Een PWA is een traditionele webapplicatie met een paar extra verbeteringen die ervoor zorgt dat er een betere user experience geleverd kan worden. Een belangrijk voorbeeld hiervan is offline gebruik.

(Harris, 2017)

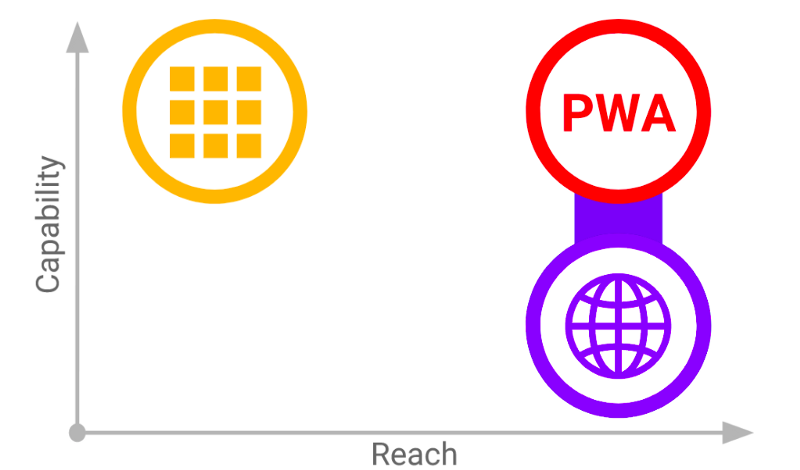
Deze websites kunnen nog steeds gevonden en gebruikt worden in alle moderne browsers, maar een PWA kan, in tegenstelling tot een traditionele website, geïnstalleerd worden op het toestel van de gebruiker. Dit zowel op Android, IOS en desktop.

Een website moet aan 3 voorwaarden voldoen om een PWA te zijn.

1. De website moet een https-connectie hebben.
2. De website heeft een app-manifest bestand nodig. Dit is een bestand die de look een feel van een applicatie bepaald
3. De website moet een serviceworker registreren. Dit is een javascript bestand die voor de extra functionaliteiten zal zorgen.

(Ivano, 2016)

Een PWA kan gezien worden als een combinatie van een website en een native app. Het heeft het bereik van het web maar heeft wel de functionaliteiten van een native applicatie.



(<https://www.freecodecamp.org/news/progressive-web-apps-bridging-the-gap-between-web-and-mobile-apps-a08c76e3e768/>)

**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> – page 12

Refereeert naar <https://www.amazon.com/Single-Page-Applications-end-end/dp/1617290750>

**Service workers**

Een service worker is een type web worker, een web worker is een bestand dat in de achtergrond van een applicatie werkt en die onafhankelijk is van de andere scripts. Web workers hebben geen impact op de prestaties van op de webapplicatie die er gebruik van maakt.

Een service worker is een web worker die tussen het netwerk en de applicatie wordt geplaatst. Deze worker zal zich gedragen als een client-side proxy die de netwerk requests die de applicatie maakt kan controleren.

De service worker werkt onafhankelijk van de applicaite. Dit wil zeggen dat een service worker wel nog kan werken terwijl de applicatie afgesloten is, hierdoor zijn 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 internet verbinding 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.

* Push notifications

Push notificaties zijn notificaties die op het toestel van de gebruiker getoond worden. Deze notificaties zijn afkomstig van de server en dus niet van de client. Notificaties van een website worden door de server naar de service worker verzonden, deze service worker zal er dan voor zorgen dat de notificatie op het toestel zal verschijnen.

* Background sync

Achtergrond synchronizatie houdt in dan een applicatie een API-verzoek kan vragen op het moment dat er geen netwerkconnectie is. Dit verzoek zal verder afgehandeld worden eens er terug verbinding is, de applicatie mag ondertussen afgesloten zijn.

(hiltunen, 2018)

<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>

service worker lifecycle

Om een service worker te installeren moet deze geregistreerd worden in de javascript van het de web-applicatie. Als een service worker wordt geïnstalleerd worden de opgegeven statische bestanden (fotos, css bestanden, javascript bestanden,) gedownload. Als dit geslaaged, is de service worker geactiveerd, als dit niet slaagd zal dit proces zich herhalen tot het slaagt.

Tijdens de activatie fase 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>

**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 |
| Name | Naam die op alle andere plekken gebruikt zal worden: vb bij de vraag als de app geïnstalleerd mag worden |
| 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. |