 6 Maart - 2020

IndexedDB is een mechanisme dat gebruikt wordt om lokaal gestructureerde data op te slaan. Het kan vergeleken worden met traditionele relationele databasemanagementsystemen, er wordt echter geen gebruik gemaakt van kolommen maar van javascript objecten.   
Een IndexedDb maakt gebruik van indexen, dit heeft als voordeel dat het uitlezen van data heel snel gaat.

<https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API> - 6 Maart - 2020

**Push notifications**

Er zijn twee soorten notificaties lokale notificaties en push notificaties.   
Het verschil tussen beide is dat lokale applicaties geactiveerd worden vanuit de code van de applicatie, deze notificaties kunnen dus enkel geactiveerd worden als de gebruiker de applicatie aan het gebruiken is, lokale notificaties hebben dus niet zoveel use-cases. Dit is mogelijk zonder service worker.   
Push notificaties worden niet geactiveerd binnen de applicatie maar worden geactiveerd door een server.

Om push notificaties te gebruiken moet er gebruikt gemaakt worden van twee webAPI’s.

* Notifications API

Dit is een API die het uiterlijk en het gedrag van een notificatie zal bepalen. Deze API wordt zowel gebruikt voor lokale als voor push notifications.  
Om gebruik te maken van deze API moet de gebruiker expliciet toegang geven aan de applicatie.

Een voorbeeld van een notificatie kan er als volgend uitzien

function displayNotification() {  
  if (Notification.permission == 'granted') {  
    navigator.serviceWorker.getRegistration().then(function(reg) {  
      var options = {  
        body: 'Here is a notification body!',  
        icon: 'images/example.png',  
        vibrate: [100, 50, 100],  
        data: {  
          userId: “383209489398274”  
        },  
        actions: [  
          {action: 'explore', title: 'Explore this new world',  
            icon: 'images/checkmark.png'},  
          {action: 'close', title: 'Close notification',  
            icon: 'images/xmark.png'},  
        ]  
      };  
      reg.showNotification('Hello world!', options);  
    });  
  }  
}

Om een notificatie weer te geven wordt er een object verwacht waar de inhoud van de melding wordt vastgelegd. Volgende keys kunnen meegegeven worden:

|  |  |
| --- | --- |
| Body | De boodschap die in de melding staat komen. |
| Icon | Het icoontje dat in de notificatie wordt getoond |
| Vibrate | Het vibratiepatroon dat melding zal maken in miliseconden |
| Data | Data is een object dat gebruikt kan worden als de gebruiker om de notificatie klikt. Dit object zal dan ontvangen worden in de applicatie. Hier zal vaak het id van de gebruiker teruggevonden worden. |
| Actions | Er kunnen ook acties toegevoegd worden aan de melding. Elk object in deze array zal een knop worden op de melding met een andere functie. Het gedrag van de knoppen wordt bepaald in de applicatie aan de hand van de ‘ation’ |

<https://developers.google.com/web/ilt/pwa/introduction-to-push-notifications> - 6 Maart - 2020

<https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API> - 6 Maart - 2020

* Push API

De push API wordt gebruikt door de service worker. Als de server een notificatie verstuurd wordt deze opgevangen door deze push API. Deze push API zal dan gebruik maken van de notifications API om een melding op het toestel van de eindgebruiker te tonen.

<https://developer.mozilla.org/en-US/docs/Web/API/Push_API> - 6 Maart - 2020

<https://web-push-book.gauntface.com/>- 6 Maart - 2020

**Background synchronisatie**

Een PWA kan gebruik maken van de background sync API om achtergrondsynchronisatie toe te passen.

Achtergrondsynchronisatie kan toegepast worden als een trage of geen netwerkverbinding is.

Achtergrondsynchronisatie is het proces waarbij een netwerkverzoek, dat uitgevoerd werd als er geen of een te zwakke internetverbinding is, wordt opgeslagen in de service worker en wordt uitgevoerd als er wel een internetconnectie is.

Een voorbeeld hiervan is het verzenden van een bericht via een sociaal media platform. Als het bericht verzonden wordt als de gebruiker offline is, zal er geen fout getoond worden maar zal dit bericht verzenden vanaf er internet is.

Ook google chrome op android maakt hier gebruikt van. Als er een website bezocht wordt als er geen internetverbinding is krijgt de gebruiker de melding ‘chrome laat je weten wanneer de pagina klaar is’. Vanaf het toestel de pagina heeft kunnen downloaden krijgt de gebruiker een melding dat de pagina nu bekeken kan worden.

**A screenshot of a cell phone

Description automatically generatedA screen shot of a computer

Description automatically generated**

<https://developers.google.com/web/updates/2015/12/background-sync>

**Performantie**

// to doooo

### Service worker lifecycle

Om een service worker te installeren moet deze geregistreerd worden in de javascript van het de webapplicatie. Als een service worker wordt geïnstalleerd worden de opgegeven statische bestanden (foto’s, css bestanden, javascript bestanden,) gedownload. Als dit slaagt, is de service worker geactiveerd, als dit niet slaagt zal dit proces zich herhalen tot het slaagt.

Tijdens de activatiefase wordt er bekeken welke gecachte gegevens upgedate moeten worden en welke niet.

Als het oude cach geheugen is upgedate waar nodig zal de service worker overgaan naar een ‘rust’ toestand, hierbij wacht de service worker op netwerk verzoeken van bestanden die binnen zijn scope vallen.

Als er een netwerk verzoek wordt verstuurd zal de service worker deze verzoeken afhandelen. Na een bepaalde tijd zal de service worker worden afgesloten om geheugen te sparen.

In beide gevallen zal de service worker terug naar de rust staat gaan om nieuwe verzoeken af te handelen.

A picture containing text

Description automatically generated

<https://developers.google.com/web/fundamentals/primers/service-workers>

## A2HS

Als een applicatie voldoet aan bepaalde criteria kan deze geïnstalleerd worden op het toestel van de gebruiker. Deze functie is beschikbaar voor op verschillende besturingssystemen: Windows, Mac OS, Android, IOS.

Een website moet voldoen aan volgende criteria

* Nog niet geïnstalleerd zijn
* Een HTTPS connectie hebben
* Een manifest.json bestand hebben
* Een service worker registreren.

Als een website aan alle criteria voldoet zal er een beforeinstallprompt event gestart worden. Elke browser gaat hier anders met om.

Google chrome desktop:

A picture containing black

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Op Apple toestellen (iphone, Mac) heeft het beforeinstallprompt geen effect. De gerbuiker moet zelf op zoek gaan in het menu om de applicatie te installeren, maar dit is wel mogelijk.

Het beforeinstallprompt kan wel in de code opgevangen worden, De applicatie kan dan de gebruiker informeren dat de applicatie geïnstalleerd kan worden en hoe hij dit moet doen.

## Application shell

// to do

## Progressive enhancement

Dit is een principe waarbij een web applicatie gebouwd moet worden die op elke browser (ook verouderd) moet werken. Deze basis website kan dan uitgebreid worden met functies die ondersteund worden door moderne browsers. Maar de website mag niet afhankelijk zijn van deze moderne functies.

Als dit principe succesvol wordt geïmplementeerd dan is de website op alle platformen bruikbaar en is het bereik van de applicatie maximaal.

<https://aaltodoc.aalto.fi/bitstream/handle/123456789/29096/master_Vanhala_Janne_2017.pdf?sequence=1&isAllowed=y>

## App manifests

het web app manifests is een JSON bestand dat informatie bevat over de applicatie. Deze informatie is nodig voor het installeren van een PWA op een toestel.

Voorbeeld van een minimum app manifest voor de google maps PWA.



|  |  |
| --- | --- |
| short\_name | Naam van de applicatie die op het startscherm gebruikt zal worden. Deze mag maximaal 12 karakters lang zijn. |
| Name | Naam die op alle andere plekken gebruikt zal worden: vb bij de vraag als de app geïnstalleerd mag worden. Deze mag maximaal 45 karakters lang zij. |
| Icons | Een object die het icoon bepaald dat de applicatie zal gebruiken.  Dit object heeft volgende eigenschappen   * Src * Type * size |
| Start\_url | De url naar waar de PWA moet gaan als de applicatie gestart wordt vanaf het startscherm van een toestel |
| Background\_color | Hier wordt een kleur gedefinieerd, dit kleur zal gebruikt worden voor het opstart scherm |
| Display | Bepaald in wat voor webview de PWA getoond zal worden. Mogelijkheden zijn:   * fullscreen – opent de browser zonder UI-elementen (adresbalk, terug knop, ….) * standalone – opent de applicatie als een native applicatie los van de browser. Er worden geen UI elementen van de browser getoond. * minimal-ui – open de applicatie in de browser maar toont slecht beperkte ui elementen van de browser. De adresbalk is weg maar te terugknop is er nog. * Browser – opent de PWA in een normaal browser tapblad. |
| Scope | De scope bepaalt alle links die binnen de PWA vallen. |
| Theme\_color | De kleur die de adresbalk zal innemen. |

# Besturingssystemen en PWA’s

## Inleiding

In deze sectie van het onderzoek zal er gekeken welke functies het web aanbiedt en dus ook door PWA’s.

Dit onderzoek werd gevoerd met behulp van de website whatwebcando.today en caniuse.com. Als er meer informatie nodig was over de web API’s werd deze gevonden op developers.google.com of op developer.mozilla.org.

Er werd gekeken op welke platformen bepaalde functies wel en niet werkten. De volgende platformen werden onderzocht

* Microsoft edge versie 80 op Windows 10
* Mozilla firefox versie 73.0 op Windows 10
* Google chrome versie 79.0 op Windows 10
* Safari (desktop), kan niet geïnstalleerd worden op windows, alle gegevens voor safari komen van caniuse.com.
* Google chrome versie 80 op Android 10 op een OnePlus 6
* Safari (mobiel) versie 13 op IOS 13 op een Iphone SE

Het onderzoek werd opgesplitst in verschillende delen:

* Media
* Verbinding
* Toestel kenmerken
* Native gedrag
* Besturingssysteem
* Input
* User experience
* Locatie en positionering
* Scherm en output

## Onderzoek

**Media**

Video met audio – Om video en audio vast te leggen met een webapplicatie wordt er gemaakt van de Media Capture API.

Voor het starten van opnemen van een fragment zal het systeem toegang vragen tot de camera en de microfoon.

Via het window object kan er toegang verkregen worden tot het navigator object. Dit is een object dat informatie en methode bevat over de huidige applicatie.

Een van deze (read-only) properties is ‘mediaDevices’, dit geeft een object terug dat toegang heeft tot alle verbonden geconnecteerde media input toestellen. Dit kunnen camera’s, microfoons of apparatuur om het scherm te delen zijn.

Op dit object kan volgende methode uitgevoerd worden: ‘getUserMedia(constraints)’.

Dit zal een promise teruggeven dat ‘resolves’ naar een MediaStream die uitgegeven kan worden op een canvas. Als de user geen toegang geeft of er is geen camera aanwezig zal er een error gegooid worden.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ja |

Foto’s (met controlls) – Er is een API voorzien voor het nemen van foto’s binnen de browser. Dit heeft het voordeel dat het gedrag van de camera kan aangepast worden in de code. Instellingen zoals witbalans en shutterspeed kunnen aangepast worden.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Ja | Nee | Ja | Nee |

Deze beperkte beschikbaarheid geldt enkel voor het nemen van een foto zonder de website te verlaten. Het is op alle platformen mogelijk om automatisch de app te verlaten en een foto te nemen in de native camera app, deze foto wordt dan automatisch gebruikt in de website.

<https://developers.google.com/web/fundamentals/media/capturing-images>

Geluid opname – de mediarecorder API die meerdere browsers aanbieden is een manier om eenvoudig geluidsfragmenten op te nemen en te importeren in een webapplicatie. Dit wordt op een vergelijkbare manier gedaan als bij het opnemen van video. Opnieuw wordt het ‘userMedia’ object gebruikt.

Helaas is de ondersteuning voor deze web-API beperkt. In de toekomst zal deze functie waarschijnlijk ook beschikbaar worden op Apple toestellen. Dit wordt in de volgende versie van safari (safari 14) voor desktop verwacht. Voor safari voor IOS bestaat deze functie al maar is het nog een experimentele functie die de gebruiker zelf moet activeren.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Nee | Ja | Nee |

Gelukkig is er een alternatief voorzien met HTML5 tags. Dit is een methode die voor alle platformen zal werken maar niet op dezelfde manier.

<input type="file" accept="audio/\*" capture>

Er wordt gebruik gemaakt van een gewoon input veld waar de gebruiker een bestand kan uploaden. Door het accept attribuut wordt duidelijk gemaakt dat enkel audio fragmenten geüpload kunnen worden. Het capture attribuut zorgt ervoor, indien mogelijk, dat de gebruiker een audiofragment kan opnemen in de default geluidsopname app, dit fragment wordt dan automatisch geïmporteerd in de webapplicatie. Dit is enkel mogelijk op mobiele toestellen en dus niet in desktopbrowsers.

<https://developers.google.com/web/fundamentals/media/recording-audio>

Real-time communicatie/ stream – ‘real-time communication in the web’ of WebRTC is een verzameling van API’s die het verzenden en ontvangen van real-time video en audio mogelijk maakt, zonder afhankelijk te zijn van een gecentraliseerde server. Deze server is echter wel nodig om een connectie tot stand te brengen. Eens deze connectie er is, is er een peer-to-peer verbinding.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ja |

**Verbinding**

Bleutooth – Met de Web Bluetooth API kan er vanuit de browser verbinding gemaakt worden met bluetooth toestellen. Dit kan handig zijn voor applicaties die een bluetooth toestel ondersteunen zoals hartslag meters.

Er kan enkel gebruik gemaakt worden van de API als de website een SSL-certificaat heeft.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Nee | Ja | Nee |

USB – Verkopers van toestellen met usb kunnen nu gebruik maken van de WebUSB API. Bij het verbinden van een USB-toestel kan er automatisch een website geopend worden waarmee het toestel kan interageren.

Dit kan een interessant zijn voor toestellen die een eenmalige set-up nodig hebben. Met deze technologie kan vermeden worden dat er overbodige software moet geïnstalleerd worden op het toestel van de gebruiker.

Dit is echter enkel mogelijk met een beperkt aantal browsers en er moet een HTTPS-verbinding zijn.

<https://developers.google.com/web/updates/2016/03/access-usb-devices-on-the-web>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Nee | Ja | Nee |

NFC – Near field communication of NFC is een technologie om een kleine hoeveelheid informatie uit te wisselen over een kleine afstand (max 20cm). NFC wordt gebruikt om draadloze betalingen uit te voeren moet een betaalkaart of met een gsm.

Bron 20cm: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.392.7860&rep=rep1&type=pdf>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Nee | Nee | Nee | Nee |

Dit is een functie met veel mogelijkheden die helaas niet beschikbaar is voor webapplicaties.

Er bestaat echter wel een API om gebruik te kunnen maken van NFC, maar dit is een experimentele functie. Dit betekent dat de eindgebruiker dit nog niet kan gebruiken.

**Toestel kenmerken**

Netwerkinformatie – De Network information API voorziet informatie over het type netwerkverbinding de gebruiker momenteel bezit. Deze informatie bevat het connectietype (2g, 3g, 4g) en wat de maximale downloadsnelheid is van deze verbinding.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Nee | Ja | Nee |

Online status – dit is een eenvoudige eigenschap die kan opgeroepen worden op het navigator object. Deze eigenschap is een boolean die true zal zijn als de gebruiker een connectie heeft met het internet. Deze informatie kan interessant zijn bij het ontwikkelen van een PWA met offline functionaliteiten.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ja |

Vibratie motor – De Vibration API zorgt ervoor dat de vibratiemotor kan aangesproken worden vanuit de een webapplicatie. Dit kan voor een meer native gevoel zorgen. Niet alle toestellen beschikken over een vibratiemotor, deze functie mag dus enkel als ondersteuning van iets anders dienen.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Nee | Ja | Nee |

Batterij status – Aan de hand van de Battery Status API kan er informatie over de batterij van het toestel verkregen worden.

Volgende informatie kan verkregen worden

* Aan het opladen
* Batterijpercentage
* (Bij opladen) tijd tot volladen
* (Bij niet opladen) tijd tot batterij leeg

Aan de hand van deze API kunnen er ook evenement uitgevoerd worden op basis van een actie op de batterij. Er kan bijvoorbeeld een functie uitgevoerd worden als de gebruiker zijn toestel met een energiebron verbindt.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Nee | Ja | Nee |

Toestel geheugen – de Device Memory API geeft informatie over het RAM-geheugen van het toestel van de gebruiker. Dit kan interessant zijn voor het laden van een eventuele lichtere versie van een website.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Nee | Ja | Nee |

**Native gedrag**

Lokale notificaties – lokale notificaties zijn beschikbaar via de Notifciations API. Lokale notificaties zijn notificaties die geen internet of server nodig hebben. Deze kunnen gepland worden bij het laden van de website. Ze worden dus lokaal geactiveerd.

Binnen lokale notificaties kan er ook onderscheid gemaakt worden tussen twee soorten

* Persistent notifications – Dit zij notificaties waarbij het tabblad niet meer geopend moet zijn om een notificatie te ontvangen. Om dit te verkregen moet er een actieve service worker geregistreerd zijn.
* Non-persistent notifications – bij deze notificaties moet het tabblad wel nog geopend zijn, Hier is geen service worker voor nodig.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Nee |

Push messages – Push notificaties maken ook gebruik van de Notifications API. Het verschi met lokale notificaties is dat deze notificaties geactiveerd worden vanop een server. Als de admin wil dat er notificatie verzonden wordt, kan hij vanop afstand, een notificatie versturen naar alle gebruikers.

Om notificaties te ontvangen van een server moet er ook nog gebruik gemaakt worden van de Push API.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Nee | Ja | Nee |

Home screen installatie – Door het toevoegen van een web app manifest kan je de browser duidelijk maken hoe een applicatie er moet uitzien als het toegevoegd wordt aan het home screen.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Nee | Ja | Ja |

Voorgrond detectie – De Page Visibility Detection API voorziet een methode waarmee een applicatie weet als het in de voorgrond gebruikt wordt of niet.

Dit kan een handige functie zijn voor meerdere redenen. Het laden van resource-intensieve animaties is nodig als de gebruiker op dat moment niet op de pagina is.

Ook wordt het gebruikt om adverteerders te garanderen dat gebruikers een bepaalde tijd effectief naar de advertentie kijken op een website.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | ja |

Toestemmingen (permissions) – Om gebruik te maken van hardware functies van een toestel is vaak, om privacy redenen, de toestemming van de gebruiker nodig. Hiervoor is de permissions API ontwikkeld. Er kan toestemming gevraagd worden op een gelijkaardige manier voor verschillende functies.

Functies waarvoor toestemming gevraagd kan worden:

* Locatie
* Notificaties
* Push-notificaties
* Midi (musical instrument digital interface)
* klembord
* Camera
* Microfoon
* Achtergrond synchronisatie
* Licht sensor
* versnellingsmeter
* Gyroscoop
* magneet sensor
* betalingen

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | nee | Ja | nee |

**Besturingssysteem**

Offline opslag – er zijn verschillende technologieën om data offline op te slaan.

* Web storage – De meeste eenvoudige manier om data op te slaan. Er kunnen key-value paren opgeslagen worden in het localstorage of in het sessionStorage.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | ja | Ja | ja |

* indexedDB – een API voor het opslaan van grote hoeveelheden gestructureerde data op het toestel van de eindgebruiker. De data kan snel gelezen worden omdat er indexen gebruikt worden.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | ja | Ja | ja |

* cacheAPI – gespecialiseerd in het opslaan van request/ request paren. Dit is heel handig in samenwerking met een serviceworker. API-calls kunnen opgeslagen worden voor offline gebruik.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | ja | Ja | nee |

* storage API – data die is opgeslagen in een van vorige methoden kan eenvoudig verwijderd worden door de browser. Met de storage API kan data, met de gebruikers toegang, opgeslagen worden op het systeem voor een langere periode

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | ja | Ja | ja |

Bestanden toegang – toegang tot het bestandssysteem, de gebruiker kan bestanden uploaden met de file API.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | nee | Ja | nee |

Contacten – De contacten die opgeslagen staan op het systeem van de gebruiker kunnen geïmporteerd worden in een webapplicatie met de Contacts API.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Nee | nee | Beperkt \* | nee |

\* Op het moment van schrijven is dit een nieuwe en experimentele functie die enkel werkt op   
Android 10. Verdere ondersteuning is nog onbekend.

Sms – met de SMS receiver API kan er gekeken worden naar inkomende sms’en. Dit kan handig zijn bij het ontvangen van een verificatiecode.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Nee | nee | Beperkt \* | nee |

\*Op het moment van schrijven is dit een nieuwe en experimentele functie die enkel werkt op Android 10. Verdere ondersteuning is nog onbekend.

Taak planning – De task scheduler API kan er voor zorgen dat taken zoals alarmen, herinneringen en gelijkaardige taken kunnen ingepland worden in het systeem. Deze API is slechts een voorstel en heeft dus nog geen ondersteuning.

**Input**

Touch gebaren – HTML5 voegt aan de reeds bestaande input toestellen nu ook touch-controlls toe. Dit is belangrijk om een applicatie intuïtief te laten werken. Het is logisch dat Safari op desktop dit niet ondersteund aangezien safari enkel kan gedownload worden op Mac toestellen en geen enkel Mac toestel een touchscreen heeft.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Nee | Ja | Ja |

Klembord toegang – De Clipboard API geeft een ontwikkelaar de mogelijkheid om te interageren met het klemboard. Er kunnen zowel items van het klembord gelezen worden als dat er items kunnen geschreven worden naar het klembord.

Er worden ook methode voorzien voor het reageren op de actie waarbij een gebruiker zelf iets kopieert of plakt.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ja |

**User experience**

Offline gebruik – Door het gebruik van serviceworkers kan een website offline gebruikt worden. Deze website moet eerste bezocht worden als de gebruiker online. De geladen pagina’s en andere items zoals foto’s kunnen opgeslagen worden voor offline gebruik.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ja |

Achtergrond synchronisatie – Een actie kan van start gaan als de gebruiker een trage verbinding heeft of als hij offline. Achtergrond synchronisatie zal ervoor zorgen dat deze actie uitgevoerd wordt vanaf er een stabiele internetconnectie is, zelfs al is de applicatie reeds gesloten.

<https://developers.google.com/web/updates/2015/12/background-sync>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| ja | Nee | Ja | Nee | Ja | Nee |

Inter-app communicatie – de Web Share API samen met de Web share Target API zorgen ervoor dat links van websites kunnen geopend worden in native applicaties.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Nee | Nee | Ja | ja |

Betalingen – Aan de hand van de payment request API kan heel snel een betaling uitgevoerd worden. Bij apple toestellen zal dit gebeuren via apple pay. Bij andere browsers en toestellen is dit aan de hand van google pay of met een betaalkaart.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Ja | Ja | ja |

Credentials – De credential management API levert methodes voor het ophalen en opslaan van de credentials van een gebruiker. Op deze manier kan de gebruiker eenvoudiger en veiliger aanmelden op een platform.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | Nee | Ja | Nee |

**Locatie en positionering**

Geolocatie – De Geolocation API zorgt ervoor dat een applicatie toegang heeft tot de locatie van een toestel. Dat wordt gedaan op basis van de gps-sensor of op basis van het netwerk.

De API voorziet niet enkel de locatie maar ook methodes die de applicatie informeren als de locatie veranderd.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ja |

Geofencing – Geofencing is een technologie waarbij er een geografische zone wordt ingesteld. Als de gebruiker in deze zone komt wordt er automatisch een actie uitgevoerd.

Er was een voorstel om deze API uit te werken voor het web, maar op het moment van schrijven heeft geen enkele browser dit geïmplementeerd. Dit is functie die enkel beschikbaar is voor native IOS en Android-applicaties.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Nee | Nee | Nee | Nee |

Toestel oriëntatie – De Device Orientation API levert methodes voor het detecteren van de oriëntatie van het toestel. Er zijn drie eigenschappen die de oriëntatie bepalen.

* Alpha – Dit is de richting naar waar het toestel gericht is.
* Beta – Dit is het aantal graden dat het toestel voorwaarts of achterwaarts gekanteld is.
* Gamma – Dit is het aantal graden dat het toestel naar links of naar rechts gekanteld is.

De toestel oriëntatie kan, net zoals bij native applicaties, gebruikt worden als een input methode voor applicaties. Dit wordt vaak gebruikt bij games.

Deze API levert ook methodes die website in een bepaalde oriëntatie forceren.

Niet elk toestel heeft deze sensoren. Als ze aanwezig zijn worden ze ondersteund in volgende browsers. Deze API is vooral gericht naar mobiele toestellen.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Nee | Ja | Ja |

Toestel beweging – De Generic Sensor API is een verzameling van APi’s voor het gebruiken van verschillende sensoren van het toestel.

* deviceMotionEvent – levert informatie over de snelheid waarmee een toestel zijn oriëntatie en positie veranderd.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | onbekend | Ja | ja |

* accelerometer – levert information over de snelheid waarbij het toestel zich beweegt in een ruimte. De API levert zowel X, Y als Z coördinaten.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | onbekend | Ja | Nee |

* gyroscope – Levert informatie over de snelheid waarmee een toestel aan het roteren is. Om toegang te krijgen tot deze gegevens moet er eerst toestemming gevraagd worden aan de hand van de Permissions API.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | onbekend | Ja | Nee |

* Magnetometer – Meet het magnetische veld rond het toestel. Ook hier moet er eerst toestemming gevraagd worden om de sensor te mogen gebruiken.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Nee | Ja | onbekend | Nee | Nee |

* LinearAccelerationSensor – levert informatie over de snelheid waarbij een toestel beweegt, maar dit zonder de impact van de zwaartekracht.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | Ja | onbekend | Ja | Nee |

Nabijheid sensoren – geeft informatie over de afstand tussen het toestel en een object. Deze sensor wordt gebruikt om het scherm uit te zetten als een persoon aan het bellen is.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Nee | Ja | Nee | Nee | Nee | nee |

**Scherm en output**

Virtual and augmented reality – De webXR Device API zorgt voor een interface voor het connecteren van een virtual reality toestel met de browser. De sensoren van het toestel kunnen gebruikt worden om het canvas element te laden in het VR-toestel.

De oudere webVR API levert gelijkaardige methodes maar is beter ondersteund.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja (webXR) | Ja (webVR) | Ja (webVR & webXR) | Nee | Ja (webVR & webXR) | Nee |

Fullscreen – de Fullscreen API geeft een aantal methodes die ervoor zorgen dat de browser elementen verwijderd worden. Dit laat een website meer aanvoelen als een native applicatie.

Er worden twee methodes voorzien, een voor in fullscreen mode te gaan en een om deze te verlaten.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Ja | Ja | Ja | Ja | Ipad – ja Iphone – nee |

Wake lock – Veel toestellen gaan het scherm dimmen na een bepaalde tijd van inactiviteit. Dit kan onhandig zijn voor bepaalde applicaties zoals een gps. Met de Wake Lock API kan het automatisch dimmen van het scherm tegengegaan worden.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Firefox | Chrome | Safari | Android (chrome) | IOS (safari) |
| Ja | Nee | ja | Onbekend | Ja | Nee |

## Conclusie

### Browsers

We kunnen concluderen dat Google verder staat met zijn producten dan Apple. Zowel op de desktop-browser als op de mobiele browser worden meer functies ondersteund.

De nieuwste versie van Microsoft Edge ondersteunt verrassend veel functies. Dit komt omdat deze laatste versie (v80.0) gebaseerd is op de open source browser chromium die ontwikkeld is door google. Ook Google Chrome is op chromium gebaseerd.

Het is ook opvallende dat het bij Apple niet het besturingssysteem is die deze functionaliteiten tegen houdt maar de browser safari. Als Google Chrome gedownload wordt op een Apple toestel zijn deze functionaliteiten wel beschikbaar voor dit toestel.

### Consistentie

Het is opvallend hoeveel functies beschikbaar zijn voor het web. Slechts een beperkt aantal native applicaties zouden niet met web-technologieën gemaakt kunnen worden.

Het probleem is echter consistentie, het voordeel van het web is dat je één codebase hebt en dat deze applicatie op verschillende soorten toestellen werkt. Dit is voor een basis applicatie het geval, maar als er specifieke functies gebruikt moeten worden wordt het moeilijker. Verschillende systemen verwachten verschillende web-API’s. Veel van de API’s bestaan en kunnen gebruikt worden zullen ook niet op alle browsers werken.

Deze technologieën kunnen dus gebruikt worden om de functionaliteit van een applicatie te ondersteunen. Maar een applicatie zou niet mogen afhangen van een van de functionaliteiten want niet alle gebruikers zullen de toepassing kunnen gebruiken.

### Mobiele besturingssystemen

Zoals verwacht kunnen toestellen die Android gebruiken als besturingssysteem meer genieten van de functionaliteiten die het web te bieden heeft.

De grootste limitatie die IOS heeft in vergelijking met Android is het niet ondersteunen van push-notificaties. Dit is een functie die belangrijk is om gebruiker om betrokken te houden met een platform. Push notificaties zijn voor e-commerce platformen vaak een heel belangrijke tool binnen hun marketingplan  
<https://fivepints.com/PDF/5pp_Why_Are_Push_Notifications_So_Important.pdf>

Achtergrondsynchronisatie is een functie die de gebruikers ervaring kan verhogen. Deze functie die mogelijk gemaakt wordt door service workers is ook niet beschikbaar op een IOS-toestel.

Apple laat PWA’s toe om slechts 50MB aan data offline op te slaan, als deze applicatie voor een bepaalde tijd niet gebruikt wordt zal al deze data verwijderd worden om ruimte te op het toestel te besparen. Bij Android is dit 6% van de beschikbare opslagruimte en zullen de opgeslagen gegevens niet automatisch verwijderd worden.

Ook voor de functie om een web-applicatie te installeren is de ervaring op een Android toestel beter. Als een web-applicatie voldoet aan de normen om geïnstalleerd te worden zal Google Chrome op Android de gebruiker voorstellen om deze applicatie toe te voegen aan het startscherm. Deze functionaliteit is ook beschikbaar voor Safari op een Iphone maar hier moet de gebruiker zelf actie ondernemen en naar de instellingen van de website gaan om de knap ‘zet op beginscherm’ te vinden.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

De integratie van de ingebouwde slimme assistent is beperkter. Als een applicatie geïnstalleerd is op een Android toestel, kan Google assistant deze openen aan de hand van een stemcommando. Siri heeft deze functionaliteit niet.

<https://aureatelabs.com/pwa/ios-getting-into-pwa-space-faster-than-ever/>

Het is verrassend dat Apple niet meer ondersteuning biedt voor PWA’s. Tijdens de voorstellening van de eerste IPhone (2007) had Steve Jobs het over het integreren van het web met de nieuwe Iphone. Tijdens zijn keynote had hij volgende uitspraken over het web:

‘you can write apps that look and behave exactly like apps on the phone’

‘Just put them on an internet server, They’re easy to update’.

<https://medium.com/@firt/progressive-web-apps-on-ios-are-here-d00430dee3a7>

# Waarom een PWA

## Platform onafhankelijk en bereik

Een van de grootste voordelen van een PWA en het web in het algemeen is dat het platform onafhankelijk is. Een applicatie hoeft maar eenmaal ontwikkeld worden en kan dan op meerdere platformen gebruikt worden. Zowel op Apple toestellen als op Android en Windows. Het web is ook niet gebonden aan deze conventionele platformen. De dag van vandaag zijn meer en meer toestellen verbonden met het internet, toestellen zoals televisies, game consoles en e-readers.

## Omzet

Conversion rate is een meeteenheid die gebruikt wordt om de omzet van een website te pijlen. Hoe hoger de conversion rate hoe beter. De conversion rate wordt bepaald door het aantal ‘conversions’ die een website heeft ten opzichte van het aantal bezoekers. Een conversion kan voor elke website anders gedefinieerd worden. Voor een e-commerce website zal dit vaak een aankoop zijn. Voor andere websites kan dit het aanmelden van de gebruiker op de nieuwsbrief.

<https://support.google.com/google-ads/answer/2684489?hl=en>

Nikkei is een nieuws website die in 2018 zijn website ombouwde tot een PWA. Door het gebruik maken van serviceworkers konden ze de laadtijd van hun website drastisch verlagen (14 seconden sneller – van +- 20 seconden naar +-6 seconden). Dit had als gevolg dat gebruikers steeds vaker naar Nikkei gingen als nieuwsbron.

De conversion rate steeg met 58% (premium abonnement) en er was een stijging van 49% in het aantal dagelijkse gebruikers. Deze gebruikers lazen gemiddeld het dubbel aantal artikelen.

<https://developers.google.com/web/showcase/2018/nikkei>

## bundle size

Tinder besliste om zijn service ook aan te bieden als een PWA. Tinder slaagde erin om alle functionaliteiten die hun native applicaties hebben over te nemen in de PWA. Ze slaagden hierin door gebruik te maken van verschillende web API’s.

Een van de eerste voordelen is dat de native applicatie op Android op het moment van schrijven (versie 11.8.1) 130MB in beslag neemt. Als de web-versie geïnstalleerd wordt op het toestel neemt deze slechts 397KB in beslag.

In westerse landen is dit handig, maar zal het zelden bepalen als een gebruiker een applicatie effectief gebruikt of niet. In andere markten zoals Afrika is dit heel belangrijk. De toestellen die er gebruikt worden zijn vaak verouderd en hebben mindere specificaties dan dat we in het westen gewoon zijn.

De gemiddelde prijs van de top 5 meest verkochte smartphones in Afrika was 135.6 USD. Deze toestellen beschikken vaak over slechts 8 of 16GB opslagruimte.

<https://www.itnewsafrica.com/2017/10/top-5-cost-conscious-mobile-phones-for-the-african-consumer/>

Slechts 7% van Afrika beschikt over een 4g connectie. De andere gebieden moeten het vaak doen met een tragere 3g connectie.

<https://www.gsmaintelligence.com/research/?file=36b5ca079193fa82332d09063d3595b5&download>

Hier is het dus zeer handig en belangrijk om zo klein mogelijke applicaties te leveren.

|  |  |
| --- | --- |
| A screen shot of a smart phone  Description automatically generated | A screen shot of a smart phone  Description automatically generated |
| Android native applicatie versie 11.8.1 | PWA versie 118 |

<https://medium.com/@addyosmani/a-tinder-progressive-web-app-performance-case-study-78919d98ece0>

## Offline gebruik

Een webapplicatie kan nu geïmplementeerd worden met een ‘offline-first’ benadering. Offline first is gelijkaardig aan progressive enhancement, eerst wordt er een applicatie gebouwd die volledig offline beschikbaar is. Deze offlineapplicatie wordt vervolgens uitgebreid met online functionaliteiten.

Door gebruik te maken van de fetch API in een service worker kunnen API-calls onderschept worden. De service worker kan vervolgens controleren als de gebruiker online is of niet, als dit niet het geval is kan de service worker de API-call annuleren en zelf een antwoord sturen met een 200 status en een melding dat er geen internet is. Op basis hiervan kan er een gepaste boodschap getoond worden aan de gebruiker en zal de applicatie niet crashen.

<https://developers.google.com/web/ilt/pwa/introduction-to-service-worker>

Data van API-calls kan ook lokaal opgeslagen worden. Als een bepaalde API-call herhaald wordt, kan in plaats van de backend aan te spreken de lokale data gebruikt worden. Dit zal de ervaring voor gebruikers met een zwakke of inconsistente netwerk-connectie verbeteren.

<https://aaltodoc.aalto.fi/bitstream/handle/123456789/29096/master_Vanhala_Janne_2017.pdf?sequence=1&isAllowed=y>

## Betrokkenheid (engagement)

Een van de redenen om een native applicatie te maken in plaats was vaak betrokkenheid. Via het web heb je een groot bereik, maar gebruikers op een bepaald platform houden was moeilijk met de technologiën die beschikbaar waren voor het web. Progressive web apps bieden hier een oplossing voor.

Push notifications zijn meldingen die op het toestel van een gebruiker tevoorschijn komen. Deze worden verstuurd van een server en zijn dus niet afhankelijk van input van een gebruiker. Door gebruik te maken van de Push API en de Notifications API kunnen deze meldingen gebruikt worden bij een PWA.

Push notifications kunnen gebruikers herinneren om een bepaalde applicatie terug te gebruiken.

Push notificaitons kunnen ook gebruikt worden om de conversion rate te verhogen. Bijvoorbeeld bij een e-commerce platform kan er een melding gestuurd worden als de klant items in zijn mandje heeft geplaatst maar deze nog niet heeft besteld.

<https://web-push-book.gauntface.com/> - pdf version - <https://web-push-book.gauntface.com/downloads/web-push-book.pdf>

(hiltunen, 2018 – creating multiplatform experiences with progressive web apps)

Als een webapplicatie voldoet aan de criteria (**zie wat is een PWA**) van een PWA kan deze toegevoegd worden aan het startscherm van een toestel. Hierdoor wordt het voor de gebruiker gemakkelijker om een PWA herhaaldelijk te gebruiken. Als een applicatie wordt geopend vanaf het startscherm kan deze het volledige scherm gebruiken en zullen adresbalk en andere tools van de browser weggelaten worden. Dit zorgt ervoor de aan PWA meer zal aanvoelen als een native applicatie dan een website.

## Kost

Het ontwikkelen van native applicaties is niet goedkoop. Services zoals Spotify moeten native applicaties ontwikkelen voor verschillende platformen (mac OS, windows, IOS, Android, Linux, ChromeOS).

Het creëren van een service voor verschillende platformen is vaak te duur voor een kleiner of startend bedrijf. Progressive web apps kan hier een oplossing bieden, een PWA kan geïnstalleerd worden op al de bovenstaande platformen. In plaats van eenzelfde applicatie te bouwen voor verschillende platformen kan er met PWA’s dus 1 applicatie geschreven worden die op verschillende platformen kan gebruikt worden.

De reduceert de ontwikkelkosten drastisch.

Spotify implementeerde een zijn service ook als een PWA, al de functies die op de native applicaties beschikbaar zijn, zijn ook beschikbaar via deze webapplicatie. Enkel de offline functie is (nog) niet geïmplementeerd.

**HERSCHRIJVEN!!**

<https://www.simicart.com/blog/spotify-pwa/>

<https://open.spotify.com/>

## deployment

Het uitgeven van een PWA is ook een stuk gemakkelijker en goedkoper dan het verdelen van native applicaties door middel van de verschillende app-stores.

Elke app-store (windows, apple, android, chrome, …) heeft zijn eigen specifieke eisen en kosten om een applicatie te kunnen uploaden. Bij een PWA hoeft er slechts één applicatie online gezet te worden. Deze moet niet gevalideerd worden door andere partijen.

# Beperkingen van een PWA

Zoals aangetoond in de sectie ‘besturingssystemen en PWA’s’ kan het web gebruik maken van verschillenden hardware functies van een toestel. Echter onderzoek toont dat voor sommige toepassingen een PWA niet de oplossing is.   
Het onderzoek van Rebecca Fransson toont aan dat de video’s die genomen worden met de mediaCapture API van een veel lagere kwaliteit zijn. Ook duurde het proces van het nemen van een video statistisch significant langer, deze methode is wel ondersteund door alle populaire browsers.

De resultaten waarbij video’s genomen worden met de nieuwere ImageCaptureAPI waren een stuk beter en benaderden de kwaliteit van een native applicatie. Het probleem bij deze techniek is dat deze API enkel ondersteund wordt door google chrome (web & mobile)

In de thesis werd ook onderzoek gedaan naar de Geolocation API. Verrassend genoeg bleek deze beter te presteren dan de native applicaties die gebruik maakten van locatievoorzieningen.

<http://www.diva-portal.org/smash/get/diva2:1105475/FULLTEXT01.pdf>

Met native applicaties kan de ontwikkelaar kiezen voor welke platformen hij de applicatie wil ondersteunen. Het web, en dus ook PWA’s kunnen van op verschillende toestellen bezocht worden. Dit is een van de voordelen van het web, echter dit zorgt ervoor dat de ontwikkelaar rekening moet houden met de verschillende mogelijkheden van de verschillende toestellen. Er kan een applicatie geschreven worden die afhankelijk is van een cameratoepassing, toestellen die geen camera hebben (oudere telefoons, smart-tv, desktops, …) kunnen perfect op deze site belanden. Helaas zullen zij niet kunnen genieten van de functionaliteit van de applicatie.

Integratie met het besturingssysteem is ook niet mogelijk. Een PWA kan bijvoorbeeld geen widgets plaatsen op het startscherm van een Android toestel. De instellingen van een toestel kunnen ook niet gemanipuleerd worden van een website. Native applicaties kunnen bepaalde acties uitvoeren door de ingebouwde smart assistant (google assistant voor android, Siri voor IOS), Deze assistenten kunnen nog niet interageren met geïnstalleerde PWA’s.

Toegang tot de hardware van toestellen is nog steeds beperkt. Er bestaan reeds heel wat web-API’s voor het aanspreken van deze sensoren. De ondersteuning van deze technologieën is nog niet goed ondersteund door de veel voorkomende browsers (zie sectie ‘besturingssystemen en PWA’s).

De toegang tot persoonlijke informatie is ook beperkt. Er is voor webapplicaties geen toegang tot belgeschiedenis, berichten, kalender, … Dit is data waar native applicaties wel gebruik kunnen van maken.

(<https://is.muni.cz/th/433364/fi_b/bachelor-thesis-pavel-brousek-pwa.pdf>)

Progressive webapplicaties zijn gemakkelijk te vinden via het web, maar bepaalde potentiele gebruikers zullen een applicatie in de een app-store verwachten. Deze zal hier niet te vinden zijn.

## Tools voor het ontwikkelen van een PWA en alternatieven

### Inleiding

In deze sectie zal er onderzoek gedaan worden naar technologieën waarbij er een applicatie ontwikkeld kan worden die op meerdere platformen waarbij er ook maar 1 codebase is Er zal gekeken worden wat de voor en nadelen zijn van de methodes. Er zullen methodes en technologieën bekeken worden om PWA’s te ontwikkelen, er zal ook op zoek gegaan worden naar eventuele alternatieven.

Het doel van cross platform ontwikkeling is dat het gemakkelijker en sneller moet zijn om een applicatie te creëren en te onderhouden voor meerdere platformen.

Het ontwikkelen van een applicatie aan de hand van een cross platform development benadering zal minder tijd in beslag nemen, aangezien er minder code moet geschreven worden, dan het ontwikkelen van native toepassingen voor meerdere platformen. Dit zal voor de opdrachtgever een goedkoper proces zijn.

Een ander voordeel van deze technologie is dat het gemakkelijker is om consistent te zijn met een applicatie over meerdere platformen. Als een applicatie op de traditionele manier wordt ontwikkeld, is er vaak een team voor elk platform. Deze teams maken dezelfde applicatie maar voor een ander platform. Dit heeft als gevolg dat beide applicatie niet 100% gelijk zullen zijn. Dit kan de gebruiker verwarren.

### Hybrid applicaitons

<http://iisit.org/Vol14/IISITv14p049-065Huynh3472.pdf>

Hybride mobiele applicaties zijn apps die waarbij de userinterface gerenderd wordt in een webbrowser.  
Dit type applicatie kan gebouwd worden met de technologieën die beschikbaar zijn voor het web.

Bij het creëren van een hybride applicatie wordt er een minimale browser aangemaakt in het package van de applicatie. De applicatie wordt vervolgens in deze minimale browser gebouwd. Deze combinatie van bestanden kan vervolgens upgeload worden naar platformen zoals de google play store of de app store van Apple.

#### Pro’s

Het grote voordeel van hybride applicaties is dat er een lage leercurve is. De technologieën die gebruikt worden bij web-ontwikkeling kunnen overgenomen worden om app-ontwikkeling te doen.

Dit betekent ook dat de vele libraries en packages die beschikbaar zijn voor het web ook gebruikt kunnen worden voor app-ontwikkeling.

#### Con’s

De userinterface die gemaakt wordt aan de hand van een hybride applicatie is op elk toestel dezelfde. Dit zorgt ervoor dat de app niet intuïtief kan aanvoelen. Een native Androidapplicatie heeft bijvoorbeeld een ander navigatiesysteem dan een iOS applicatie. Bij het maken van een hybride oplossing zal deze navigatie op beide platformen dezelfde zijn.

De prestaties van een website die in een webview werkt zijn minder goed dan bij een native applicatie.

**BRON TOEVOEGEN**

#### Apache Cordova

Apache Cordova is een technologie die een website ontsluit in een webview. Deze webview kan gezien worden als een basisversie van een gewone mobiele browser zonder interface element zoals een url-veld of een status bar. Er bestaan ook veel plugins die het gemakkelijk maken om de functies van het besturingssysteem te gebruiken. Voorbeelden hiervan zijn het gebruik van de camera of de flashlight.

#### Ionic

Omdat applicaties bouwen aan de hand van cordova niet eenvoudig is werd de technologie minder en minder gebruikt. Ionic is een framework dat gebruik maakt van de webview en de plugins die cordova biedt. Maar deze kunnen nu gebruikt worden vanuit andere front-end frameworks zoals Angular, React en Vue.

Dit framework heeft ook een ruime bibliotheek aan plug-ins die ervoor zorgen dat een applicatie gebruik kan maken van de functies die een besturingssysteem heeft.

Voorbeelden hiervan zijn

* Offline storage
* Push-notificaties
* Vingerafdruk authenticatie
* 250 meer (<https://ionicframework.com/docs/native>)

Doordat een applicatie gebouwd wordt met bestaande front-end frameworks is ionic ook heel geschikt voor het ontwikkelen van PWA’s. Verschillende plugins die beschikbaar zijn voor native development zijn ook beschikbaar voor het web.

<https://ionicframework.com/blog/native-or-pwa-how-to-choose-the-right-approach-for-mobile-app-development/>

De ontwikkelervaring van een ionic applicatie is ook aangenaam omdat ionic ‘live reloading’ ondersteund. Dit wil zeggen dat de applicatie niet opnieuw gebuild moet worden om de aanpassingen te zien.

<https://ionicframework.com/pwa/toolkit>

Het ionic framework zal er ook voor zorgen dat de applicatie meer native aanvoelt door de UI-elementen die het aanbiedt. Als een applicatie gebuild wordt zal ionic ervoor zorgen dat de juiste ui-componenten voor elk toestel gebruikt worden. Zo zal een android applicatie het material design volgen en een IOS-applicatie het human design.

#### Electron

Electron is een framework die gebruikt kan worden om desktopapplicaties te bouwen gebruik makende van web technologieën. De code kan gecompileerd worden naar een native programma voor linux, mac of windows.

Een applicatie dat geschreven is met electron heeft, in tegenstelling tot een website, wel de rechten om bestanden die op het toestel aan te passen of aan te maken.

Voorbeelden van software die ontwikkeld zijn met Electron zijn

* Slack
* Visual studio code
* discord

### Cross platform development – native

<https://scholarspace.manoa.hawaii.edu/bitstream/10125/50605/paper0718.pdf>

<http://www.diva-portal.org/smash/get/diva2:1349917/FULLTEXT01.pdf>

<https://blog.codemagic.io/flutter-vs-ios-android-reactnative-xamarin/>

Het grote verschil met hybrid applications is dat bij cross platform development de code gecompileerd wordt naar native componenten.

#### Pro’s

De meeste cross platform development frameworks hebben een functie ‘hot reload’. Dit wil zeggen dat de app tijdens het ontwikkelen op een emulator kan uitgevoerd worden. Elke keer een verandering aangebracht wordt, zal deze app direct updaten. Dit vereenvoudigt het ontwikkelproces.

Bij cross platform development wordt de code omgevormd tot native componenten. Dit zorgt dat een applicatie een meer ‘native feeling’ zal hebben dan bij hybrid applications.

De performantie van cross platform

BRON

#### Con’s

De performantie is dan wel beter dan bij een hybrid applicatie, maar voor applicaties die veeleisend zijn op vlak van cpu en geheugen zijn cross platform development frameworks niet de beste oplossing. Hier wordt beter een native oplossing gebruikt waar bij er toch nog twee codebases zijn.

BRON

#### Xamarin

<https://dotnet.microsoft.com/apps/xamarin>

<https://brainhub.eu/blog/react-native-vs-xamarin/>

Xamarin is een platform die gebruikt kan worden om applicaties voor Android, IOS, Windows en Mac Os te maken. De taal en technologie die bij dit platform gebruikt wordt is C# en .NET. Xamarin is ontwikkeld en wordt onderhouden door Microsoft.

De IDE Visual studio voorziet veel tools die het ontwikkelen van xamarin applicaties makkelijker maakt.

Xamarin is het framework die de cross-platform trend heeft gestart. In begin van 2013 werd het framwork uitgebracht. Doordat het al enkele jaren bestaat zijn er al veel vragen gepost op fora, waardoor de kans groot dat iemand een probleem dat jij als ontwikkelaar hebt als heeft gehad voor jou.

Dit is ook terug te zien in het aantal zoekopdrachten op stackoverflow. Bij de categorie ‘frameworks’ staat react native op de 10e plaats. Enkel react native en cordova staan hoger op deze lijst.

#### React native

<https://facebook.github.io/react-native/>

<https://desmart.com/blog/how-to-build-cross-platform-apps-with-react-native-bridges>

<https://insights.stackoverflow.com/survey/2019#technology>

React native is een framework gebaseerd op react. Beide frameworks zijn ontwikkeld en worden onderhouden door Facebook en worden geschreven in javascript.

Het voordeel van react is dat er een ‘learn once, write anywhere’ benadering is. React en React native zijn gelijkaardig, dit zorgt er dus voor dat een ontwikkelaar maar 1 technologie moet leren en dan voor zowel web als mobiel kan ontwikkelen.

React was het eerste cross-platform framework dat succes kende. Dit heeft als voordeel dat het al langer bestaat dan zijn concurrenten. Hierdoor is er al een grote online comunity waar je terecht kan bij eventuele problemen. Dit is ook terug te zien in het aantal zoekopdrachten op stackoverflow. Bij de categorie ‘frameworks’ staat react native op de 6e plaats. Dit is het hoogst genoteerde framework dat mobiele applicaties produceert.

React native maakt gebruik een concept genaamd ‘bridges’. Dit is de vertaling die gemaakt wordt van javascript naar native. React native voorziet deze ‘bridges’ voor de veel voorkomende items zoals een button, een tekst field, een kalender, Als er meer specifieke en minder courante functies gebruikt worden moeten deze bridges zelfgeschreven worden. Deze bridges moeten geschreven worden in de taal van het platform. Voor android is dit java of kotlin voor IOS is dit Objective c of swift.

**Expo**

Expo is een verzameling van tools die aan react native toegevoegd worden. Het levert bijvoorbeeld een command line interface en applicatie waardoor een applicatie direct kan getest worden op een fysiek mobiel toestel.

Expo heeft verschillende API’s om toegang te krijgen tot de verschillende functies van een toestel. De meeste applicaties kunnen gebouwd worden door gebruik te maken van deze API’s.

Het gebruik van expo kan echter ook limiterend zijn voor een applicatie. Als ontwikkelaar ben je volledig afhankelijk van deze API’s. Er kan enkel JavaScript geschreven worden en dus geen native code zoals dit bij een traditioneel react native project wel mogelijk is.

Het voordeel van Expo is dat er snel een applicatie kan ontwikkeld worden door enkel gebruik te maken van javascript.

**React native web**

Een react native project dat gebruik maakt van Expo heeft nu ook web ondersteuning. Dit houdt in dat een applicatie geschreven wordt in javascript en kan geëxporteerd worden naar een IOS app, android App en een PWA.

Deze functionaliteit is nog in de Beta fase en er wordt aangeraden dit nog niet te gebruiken in productie.

Browser support

* Chrome
* Firefox
* Edge
* Safari
* IE

Features die beschikbaar zijn voor expo native

<https://github.com/expo/web-examples/blob/master/docs/FEATURES.md>

#### Flutter

<https://flutter.dev/>

<https://flutter.dev/web>

Flutter is een relatief nieuw framework voor het ontwikkelen van mobiele applicaties. De programmeertaal die gebruikt wordt bij flutter is dart. Zowel flutter als dart zijn gecreëerd en worden onderhouden door Google.

Flutter heeft een functie ‘hot reload’ hierbij kunnen de aanpassingen direct bekeken worden in een emulator of op een toestel. Hot reload is beter dan ‘live reload’ van ionic omdat de state hier wel behouden wordt

Net zoals react native is het nu ook mogelijk om een PWA te maken aan de hand van flutter. Ook bij flutter is het nog niet aangeraden om dit al effectief te gebruiken omdat PWA-ondersteuning nog een de beta fase is.

Een nadeel is dat flutter een heel nieuw framework is en dat er online minder hulp te vinden zal zijn voor de ontwikkelaar dan bij react native. Dit is ook terug te zien in het aantal zoekopdrachten op stackoverflow. Bij de categorie ‘frameworks’ flutter slechts op de 12e plaats. Dit is slechts het 4e framework voor app development.

# Tools voor het ontwikkelen van een PWA

## Lighthouse

Lighthouse is een tool die een audit zal uitvoeren op een website en een rapport zal uitgeven. Dit rapport kan een ontwikkelaar veel interessante informatie geven om de ervaring van deze website te verbeteren. De audit geeft inzichten op vlak van

* Prestaties
* Toegankelijkheid
* Best practices
* SEO
* PWA’s

Voor elk van deze onderdelen, op PWA’s na, zal er een score op 100 gegenereerd worden. De audit zal ook voorstellen doen om deze score’s te verhogen en de evaring voor de eindgebruiker dus te verbeteren.

Voor het onderdeel PWA zal er een checklist gegenereerd worden met items die moeten voldaan worden zodat de applicatie geïnstalleerd kan worden op een toestel.

A screenshot of a cell phone

Description automatically generated

## Workbox

Workbox is een verzameling van libraries die helt bij het ontwikkelen van service worker geralateerde functionaliteiten.

Workbox speelt wil het gemakkelijker maken voor de ontwikkelaars om middelen te cachen en op deze manier een snellere en minder netwerk-afhankelijke applicatie te maken.

Het debuggen van een PWA kan ook moeilijk worden omdat code lokaal opgeslagen kan worden en je als ontwikkelaar niet altijd weet welke versie van de code voor een bepaald gedrag zorgt. Ook om deze problemen te debuggen voorziet workbox tools.

## Pwabuilder

Pwabuilder.com is een website die ervoor zorgt dat een PWA toch in de app-stores kan terechtkomen. Als ontwikkelaar moet je juist een link van een PWA opgeven en de tool maakt van deze PWA vier paketten die kunnen upgeload worden naar hun bijhorend app-store.

Volgende platformen zijn ondersteund:

* Android
* Samsung (eigen samsung store)
* Windows
* IOS

## Chrome developer tools

De chrome developer tools bieden een grote hulp bij het ontwikkelen van web-applicaties. Maar het onderscheid zich met andere browsers als op PWA’s aankomt.

Chrome zorgt ervoor dat er eenvoudig gecachte items tijdelijk verwijderd kunnen worden. Hierdoor kan de ontwikkelaar testen als een service worker de juiste bestanden offline beschikbaar maakt

Google chrome biedt tools om eenvoudig een website offline te testen of om bepaalde service workers uit te schakelen.

Ook de volledigheid van het app-manifest bestand kan in de chrome developer tools bekeken worden.

# Ontsluiten van een PWA?

In bepaalde situaties zal een PWA niet alle functionaliteiten kunnen ondersteunen om alle requirements van een project te voldoen. Een intuïtieve beslissing zou hier zijn om over te schakelen naar native applicaties en PWA’s links te laten liggen. Dit zou wel als gevolg hebben dat er veel potentiële gebruikers verloren gaan omdat de applicatie dan enkel te vinden is in de app stores en niet via de browser.

In deze sectie van het onderzoek zal er gekeken worden als er een hybride oplossing mogelijk is. Hierbij worden zoveel mogelijk functionaliteiten van de applicatie geïmplementeerd als PWA.

Vervolgens kan er een native applicatie gemaakt worden die de ontbrekende functionaliteiten implementeert. Deze native applicatie kan de functionaliteiten die al ontwikkeld zijn in de PWA implementeren als een webview. Op deze manier moet er geen redundante code geschreven worden

Data toont aan dat de meeste tijd van een gebruiker op een smartphone in native applicaties is (87%) maar van deze 87% wordt 80% van de tijd in slechts 3 verschillende applicaties gespendeerd. Terwijl er veel tijd besteed wordt in native applicaties, is het toch heel moeilijk om tijd te krijgen van de gebruiker om jouw applicatie te gebruiken.

Het is ook niet gemakkelijk om gebruikers te introduceren tot jouw applicatie, de gemiddelde gebruiker download 0 nieuwe applicaties per maand.

Echter op het web, waar slechts 17% van de tijd besteed wordt, worden er maandelijks ongeveer 100 websites bezocht. Hier is er voor bedrijven dus een kans om een gebruiker jouw platform te laten ontdekken.

<https://www.youtube.com/watch?time_continue=15&v=1QILz1lAzWY>

Hier wordt het grote voordeel van deze benadering duidelijk. Gebruikers kunnen de applicatie ontdekken en uittesten via de PWA. Als ze da applicatie interessant vinden kan een volledigere versie gedownload worden via de app-store

Deze benadering zal een aantal uitdagingen met zicht meebrengen maar biedt ook een aantal voordelen.

De grootste uitdaging zal data uitwisseling van data tussen de native app en de web app zijn. De state gesynchroniseerd houden tussen beide projecten zal niet eenvoudig zijn. Er zullen meer data requests van de backend nodig zijn om deze state gesynchroniseerd te houden.   
Het is ook belangrijk dat een applicatie een ‘native feeling’ heeft. Er zullen code optimalisaties nodig zijn om de webviews snel en responsive te laten werken.

De technologie die hiervoor gebruik zal worden is expo. Expo is een framework dat gebruikt maakt van react native, het voordeel van expo is dat de ontwikkelaar enkel javascript schrijft om een native applicatie te ontwikkelen. Bij de ‘traditionele’ react-native moet er vaak platform specifieke code geschreven worden in Java of Objective-c.

Verder in deze thesis zal er een voorbeeld uitgewerkt worden die deze methode demonstreert.

Apple maakt echter wel duidelijk dat het niet de bedoeling is om een website te maken en deze zonder toegevoegde native functionaliteiten in de app-store te plaatsen. De regelse van de appstore stellen het volgende:

*“Your app should include features, content, and UI that elevate it beyond a repackaged website. If your app is not particularly useful, unique, or “app-like,” it doesn’t belong on the App Store.”*

<https://developer.apple.com/app-store/review/guidelines/>

De google play store heeft hier geen specifieke regels over. <https://play.google.com/about/developer-content-policy-print/>