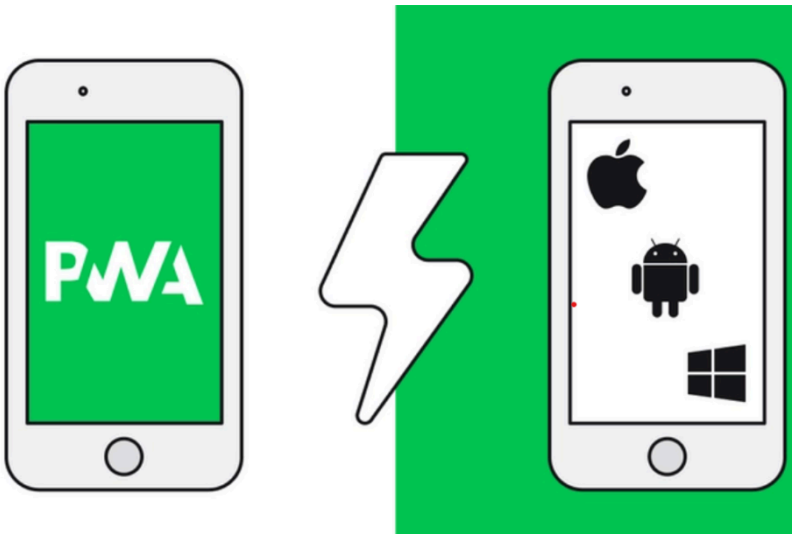
PXL DIGITAL

Research Project Toegepaste Informatica



Het optimaliseren van een paardenfokkerij webapplicatie met de PWA-technologie

Groep AON 6

Anton Van Kimmenade

Jamie Luyten

Kian Van Dyck

Sander Van Overloop

Zidane Boutayniout

Academiejaar 2023-2024

Research Project Toegepaste Informatica

Projectomschrijving

Dit onderzoek, uitgevoerd binnen het vak Research Project aan de Hogeschool PXL, richt zich op het verbeteren van de gebruiksvriendelijkheid van de webapplicatie die is ontwikkeld voor paardenfokkerij Tinkerstal ’T Boesehof.

Om de gebruiksvriendelijkheid van de applicatie te verbeteren, ligt de focus van dit onderzoek op de technologie genaamd *Progressive Web Apps* (PWA-technologie). De resultaten zijn alleen van toepassing op de applicatie van Tinkerstal 'T Boesehof, maar bepaalde analyses kunnen wel hergebruikt worden voor soortgelijke projecten.

Dit onderzoek heeft als doel de voordelen van PWA’s te onderzoeken en toe te passen om de applicatie van de klant te optimaliseren, rekening houdend met wet- en regelgeving zoals de *Digital Markets Act* (DMA). De DMA kan PWA's beïnvloeden door regels op te leggen aan digitale marktplaatsen. Het resultaat zal conclusies bieden over de manier waarop de implementatie van PWA-technologie gebruiksvriendelijkheid in mobiele applicaties vergroot.

Door middel van een casestudy wordt gericht onderzoek gevoerd naar de voor de klant relevante aspecten van de PWA-technologie.

Deze voordelen worden onderzocht aan de hand van een literatuurstudie en vervolgens toegepast in de vorm van een *Proof Of Concept*. De technologie zal identiek zijn aan de applicatie ontwikkeld voor Tinkerstal 'T Boesehof.

Het onderzoek bestrijkt verschillende onderwerpen, waaronder gebruiksvriendelijkheid, installatiegemak, offline beschikbaarheid, gegevensbeheer en de (toekomstige) implicaties met betrekking tot de nieuwe DMA-wetten.

Research Project Toegepaste Informatica

Inhoudsopgave

Research Project Toegepaste Informatica

Lijst van gebruikte figuren

Research Project Toegepaste Informatica

Lijst van gebruikte afkortingen

API	Application Programming Interface
CSR	Client-Side Rendering
DMA	Digital Markets Act
DNS	Domain Name System
PoC	Proof Of Concept
PWA	Progressive Web Apps
PM2	Process Manager 2
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
SVG	Scalable Vector Graphics
SSL	Secure Socket Layer
SSR	Server Side Rendering
IDB	Indexed Database

Research Project Toegepaste Informatica

1. Onderzoeksvraag en hypothese

1.1 Onderzoeksvraag

Hoe kan de implementatie van de PWA-technologie de gebruiksvriendelijkheid van de mobiele webapplicatie van paardenfokkerij Tinkerstal ‘T Boesehof verbeteren?

1.2 Deelvragen

1. Hoe onderscheiden de kernkenmerken van PWA’s zich van traditionele mobiele applicaties, en op welke manier dragen deze specifieke kenmerken bij aan een verbeterde gebruikerservaring voor de applicatie van Tinkerstal ‘T Boesehof?

2. Op welke manier zorgt de PWA-technologie voor offline functionaliteit en is het mogelijk om dit toe te passen voor de applicatie van Tinkerstal ‘T Boesehof?

3. Kunnen PWA’s zonder internetconnectie functioneren en in hoeverre is dit van belang voor de applicatie van Tinkerstal ‘T Boesehof?

4. Hoe draagt het gebruik van lokale opslag en *caching* binnen PWA’s specifiek bij aan een efficiënt beheer van gegevens voor de applicatie van Tinkerstal ‘T Boesehof?

5. Wat zijn de mogelijke implicaties van de Digital Markets Act voor de distributie, toegankelijkheid en zichtbaarheid van PWA’s in vergelijking met *native apps*, met aandacht voor het ontbreken van appstore-installaties?

1.3 Hypothese

De implementatie van Progressive Web Apps (PWA) zal naar verwachting de gebruikerservaring verbeteren door snellere laadtijden, betere offline functionaliteit en het gebruik van lokale opslag. Er zijn ook enkele nadelen die in overweging moeten genomen worden bij het kiezen voor deze technologie. We vermoeden dat deze nadelen niet van belang zijn voor de applicatie van Tinkerstal ‘T Boesehof.

2. Onderzoeksmethode

De onderzoeksmethode omvat een literatuurstudie om de belangrijkste kenmerken van PWA's, hun voordelen en de implicaties van de nieuwe DMA-wetten in kaart te brengen. Er zijn verschillende studies geanalyseerd die zich richten op gebruikerservaringen, distributie en zichtbaarheid van PWA's, met specifieke aandacht voor de DMA-gerelateerde aspecten.

Vervolgens is een casestudy uitgevoerd om te onderzoeken hoe gebruikers de ervaring van een mobiele applicatie beoordelen die PWA-technologie gebruikt.

Na afronding van de casestudy is een Proof Of Concept (PoC) ontwikkeld, waarbij de focus lag op aspecten zoals gebruiksvriendelijkheid, laadtijden, offline functionaliteit en lokale opslag. Op deze manier is de effectiviteit van PWA-technologie vastgesteld.

Op basis van dit onderzoek zijn conclusies getrokken over hoe de implementatie van PWA-technologie de gebruiksvriendelijkheid van de webapplicatie van paardenfokkerij Tinkerstal 'T Boesehof kan verbeteren.

3. Literatuurstudie

3.1 Wat is een Progressive Web App?

Een *Progressive Web App* (PWA) is een webapplicatie die is ontwikkeld met behulp van moderne webtechnologieën om een ervaring te bieden die vergelijkbaar is met die van *native apps*, ongeacht het gebruikte apparaat of platform, zoals mobiel, desktop en web. PWA's maken gebruik van *service workers* om functies zoals offline ondersteuning mogelijk te maken en kunnen worden vastgemaakt aan het startscherm van een gebruiker. Ze zijn snel, kosteneffectief en gemakkelijk vindbaar. Bovendien bieden PWA's een naadloze gebruikerservaring over verschillende apparaten heen en kunnen ze snel worden geïmplementeerd zonder de goedkeuring van appstores [1] [2] [4].

Een groeiend aantal bedrijven toont interesse in de PWA-technologie, een trend die ook opvalt binnen de technologiesector. Bekende namen zoals Starbucks, AliExpress, Uber en Twitter hebben de overstap naar PWA's gemaakt, waardoor de toekomst van mobiele applicaties steeds duidelijker wordt. Hoewel sommige bedrijven de voordelen van PWA's omarmen, blijven anderen, zoals Amazon, trouw aan hun *native apps* [1].

Vanuit zakelijk perspectief speelt kostenbesparing vaak een cruciale rol bij de keuze tussen een PWA en een *native app*. PWA's zijn goedkoper in ontwikkelings- en onderhoudskosten vanwege het gebruik van één codebase die op verschillende besturingssystemen werkt. In tegenstelling tot *native apps*, die vaak aparte codebases vereisen voor verschillende besturingssystemen, wat resulteert in extra werk en dus hogere kosten. Dit betekent echter niet dat een PWA altijd de beste keuze is; als een applicatie afhankelijk is van de bluetooth-verbinding, locatie of camera van het apparaat, is het noodzakelijk om te kiezen voor een *native app* omdat PWA's hier geen toegang tot hebben [1].

Veiligheid vormt een andere belangrijke overweging. PWA's vertrouwen op het HTTPS-protocol voor beveiliging, terwijl *native apps* ingebouwde beveiligingsfuncties bieden en moeten voldoen aan strenge normen van appstores. Vooral in sectoren zoals de gezondheidszorg en het bankwezen, waar gevoelige gegevens worden verwerkt, kunnen de ingebouwde beveiligingsfuncties van *native apps* doorslaggevend zijn [1].

Voor gebruikers draait de keuze tussen PWA's en *native apps* om bruikbaarheid en ervaring. PWA's moeten niet gedownload worden en gebruiken minder opslagruimte op apparaten. *Native apps* bieden betere functionaliteit, personalisatie en prestaties vanwege hun toegang tot de hardware functies. Hoewel PWA's kostenefficiënt zijn en snel kunnen worden geïmplementeerd, is de keuze tussen beide afhankelijk van factoren zoals de aard van de applicatie, doelgroep en beschikbare middelen. Een grondige analyse van deze aspecten is essentieel voor bedrijven om een weloverwogen beslissing te nemen die aansluit bij hun specifieke behoeften en doelstellingen [1].

Aan de andere kant hebben PWA's ook enkele nadelen. Hoewel PWA's in populariteit toenemen, was de ondersteuning voor iOS-apparaten lange tijd beperkt. Dit vormde een uitdaging voor bedrijven die op deze platforms actief zijn. Dit betekende dat PWA's op iOS-apparaten niet dezelfde functionaliteiten en mogelijkheden hadden als op andere platformen, zoals Android-apparaten.

Sommige functies en technologieën die essentieel zijn voor de werking van PWA's, zoals het offline gebruik, pushmeldingen en toegang tot bepaalde hardware functies, werden niet volledig ondersteund. Hierdoor ondervonden iOS-gebruikers een verminderde functionaliteit en gebruikerservaring [2].

Echter, met de recente verbeteringen in de ondersteuning van PWA's op iOS-apparaten, worden de nadelen geleidelijk minder relevant. Dit maakt dat de PWA een aantrekkelijke optie is voor bedrijven die een app willen ontwikkelen die op verschillende platforms werkt. Niettemin hebben PWA's nog steeds beperkte toegang tot sommige apparaatfuncties, zoals vingerafdruk-ID en sensoren, in vergelijking met *native apps*. Dit kan de mogelijkheden van PWA's beperken in bepaalde situaties. Een ander minpunt is het gebrek aan kennis bij zowel ontwikkelaars als gebruikers over hoe ze PWA's optimaal kunnen benutten [27].

3.1.1 Voorbeelden PWA's

Twitter heeft met Twitter Lite gekozen voor de PWA-technologie. Dit stelt gebruikers in staat om Twitter te ervaren met minimale data- en opslagvereisten, terwijl de applicatie snel laadt met een trage internetverbinding. Bovendien is het veerkrachtig op onstabiele netwerken en neemt het minder dan 1 MegaByte (MB) aan ruimte in op een mobiel apparaat, in tegenstelling tot 23.5 MB die nodig is om de *native app* te installeren. [3]

Alibaba.com, 's werelds grootste online business-to-business (B2B) handelsplatform, heeft zijn website geüpgraded naar een PWA. Met een wereldwijde aanwezigheid in meer dan tweehonderd landen en regio's, heeft Alibaba.com een aanzienlijke toename gezien in het aantal maandelijkse actieve gebruikers sinds de overgang naar een PWA. Op iOS-apparaten is er een stijging van 14 %, terwijl op Android-apparaten een groei van 30 % is waargenomen. Deze groei wordt toegeschreven aan het feit dat gebruikers nu de mogelijkheid hebben om de website toe te voegen aan hun startscherm zonder deze eerst te moeten downloaden via de appstore. Statistieken tonen aan dat gebruikers die ervoor kiezen om de PWA toe te voegen aan hun startscherm tot wel vier keer meer interactief zijn in vergelijking met gebruikers die dat niet doen [3].

AliExpress is een internationaal bekend e-commerce platform dat een divers scala aan producten aanbiedt aan klanten wereldwijd. Het heeft met succes een cross-browser PWA ontwikkeld. Deze PWA is specifiek ontworpen om te functioneren op diverse webbrowsers, ongeacht hun type of versie. De implementatie van deze PWA heeft ervoor gezorgd dat AliExpress toegankelijk is voor het brede bereik van het web, waardoor het een verdubbeling heeft gezien in het aantal nieuwe gebruikers. Bovendien is de tijd die gebruikers op het platform spenderen met maar liefst 74 % toegenomen [3].

Billings Gazette, een dagelijkse krant in de Verenigde Staten, heeft met de overstap naar een PWA ook de functionaliteit in offline modus geïmplementeerd. Dit stelt abonnees in staat om ook artikelen te lezen wanneer ze offline zijn [3].

Dit zijn voorbeelden van hoe Twitter, Alibaba, AliExpress en Billings Gazette PWA-technologie hebben gebruikt om de gebruikerservaring te verbeteren en betere resultaten te behalen in termen van betrokkenheid, conversies en prestaties.

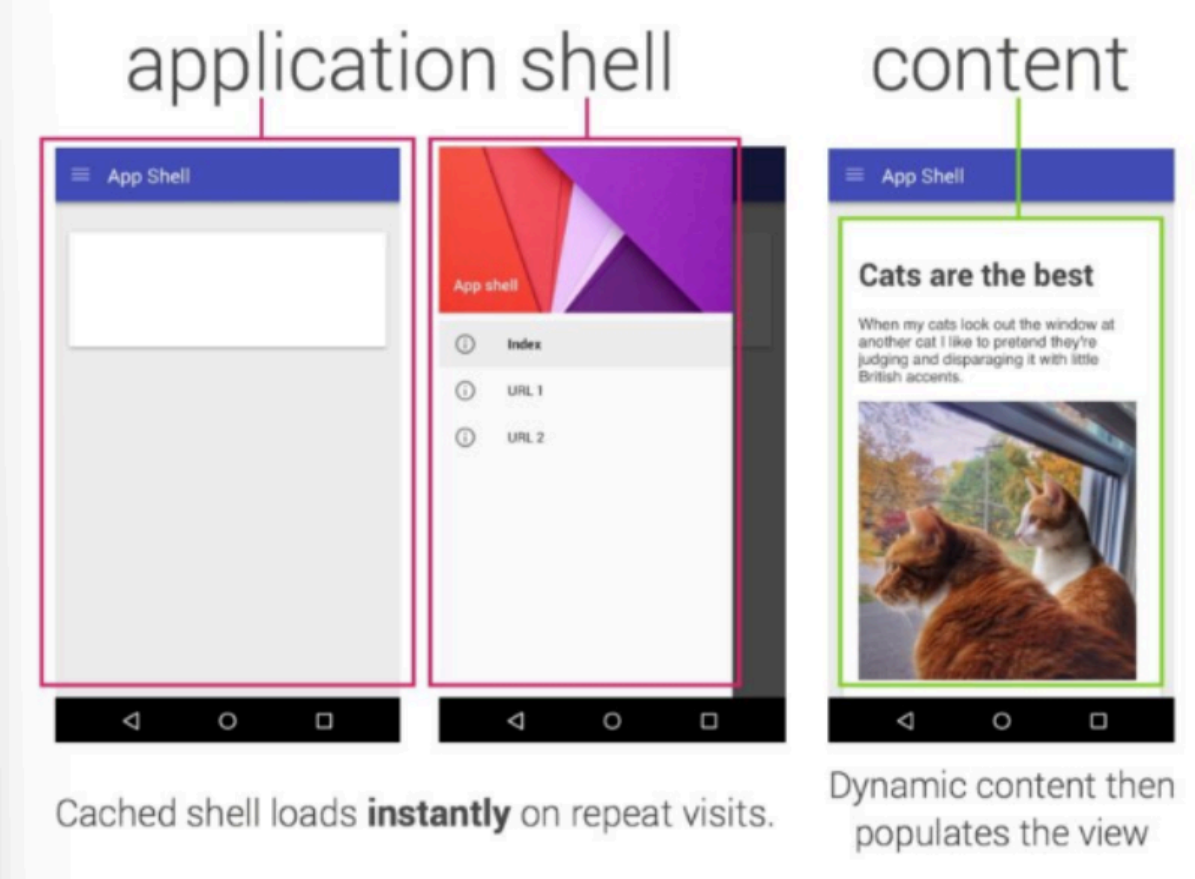
3.1.2 Technologie achter PWA's

Progressive Web Apps (PWA's) zijn een innovatieve benadering van webontwikkeling, waarbij de nadruk ligt op het bieden van een app-achtige ervaring binnen de webbrowser. De kernconcepten van PWA's zijn het *App Shell Model*, *Service Workers* en *Manifest Files* [4].

Het *App Shell Model* introduceert een minimalistische set van HTML, CSS en JavaScript. Het is het skelet van de PWA met een dynamisch gedeelte waarin de inhoud ervan verandert bij elke paginaweergave. Deze *shell* wordt gecachet op het apparaat van de gebruiker bij het eerste bezoek, waardoor snelle en betrouwbare prestaties mogelijk zijn bij toekomstige bezoeken [4].

https://tutorialzine.com/2016/09/everything-you-should-know-about-progressive-web-apps

Figuur 1: App shell model

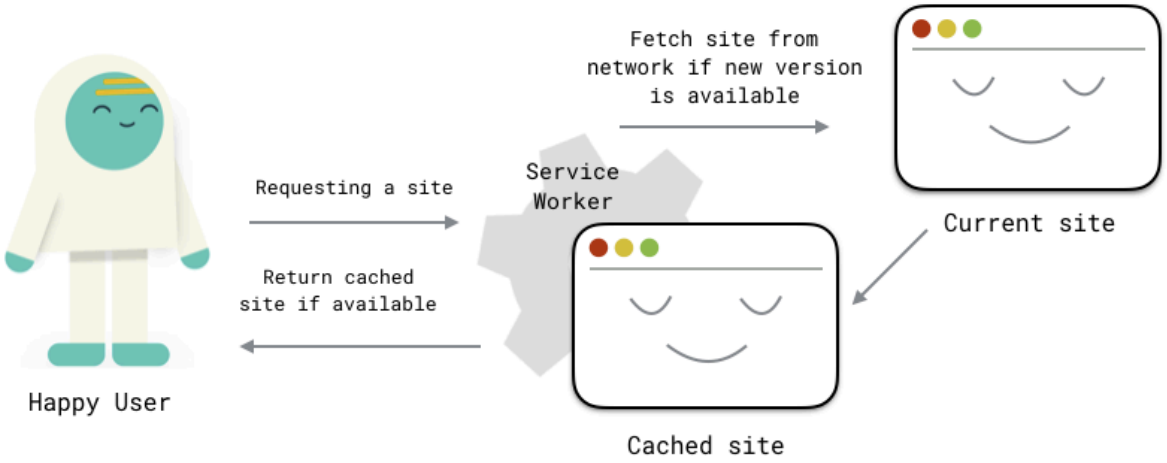


Service Workers zijn een cruciaal onderdeel van PWA's, omdat ze *scriptable* netwerk proxies zijn die tussen de webapplicatie, de browser en het netwerk zitten. Ze kunnen bekeken worden als een

tussenstation tussen het apparaat en het internet. Ze stellen PWA's in staat om controle te krijgen over netwerk verzoeken, resources te cachen en offline functionaliteit mogelijk te maken. *Service Workers* spelen ook een rol bij het mogelijk maken van pushmeldingen en het verbeteren van de algehele gebruikerservaring [4] [30].

https://www.netlify.com/blog/2017/10/31/service-workers-explained/

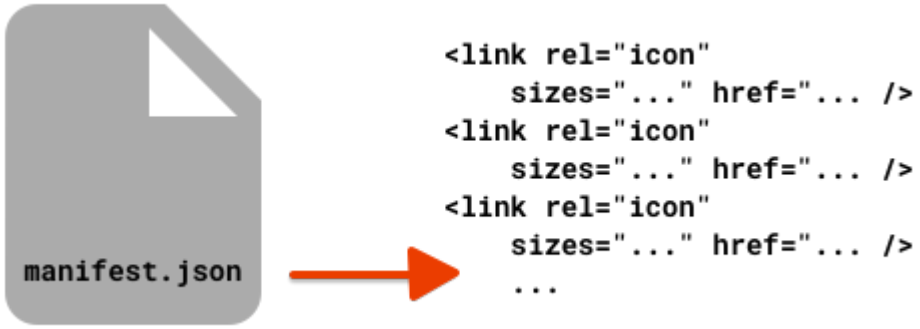
Figuur 2: Netwerk verzoeken



Manifest Files zijn JSON-bestanden waarin ontwikkelaars schrijven hoe de PWA wordt gepresenteerd aan de gebruikers van de PWA. De *Manifest Files* bevatten alle nodige informatie die het apparaat moet weten over de webapplicatie. Hierin kunnen eigenschappen zoals de naam van de app, de locatie van iconen, het versienummer en de auteur worden gespecificeerd [4] [31].

https://cube-interactive.com/what-is-web-app-manifest/

Figuur 3: Json image



3.1.3 Tools gebruikt voor het ontwikkelen van PWA's

React JS

React JS is een *open source* library die wordt gebruikt voor het ontwikkelen van *Single Page Applications (SPA's)* of mobiele applicaties in JavaScript. SPA's zijn applicaties of websites waarbij bepaalde componenten of functionaliteiten van een pagina worden herladen en weergegeven in plaats van de volledige pagina. Ontwikkelaars hebben de vrijheid om selectief componenten te kiezen uit deze bibliotheek, in tegenstelling tot standaard frameworks waarin ze gebonden zijn aan de structuur die het framework oplegt. Het wordt veel gebruikt vanwege zijn gebruiksgemak en ondersteuning van een grote community van ontwikkelaars [6] [7].

Vue.js

Vue.js is een JavaScript-framework ontwikkeld door Evan You en is officieel gelanceerd in 2014. Het framework wordt met name ingezet voor de ontwikkeling van SPA's. Vue.js heeft een compact formaat, variërend van slechts 18 tot 21 kilobyte (kB), wat resulteert in een snelle download en directe bruikbaarheid. Het staat bekend om zijn eenvoudige aanpak voor dynamische inhoudsweergave. Het is geschikt voor zowel beginners als ervaren ontwikkelaars, met een soepele leercurve en schaalbaarheid voor projecten van elke omvang [6] [8].

Angular

Angular is een op TypeScript gebaseerd *open source* frontend-framework dat ontwikkeld is door Google. Het wordt gebruikt voor het bouwen van dynamische, schaalbare en onderhoudbare webapplicaties. Angular heeft veel ingebouwde features en *libraries*. Dit zorgt ervoor dat Angular een grotere *packagesize* heeft en meer opslagruimte vereist. Bovendien betekent dit dat er weinig nood is aan *third party libraries* die bottlenecks kunnen worden bij een (security) upgrade. Angular heeft een steile leercurve, wat te wijten is aan standaard *dependencies*, *boilerplating* en complexe *syntaxis*. Het heeft een gedefinieerde app architectuur, bedoeld om grote applicaties te helpen een bepaalde standaard van architectuur te behouden door dit te forceren [6] [9].

Workbox

Workbox wordt beschouwd als een essentieel hulpmiddel voor het ontwikkelen van PWA's, met als doel het vereenvoudigen van het beheer van *service workers* en de cache-opslag logica. Het platform bestaat uit *open source libraries* die zijn ontworpen om dit beheer te vergemakkelijken. In plaats van rechtstreeks te werken met de complexe Application Programming Interfaces (API's) die door de browser worden verstrekt, kunnen ontwikkelaars gebruikmaken van deze *libraries*, die eenvoudigere en toegankelijke interfaces bieden. Dit stelt ontwikkelaars in staat om sneller en efficiënter te werken bij het bouwen van PWA's [6] [11].

3.1.4 Kritische overwegingen en uitdagingen

Progressive web-apps (PWA's) bieden een geïntegreerde ervaring tussen web- en mobiele applicaties, maar de ontwikkeling ervan wordt geconfronteerd met diverse uitdagingen. Enkele van de voornaamste obstakels zijn:

Optimalisatie van de prestaties: Het vinden van een evenwicht tussen uitgebreide functionaliteit en snelle laadtijden om zowel gebruikers als een hoge ranking in zoekmachines te behouden.

Compatibiliteit met verschillende browsers: Om ervoor te zorgen dat een PWA effectief werkt op verschillende webbrowsers, dienen ontwikkelaars grondige tests uit te voeren. In sommige gevallen kan het noodzakelijk zijn om extra scripts toe te voegen, zoals *polyfills*. Deze scripts vullen de ontbrekende functionaliteit aan in oudere browsers. Op die manier kunnen ontwikkelaars hedendaagse code implementeren zonder zich zorgen te hoeven maken over de compatibiliteit met verouderde versies van browsers.

Offline functionaliteit: hoewel PWA's uitblinken in offline toegankelijkheid, kan het implementeren van effectieve offline modus en caching-strategieën complex zijn.

Security: Veiligheid is een cruciaal aspect bij de ontwikkeling van PWA's. Het beschermen van gegevens en privacy is van groot belang, met name in het licht van de functionaliteiten zoals *service workers*, die in staat zijn om netwerk aanvragen te onderscheppen. Gezien het feit dat service workers directe toegang hebben tot netwerk verzoeken, moeten ontwikkelaars strenge maatregelen nemen om ervoor te zorgen dat deze toegang niet misbruikt wordt. Dit omvat het implementeren van versleutelingstechnieken, het beheren van toegangsrechten en het volgen van best practices voor gegevensbeveiliging, zoals het minimaliseren van gegevensverzameling en het veilig opslaan van gebruikersinformatie.

Gebruikersbetrokkenheid: Concurreren met *native apps* op het gebied van gebruikerservaring en betrokkenheid vereist een strategische implementatie van app-achtige functies zoals pushmeldingen en *splash screens*.

3.1.5 Digital Markets act

De Digital Markets Act (DMA) heeft als doel om digitale markten te reguleren door de macht van grote online platformen te beperken. Deze wetgeving richt zich met name op zogenaamde gatekeeper-platformen die een aanzienlijke invloed hebben op de online economie. Voorbeelden hiervan zijn onder meer Apple's App Store en Google's Play Store, die momenteel de belangrijkste distributiekanalen zijn voor mobiele applicaties.

3.1.6 Implicaties voor PWA's

Distributie: Onder de DMA kunnen gatekeeper-platformen verplicht worden alternatieve distributiemethoden voor PWA's toe te staan, waardoor ontwikkelaars meer vrijheid krijgen bij het aanbieden van apps aan gebruikers buiten de traditionele appstores om.

Toegankelijkheid: De maatregelen van de DMA kunnen de toegang tot PWA's verbeteren door beperkingen op externe distributiekanalen op te heffen. Dit kan het gebruiksgemak vergroten voor gebruikers bij het ontdekken en installeren van PWA's, wat met name gunstig is voor kleinere projecten.

Zichtbaarheid: PWA's profiteren van een gelijk speelveld doordat ze niet onderworpen zijn aan de algoritmen en goedkeuringsprocessen van appstores. Dit kan leiden tot grotere zichtbaarheid voor PWA's ten opzichte van traditionele apps, die afhankelijk zijn van appstores voor publiciteit.

Het ontbreken van de noodzaak om een app te downloaden via de appstore kan daarom voordelig zijn voor PWA’s. Het kan leiden tot meer concurrentie, innovatie en keuzevrijheid voor zowel ontwikkelaars als consumenten op de digitale markt.

3.2 Conclusie

Progressive Web Apps (PWA's) bieden een veelbelovende benadering voor het leveren van webgebaseerde applicaties met een verbeterde gebruikerservaring en functionaliteit. Door hun vermogen om offline te werken, snelle laadtijden te bieden en te functioneren als native apps, hebben PWA's het potentieel om de digitale markt te transformeren en de concurrentiepositie van ontwikkelaars te versterken. De regelgevende maatregelen, zoals de Digital Markets Act (DMA), kunnen de verdere groei van PWA's stimuleren door beperkingen op externe distributiekanalen op te heffen, wat leidt tot een grotere toegankelijkheid en keuzevrijheid voor zowel ontwikkelaars als consumenten. Met hun flexibiliteit en voordelen voor zowel bedrijven als gebruikers, vertegenwoordigen PWA's een belangrijke evolutie in de manier waarop mobiele applicaties worden ontwikkeld, gedistribueerd en gebruikt op de digitale markt.

4. Casestudy

4.1 Implementatie van PWA in Uber

Deze casestudy is gericht op het onderzoeken van hoe de implementatie van de PWA-technologie heeft bijgedragen aan het verbeteren van Uber. De keuze voor Uber als casestudy is verantwoord vanwege het feit dat Uber een leidend bedrijf is in de *ride-sharing* industrie, waarbij de implementatie van de PWA-technologie impact heeft gehad op de gebruikservaring en de prestaties van de mobiele webapplicatie.

De bevindingen van deze casestudy bieden inzicht in het beantwoorden van zowel de hoofdvraag als de verschillende deelvragen van dit onderzoek. Door het analyseren van de implementatie van PWA-technologie in andere scenario's, zoals bij Uber, kunnen conclusies getrokken worden over de potentiële voordelen en uitdagingen ervan voor de applicatie van Tinkerstal 'T Boesehof.

4.1.1 Kenmerken van PWA's geïmplementeerd bij Uber

1. Snelle laadtijden:

Een van de belangrijkste kenmerken van PWA's is het vermogen om snel te laden met een onstabiele netwerkverbinding. De PWA van Uber is ontworpen om binnen enkele seconden te laden. Dit resulteert in een vlotte ervaring voor gebruikers, wat vooral belangrijk is voor een service als Uber, waar klanten vaak dringend een rit nodig hebben.[5]

2. Betrouwbare prestaties:

PWA's bieden betrouwbare prestaties onder verschillende omstandigheden. De PWA van Uber kan goed functioneren bij een slechte internetverbinding, waardoor gebruikers overal en altijd toegang hebben tot de service. Dit verbetert de algehele toegankelijkheid van de applicatie.[5]

3. Offline functionaliteit:

Een ander belangrijk aspect van PWA's is hun vermogen om offline te werken. De PWA van Uber stelt gebruikers in staat om acties uit te voeren, zoals het plannen van ritten, wanneer ze geen internetverbinding hebben. Deze offline functionaliteit vergroot de bruikbaarheid van de applicatie en zorgt ervoor dat gebruikers in offline modus toegang hebben tot bepaalde essentiële functies.[5]

4.1.2 Technische Inzichten van m.uber's PWA-implementatie

Het artikel "Building m.uber: Engineering a High-Performance Web App for the Global Market" biedt een blik achter de schermen in de technische aspecten van het bouwen van een PWA zoals m.uber [25]. Door deze bevindingen samen te brengen, ontstaat een inzicht in de technologische aanpak die Uber gebruikt om een snelle, betrouwbare en toegankelijke *ride-sharing* service aan te bieden.

4.1.3 Technische Oplossingen Gebruikt in m.uber's PWA

Gebruik van Preact en Webpack

Om de bundelgrootte van de applicatie te beperken wordt Preact gebruikt voor het renderen van de componenten, terwijl Webpack wordt ingezet voor dynamische *bundle splitsing* en *tree shaking* functionaliteiten. Deze aanpak maakt het mogelijk om code op te splitsen in verschillende bundels die vervolgens op verzoek of parallel geladen kunnen worden, terwijl overbodige code wordt geëlimineerd [25].

Server Side Rendering (SSR)

M.uber reageert op het initiële verzoek door Preact op de server te renderen, waardoor de inhoud bijna onmiddellijk wordt geladen alvorens alle JavaScript-bundels gedownload zijn [25].

Service workers

Service workers worden gebruikt om URL-verzoeken te onderscheppen, waardoor caching van inhoud mogelijk is, inclusief de initiële HTML-respons en JavaScript-bundels. Dit zorgt ervoor dat de pagina snel opnieuw geladen kan worden wanneer de verbinding met het netwerk onderbroken wordt [25].

Optimalisatie van bundelgrootte

Door het beperken van bundelgrootte draagt m.uber bij aan snelle laadtijden en een efficiënte werking van de applicatie op langzame netwerken [25].

4.2 Toepassing op Tinkerstal 'T Boesehof

De technische benaderingen gebruikt in m.uber's PWA kunnen worden toegepast bij het optimaliseren van de webapplicatie van Tinkerstal 'T Boesehof. Door vergelijkbare benaderingen toe te passen, zoals het minimaliseren van bundelgrootte, *server-side* rendering en caching voor offline functionaliteit, kan Tinkerstal 'T Boesehof een verbeterde gebruikerservaring bieden aan haar gebruikers, ongeacht de internetverbinding en apparaat prestaties.

Door het integreren van de technische inzichten van zowel het artikel van Uber als het artikel over m.uber's PWA-implementatie, kan deze casestudy aantonen hoe PWA-technologie de gebruikerservaring en toegankelijkheid kunnen verbeteren, niet alleen voor Uber, maar ook voor andere applicaties zoals die van Tinkerstal 'T Boesehof.

5. Uitvoering

5.1 Proof of concept

Om te bepalen of een PWA de juiste oplossing is voor Tinkerstal 'T Boesehof, is een *proof of concept* (PoC) ontwikkeld en getest. Hierbij is de toegankelijkheid, snelheid, offline functionaliteit, lokale opslag en *caching* van een PWA geëvalueerd om te bepalen of de PWA technologie werkelijk geschikt is voor deze case.

Deze PoC biedt een antwoord op de tweede en vierde onderzoeksvraag.

Als PoC is de applicatie MemoSphere [22] ontwikkeld. Deze applicatie biedt basisfunctionaliteiten voor het aanmaken, lezen, bewerken en verwijderen van notities. De broncode voor de applicatie is publiek op Github beschikbaar [23].

5.1.1 Functionaliteit en belang van dit applicatie type

De MemoSphere-applicatie dient voor dit onderzoek als een representatief model voor het testen van de functionaliteit van een PWA in een live omgeving.

Notities moeten snel inladen en het is van belang dat ze *cross-platform* beschikbaar zijn zodat ze op verschillende apparaten kunnen worden geraadpleegd. Zowel via een browser als een smartphone moet toegang mogelijk zijn, hetzij online of offline. De PWA-technologie ondersteunt deze functionaliteiten.

De uitgevoerde testen belichten aspecten van een PWA die een toegevoegde waarde kunnen bieden voor de applicatie van Tinkerstal ‘T Boesehof:

- Efficiënte toegang tot informatie over merries en veulens met een snelle laadtijd ter verbetering van de gebruiksvriendelijkheid van de applicatie;
- Cross-platform toegang tot de applicatie en in staat zijn informatie over merries en veulens te kunnen raadplegen, met minimale configuratie;
- Zowel online als offline toegang hebben tot gegevens, bijvoorbeeld in stallen met beperkte internetverbinding.

5.1.2 Architectuur

De PWA is ontwikkeld met identieke technologieën als de Tinkerstal-applicatie om een zo goed mogelijke replicatie te bereiken. De backend van de applicatie is opgezet met het gebruik Java Spring Boot [12], terwijl voor de *frontend* Vue.js is gebruikt [13]. Tenslotte is net als in de oorspronkelijke applicatie Tailwind CSS toegepast [14].

De applicatie wordt *gehost* op een lokaal beheerde Linux-server, toegankelijk via een publiek IP-adres dat is geconfigureerd met *port forwarding*. Het *frontend* Node proces wordt beheerd door PM2 (Process Manager 2) [15], een *process manager* voor Node.js [16]. De Spring Boot applicatie wordt eveneens op deze *server gehost*.

In eerste instantie werd de applicatie zowel voor de frontend als de backend in HTTP (Hypertext Transfer Protocol) uitgevoerd. Het werd snel duidelijk dat de PWA geen HTTP ondersteunde en dat bij gebruik ervan er geen mogelijkheid was om de PWA te installeren op mobiele apparaten. Een SSL (Secure Socket Layer) certificaat of *self signed* SSL certificaat lost dit op. Echter, vanwege budgettaire beperkingen en de beschikbaarheid van gratis betrouwbare aanbieders voor SSL-certificaten, wordt er een gratis alternatief gebruikt.

Als oplossing in deze toepassing wordt er gebruikgemaakt van de Zero Trust [10] functionaliteit van Cloudflare [17]. Een verbinding met de lokale applicatie is tot stand gebracht door middel van een tunnel. Cloudflare staat voornamelijk bekend als een betrouwbare DNS server tussen website en bezoeker [18]. Via deze uitvoering is het publieke domeinnaam verbonden met een interne poort op de lokale server.

Deze tunnel maakt externe toegang tot de gehoste applicatie mogelijk, waarbij externe gebruikers via het openbare adres kunnen communiceren met de interne server. Cloudflare zorgt voor alle SSL certificaten en garandeert daarbovenop ook nog extra beveiliging.

Het onderliggende mechanisme dat deze certificaten mogelijk maakt is Universal SSL [20]. Deze certificaten zijn gratis, uniek en openbaar betrouwbaar. Ze bieden een kostenefficiënt alternatief voor traditionele SSL-certificaten, geschikt voor projecten van deze omvang, maar mogelijk minder geschikt voor grotere projecten [19].

5.1.3 PWA implementatie in MemoSphere app

Voor de implementatie is gebruikgemaakt van de officiële Vite PWA-plugin [32]. Het verbeteren van de compatibiliteit van een PWA kan effectief worden gerealiseerd binnen het Vue (Vite) framework door de beschikbare richtlijnen te volgen zoals uiteengezet in de officiële documentatie [21]. De documentatie bevat enkele belangrijke stappen voor de opzet van de applicatie:

- Registratie van de *service worker*
- Het automatisch genereren van een manifest
- Aanpassingen noodzakelijk in het index.html-bestand
- Minimale vereisten voor PWA's

De PWA-plugin genereert automatisch het manifest tijdens het bouwen van de applicatie, waarbij het manifest wordt opgeslagen in de publieke map van de frontend applicatie. Hierdoor kan de browser de configuratie overnemen voor de installatie van de webapplicatie op verschillende platformen, inclusief het instellen van app iconen, de naam van de app en de beschrijving ervan [21].

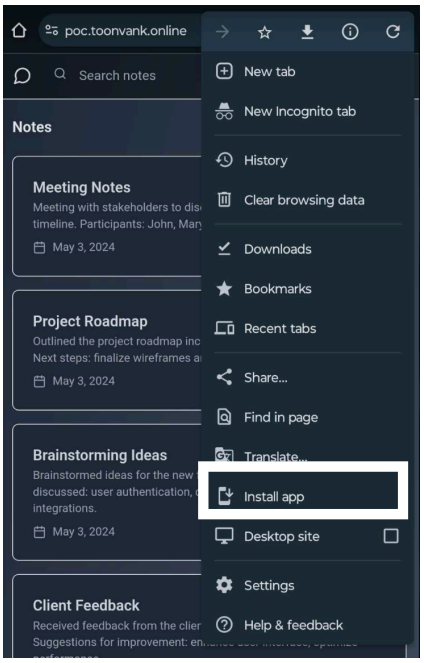
De *service worker* maakt offline functionaliteit beschikbaar door bronnen op te slaan voor later gebruik. Na registratie neemt de *service worker* de controle over de app en downloadt de bronnen die offline toegang mogelijk maken op de achtergrond. De registratie van de *service worker* vindt plaats in het vite.config.js-bestand van de applicatie [21].

Om de PWA functionaliteit volledig te voltooien, zijn verschillende afbeeldingen nodig, zoals *favicons*. Deze afbeeldingen zijn geconverteerd in diverse formaten om compatibiliteit te garanderen met verschillende apparaten en resoluties. Ze zijn op verschillende plaatsen in de applicatie gedefinieerd, zoals in het index.html-bestand en het vite.config.js-bestand. De favicon afbeeldingen bevinden zich in de public folder [21].

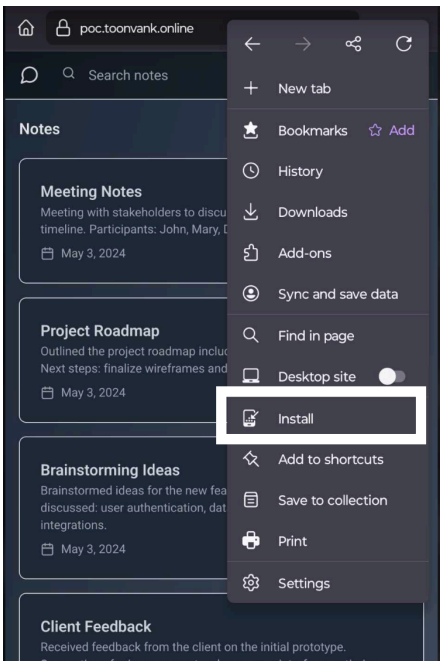
5.1.4 PWA installatieprocedure

Zodra deze configuratie is voltooid, zal de browser zowel op mobiele als op desktop apparaten aanbevelen om de PWA te installeren bij een bezoek aan de website. In Firefox en Chrome (Android) is er de mogelijkheid om de app te installeren via het menu. Na installatie is de applicatie beschikbaar op het startscherm, net als een reguliere app die is gedownload vanuit een appstore.

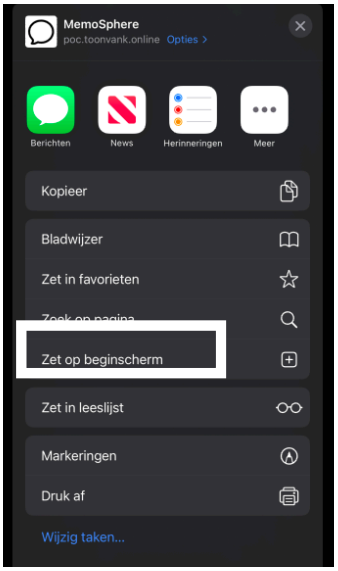
Figuur 4: Schermafbeelding App Installeren Menu (Android - Chrome)



Figuur 5: Schermafbeelding App Installeren Menu(Android - Firefox)

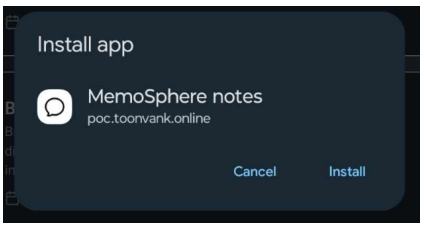


Figuur 6: Schermafbeelding App Installeren Menu (iOS - Safari)

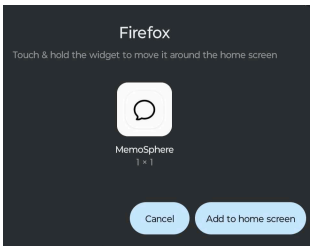


Figuur 4, Figuur 5, Figuur 6: In het menu van de browsers zijn opties zoals “Install app” (Chrome - Android) en “Install” (Firefox - Android) beschikbaar voor PWA’s, terwijl voor gewone webpagina's de optie "Toevoegen aan startscherm" beschikbaar is. In Safari op iOS is hiervoor de optie “Zet op beginscherm” aanwezig. Opvallend is dat de PWA’s in Chrome en Firefox als twee aparte applicaties worden beschouwd.

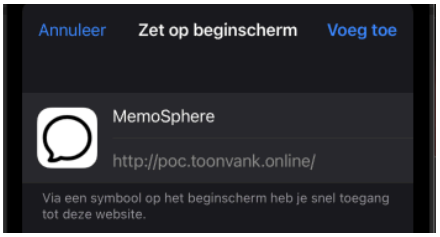
Figuur 7: Schermafbeelding App Installeren Popup (Android - Chrome)



Figuur 8: Schermafbeelding App Installeren Popup (Android - Firefox)



Figuur 9: Schermafbeelding App Installeren Popup (iOS - Safari)



Figuur 7, Figuur 8, Figuur 9: Wanneer er op de knop "Installeren" wordt geklikt, zijn volgende pop-ups voorzien voor het bevestigen van de installatie.

5.1.5 Toegankelijkheid en snelheid

Na testen van de MemoSphere applicatie blijkt dat wanneer de app geïnstalleerd wordt als PWA de toegang tot de app snel is. De app is direct beschikbaar via het startscherm. Bovendien versnelt de opbouw van een lokale cache door de *service worker* de laadtijd aanzienlijk. De prestaties van de website zijn beoordeeld door Google Lighthouse [33], waarbij een performance score van 99% is vastgesteld, met een laadtijd van 1.2 seconden tot de eerste elementen zichtbaar werden en een laadtijd van 2.1 seconden tot de grootste elementen zichtbaar waren [Bijlage A] [26].

5.1.6 Offline functionaliteit

Tijdens het testen van de offline functionaliteit binnen de applicatie zijn enkele bevindingen naar voren gekomen. Alle pagina's die met internettoegang zijn bezocht, behouden hun functionaliteit en verschillende componenten blijven functioneel. De navigatieknoppen werken naar behoren. Functionaliteiten die afhankelijk zijn van API-oproepen, zoals het aanmaken en bewerken van gegevens, zijn offline niet mogelijk. Een oplossing hiervoor is het gebruik van een lokaal geïndexeerde

database of SQLite database, die lokaal alle objecten opslaat en deze synchroniseert met de server zodra de verbinding met het internet hersteld is. Ten slotte kan het laden van bepaalde notities langer duren of in sommige gevallen zelfs falen bij offline gebruik, hoewel deze situaties niet direct zijn gerelateerd aan de *service worker* van de PWA, maar eerder aan de onderliggende architectuur van de applicatie en de interactie met de API.

5.1.7 Toegang tot lokale resources

Om de toegang tot lokale bronnen te testen, zijn twee van de meest relevante Android *resources* getest met deze PWA: toegang tot lokale bestanden en toegang tot de camera. Deze keuze is gemaakt vanwege de noodzaak om afbeeldingen en bestanden te uploaden in de applicatie van Tinkerstal 'T Boesehof. Bij het aanmaken van een nieuwe notitie is de optie toegevoegd die het mogelijk maakt om een bijlage toe te voegen aan de notitie. Dit kan zowel via de camera als via de bestanden van het apparaat.

Camera

Voor de toegang tot de camera wordt de 'vue-camera-lib' plugin [28] gebruikt, waarmee uitgebreide functionaliteit beschikbaar is voor het weergeven van camerabeelden en foto's. Hierdoor zijn alle camera's, zowel op mobiele apparaten als op desktops, beschikbaar voor gebruik.

Bestanden

Voor de toegang tot bestanden wordt SweetAlert [29] gebruikt, waardoor het kiezen van een bestand mogelijk is, zodat gebruikers toegang hebben tot foto's die op het apparaat zijn opgeslagen.

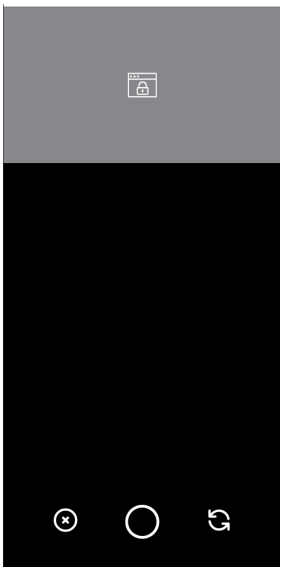
Capacitor

Tenslotte zijn de twee genoemde functionaliteiten getest met behulp van de Ionic Capacitor [36] implementatie in Vue. Capacitor is een *open source native runtime* voor het bouwen van *Web Native* applicaties, waarmee *cross-platform* iOS, Android en PWA's kunnen worden gecreëerd met behulp van JavaScript, HTML en CSS [36]. Capacitor staat bekend om zijn vermogen om de ontwikkeling van *cross-platform* applicaties te versnellen en te verbeteren dankzij zijn intuïtieve functionaliteit.

Capacitor beschikt over een reeks plug-ins die kunnen worden geïmplementeerd, waardoor cross-platform toegang tot systeembronnen mogelijk is, zoals geolocatie, haptische functies, lokale toegang tot het netwerk, pushmeldingen en andere functionaliteiten [37].

Met behulp van Capacitor is de toegang tot de camera en de bestanden onderzocht binnen de MemoSphere PWA. De bevindingen van het experiment tonen aan dat zowel de camera functionaliteit als het bestandssysteem toegankelijk zijn door het gebruik van de Capacitor plug-ins.

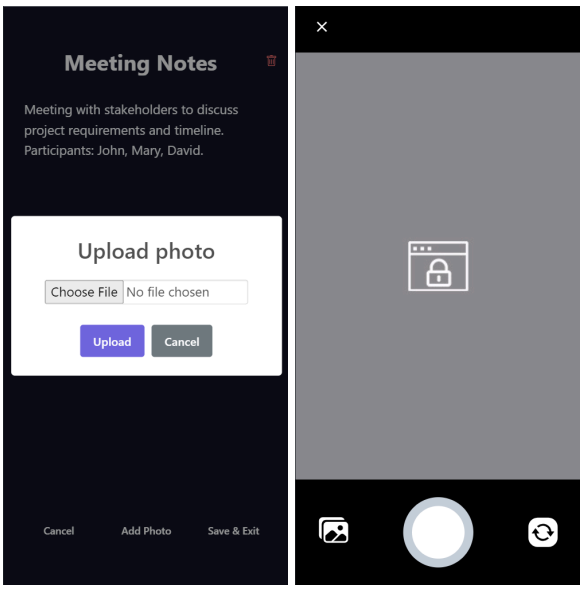
Figuur 10:Schermafbeelding vue-camera-lib plugin



Figuur 11: Schermafbeelding @capacitor/camera plugin

Figuur 10, Figuur 11: Respectievelijke implementatie met 'vue-camera-lib' plugin en '@capacitor/camera' plugin

Figuur 12: Schermafbeelding bestandstoegang SweetAlert



Figuur 12, Figuur 11: Bestandstoegang met SweetAlert & bestandstoegang met Capacitor (file picker linksonderaan)

Figuur 13: Schermafbeelding code snippet @capacitor/filesystem

```
async writePhotoToFileSystem(imageData) {
  try {
    const fileName = `photo_${Date.now()}.jpg`;
    const filePath = `${Directory.Documents}/${fileName}`;

    await Filesystem.writeFile( options: {
      path: filePath,
      data: imageData,
      directory: Directory.Documents,
      encoding: Encoding.DataUrl
    });

    console.log('Photo saved to filesystem:', filePath);
  } catch (error) {
    console.error('Error saving photo to filesystem:', error);
  }
}
```

Figuur 13: bestand schrijven naar systeem met "@capacitor/filesystem" plugin

Er is vastgesteld dat het door verschillende aanwezige *plugins* nog altijd mogelijk is om toegang te krijgen tot Android- componenten, net zoals een *native* applicatie, met de nodige extra configuratie. Hierbij zijn verschillende mogelijkheden beschikbaar, afhankelijk van het doel. Capacitor versnelt dit proces en biedt naast toegang tot de camera en bestanden ook toegang tot een andere set van systeembronnen, dit alles via één plugin.

5.2 PWA implementatie in Tinkerstal 'T Boesehof applicatie

Met als doel eerdere bevindingen op een grondige manier te onderzoeken en te valideren, is als experiment de applicatie van Tinkerstal ‘T Boesehof omgezet naar een PWA.

5.2.1 Proces

De implementatie van PWA-technologie in de applicatie van Tinkerstal ‘T Boesehof verliep vlekkeloos, vergelijkbaar met de integratie in MemoSphere. Alle functionaliteiten werken naar behoren wanneer er verbinding is met het internet. De applicatie is aanzienlijk sneller toegankelijk via het startscherm en biedt een gebruikerservaring die gelijkwaardig is aan die van een *native* app.

Toch zijn er enkele inconsistenties geconstateerd bij deze implementatie. Zo kan het voorkomen dat het JavaScript-bestand dat verantwoordelijk is voor het laden van alle pagina's niet gecachet wordt door de *service worker*. Dit leidt ertoe dat de website niet functioneel is zonder internetverbinding.

5.2.2 Snelheid

Wat betreft de snelheid van de applicatie is deze getest met behulp van Google Lighthouse [33], om zo het verschil te onderzoeken bij het gebruik van de PWA-technologie. Hieruit is geconcludeerd dat Google Lighthouse identieke snelheden meet bij beide implementaties [Bijlage B]. De laadtijd van de applicatie is aanzienlijk sneller door het gebruik van de cache, voorzien door de *service worker*. Dit is merkbaar bij het inladen van verschillende pagina’s in de applicatie.

5.2.3 Vergelijking MemoSphere

Indien de offline functionaliteit correct werkt met juiste caching, is ondervonden dat toegang tot de inlogfunctionaliteit en laden van paarden-elementen in de niet naar behoren werken vergelijkbaar is met bevindingen bij MemoSphere.

Bij het vergelijken van de resultaten met die van MemoSphere, is gebleken dat de uitdagingen met betrekking tot de inlogfunctionaliteit en het laden van merries niet naar behoren werken, vergelijkbaar zijn met de bevindingen bij MemoSphere. Deze uitdagingen zijn aangepakt door de invoering van een geïndexeerde database. De Vue-IDB (*indexed database*) plugin biedt de basis voor deze implementatie. Bovendien synchroniseert de plugin lokale objecten automatisch wanneer de verbinding met het internet wordt hersteld. Dit is eerder kort aan bod gekomen in onderdeel 5.1.6.

Conclusie

Hoe onderscheiden de kernkenmerken van PWA's zich van traditionele mobiele applicaties, en op welke manier dragen deze specifieke kenmerken bij aan een verbeterde gebruikerservaring voor de applicatie van Tinkerstal 'T Boesehof?

In traditionele mobiele applicaties ervaren gebruikers frequent langzame laadtijden, waardoor ontwikkelaars meer inspanningen moeten leveren om de configuratie te optimaliseren voor het bekijken van informatie op diverse platforms. PWA's lossen deze uitdagingen op door het aanbieden van geoptimaliseerde bundelgroottes, het combineren van server-side rendering met caching voor offline functionaliteiten, ongeacht de beschikbaarheid van internetverbinding en de capaciteiten van het apparaat.

Op welke manier zorgt de PWA-technologie voor offline functionaliteit en is het mogelijk om dit toe te passen voor de applicatie van Tinkerstal 'T Boesehof?

Het uitgevoerde Proof of Concept (PoC) heeft aangetoond dat de PWA-technologie een verbetering is voor de applicatie van Tinkerstal 'T Boesehof. De service worker fungeert als opslagplaats voor webpagina's en andere gegevens, terwijl de geïndexeerde database wordt gebruikt voor het opslaan van data-objecten. Dankzij de combinatie van toegankelijkheid, snelheid en offline functionaliteit biedt de PWA-technologie aanzienlijke verbeteringen in gebruiksvriendelijkheid. Bovendien maakt de uitgebreide documentatie en minimale configuratie de overstap naar een PWA eenvoudig.

Kunnen PWA's zonder internetconnectie functioneren en in hoeverre is dit van belang voor de applicatie van Tinkerstal 'T Boesehof?

Niet alle PWAs maken gebruik van deze mogelijkheid, zoals bijvoorbeeld Amazon. Sommige bedrijven, zoals Uber en AliExpress, hebben verbeteringen gemeld in hun webapplicaties door gebruik te maken van de PWA-technologie. Voor Tinkerstal 'T Boesehof is de mogelijkheid om offline de app te kunnen gebruiken belangrijk, omdat de applicatie vaak wordt gebruikt op locaties waar weinig of geen internettoegang mogelijk is.

Research Project Toegepaste Informatica

Hoe draagt het gebruik van lokale opslag en caching binnen PWA's specifiek bij aan een efficiënt beheer van gegevens voor de applicatie van Tinkerstal 'T Boesehof?

Lokale opslag in PWA's maakt offline toegang tot gegevens mogelijk, wat de gebruiksvriendelijkheid van de applicatie verhoogt in offline modus. Caching-strategieën zoals de *service worker* dragen hieraan bij door gegevens lokaal op te slaan en te beheren. Daarnaast leidt dit tot een verhoogde responsiviteit van de applicatie. Uit de uitgevoerde PoC blijkt dat de combinatie van lokale opslag en caching resulteert in een robuuste manier van gegevensbeheer.

Wat zijn de mogelijke implicaties van de Digital Markets Act voor de distributie, toegankelijkheid en zichtbaarheid van PWA's in vergelijking met native apps, met aandacht voor het ontbreken van appstore-installaties?

De Digital Markets Act (DMA) kan de distributie, beschikbaarheid en zichtbaarheid van PWA's beïnvloeden, voornamelijk omdat er geen vereiste meer is om een app te downloaden via de appstore. Dit kan resulteren in een breder bereik en meer toegankelijkheid voor PWA's, waardoor ze aantrekkelijker worden voor zowel ontwikkelaars als gebruikers. Hoewel dit kan leiden tot een toename van concurrentie, innovatie en keuzemogelijkheden, brengt het ook het risico met zich mee van verminderde centrale controle over distributie en beveiliging. Over het algemeen kunnen PWA's zonder de noodzaak van een tussenkomst van de appstores gemakkelijker worden verspreid via browsers, waardoor de distributie wordt vereenvoudigd.

Bibliografie

[1] R. Brown, „Progressive Web Apps (PWAs),” https://ionic.io/resources/articles/what-is-a-progressive-web-app-and-why-you-need-one#h-what-is-the-definition-of-progressive-eb-app-(pwa).

[2] N. Bajaj, “Performance, compatibility, and security in PWA optimization - Search my expert blog,” *Search My Expert Blog*, Jan. 15, 2024. https://blog.searchmyexpert.com/overcoming-challenges-progressive-web-app-development/

[3] K. Grzybowska, “20+ companies that use PWA and how it works for them | Divante,” *divante*, Apr. 25, 2021. https://www.divante.com/blog/companies-that-use-pwa .

[4] P. Opeyemi, “A deep dive into progressive web apps (PWA),” *DEV Community*, Apr. 20, 2024. https://dev.to/peter_opeyemi/a-deep-dive-into-progressive-web-apps-pwa-5beo

[5] Z. Lee, “Uber PWA: The perfect case study for progressive web app,” *Tigren*, Aug. 10, 2023. https://www.tigren.com/blog/uber-pwa/

[6] flareAI, “PWA’S: The Web-App link – 2024 Story - StudioLabs,” *StudioLabs*, Feb. 13, 2024. https://www.studiolabs.com/pwas-the-missing-link-between-web-and-app-a-story-from-2024/#:~:text=Browser%20Compatibility%3A%20Ensuring%20consistent%20behavior,integration%20compared%20to%20native%20apps.

[7] C. Deshpande, “The best guide to know what is react,” *Simplilearn.com*, Oct. 05, 2023. https://www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs

[8] “Vue.js.” https://vuejs.org/guide/introduction

[9] “Angular.” https://angular.io/guide/what-is-angular

[10] Cloudflare , “Tunnel | Zero Trust App Connector | CloudFlare,” Cloudflare. https://www.cloudflare.com/products/tunnel/ .

[11] “Workbox | Web.Dev,” *web.dev*. https://web.dev/learn/pwa/workbox

[12] spring.io [VMWare Tanzu], “Spring boot,” Spring Boot. https://spring.io/projects/spring-boot

[13] “Vue.js,” The Progressive JavaScript Framework | Vue.js. https://vuejs.org/.

[14] “Tailwind CSS - rapidly build modern websites without ever leaving your HTML.”, Tailwind CSS. https://tailwindcss.com/ .

[15] Keymetrics [Keymetrics Inc.], “PM2 - Home”, Keymetrics Inc. https://pm2.keymetrics.io/

[16] OpenJS Foundation, “Node.js — Run JavaScript everywhere”. https://nodejs.org/ .

[17] Cloudflare , “Connect, Protect and Build Everywhere | Cloudflare”. https://www.cloudflare.com/ .

[18] T. Van Eldijk, “Cloudflare, wat is het? En wat zijn de voor- en nadelen?”, Emble, 10 januari 2024. https://emble.nl/inzichten/cloudflare-wat-is-het-en-wat-zijn-de-voor-en-nadelen

[19] Cloudflare , “What is an SSL certificate? | Cloudflare”. https://www.cloudflare.com/learning/ssl/what-is-an-ssl-certificate/ .

[20] Cloudflare , “Free Universal SSL/TLS certificates · Cloudflare SSL/TLS docs”, Cloudflare Docs, 26 september 2023. https://developers.cloudflare.com/ssl/edge-certificates/universal-ssl/ .

[21] A. Fu, “Vite PWA”, Vite-pwa-org, 30 april 2024. https://vite-pwa-org.netlify.app/guide/ .

[22] A. Van Kimmenade, “MemoSphere”, MemoSphere. https://poc.toonvank.online/

[23] Toonvank, “GitHub - toonvank/researchProjectPOC-AON6”, GitHub. https://github.com/toonvank/researchProjectPOC-AON6.git

[24] SQLite, “SQLite Home page”. https://www.sqlite.org/ .

[25] “Building M.Uber: Engineering a High-PErformance Web App for the global market | Uber blog,” Uber Blog, Jun. 27, 2017. https://www.uber.com/en-BE/blog/m-uber/

[26] T. Pol, “Google Lighthouse: What it is & How to use it,” Semrush Blog, Mar. 24, 2023. https://www.semrush.com/blog/google-lighthouse/ .

[27] Marc Mac, „4 Unexpected Progressive Web App Limitations to iOS Users,” 25 April 2024. https://www.tigren.com/blog/progressive-web-app-limitations/

[28] AlexKratky, “GitHub - AlexKratky/vue-camera-lib: Simple library for Vue.js 3. Display the camera’s output or take a photo from the mobile camera/webcam.,” GitHub. https://github.com/AlexKratky/vue-camera-lib .

[29] Sweetalert, “GitHub - sweetalert2/sweetalert2: ✨ A beautiful, responsive, highly customizable and accessible (WAI-ARIA) replacement for JavaScript’s popup boxes. Zero dependencies. 🇺🇦,” GitHub. https://github.com/sweetalert2/sweetalert2 .

[30] Web.dev, “4 Unexpected Progressive Web App Limitations to iOS Users,” 3 December 2021. https://web.dev/learn/pwa/service-workers

[31] Ddgll, “GitHub - ddgll/vue-idb,” GitHub. https://github.com/ddgll/vue-idb .

[32] Vite-Pwa, “GitHub - vite-pwa/vite-plugin-pwa: Zero-config PWA for Vite,” GitHub. https://github.com/vite-pwa/vite-plugin-pwa .

[33] GoogleChrome, “GitHub - GoogleChrome/lighthouse: Automated auditing, performance metrics, and best practices for the web.”, GitHub. https://github.com/GoogleChrome/lighthouse .

[34] “Wet digitale markten”, *Consilium*. https://www.consilium.europa.eu/nl/policies/digital-markets-act/

[35] W. Akkerman, “Wet inzake digitale markten DMA: wat is dit en wat merk je ervan?”, ID.nl, 16 maart 2024. https://id.nl/huis-en-entertainment/computer-en-gaming/computer-accessoires/wet-inzake-digitale-markten-dma-wat-is-dit-en-wat-merk-je-ervan

[36] “Capacitor by Ionic - Cross-platform apps with web technology”, Capacitor. https://capacitorjs.com/ .

[37] Ionic, “Capacitor Plugins | Capacitor Documentation”, Capacitor. https://capacitorjs.com/docs/plugins .

Research Project Toegepaste Informatica

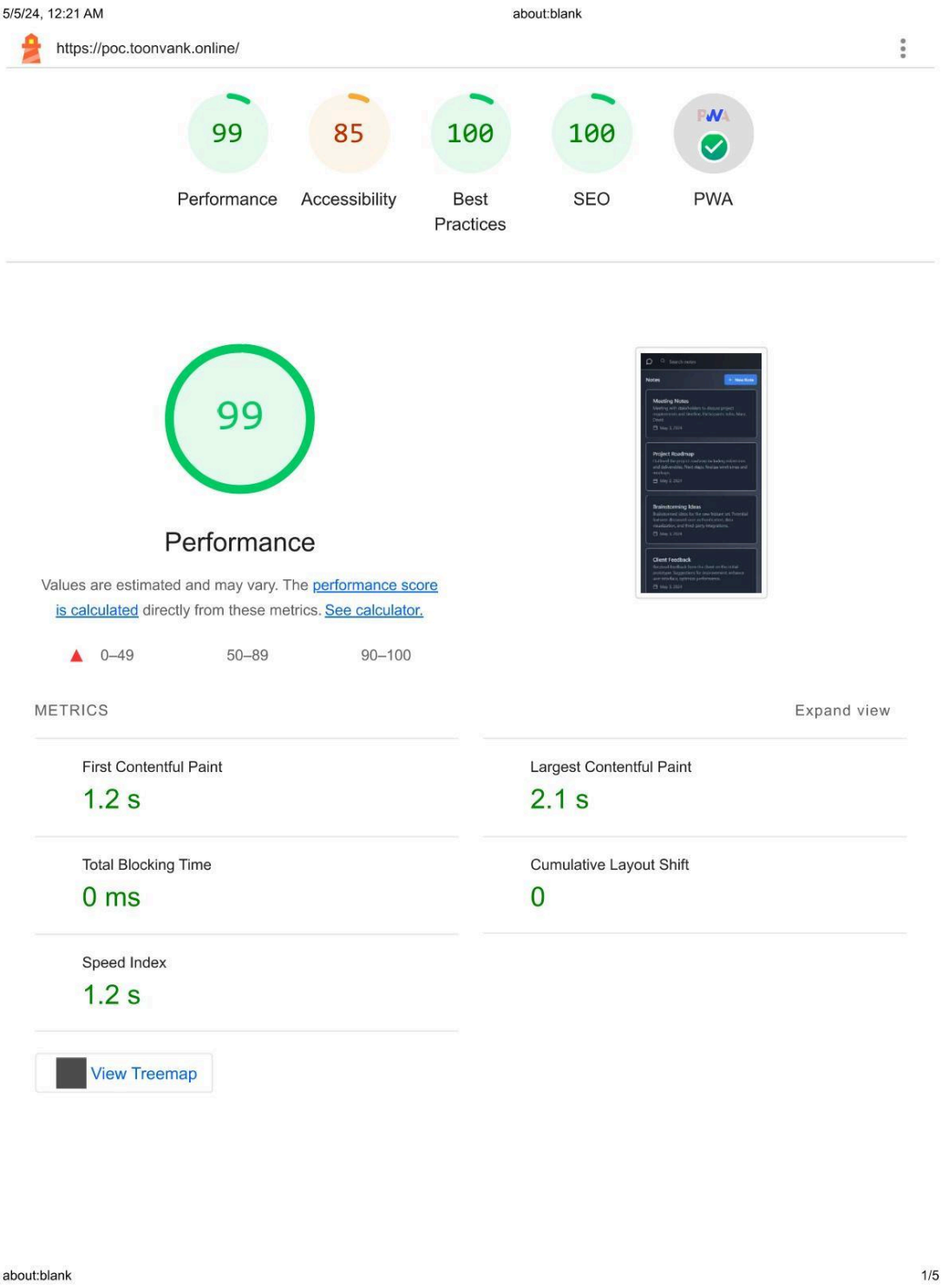
Bijlagen

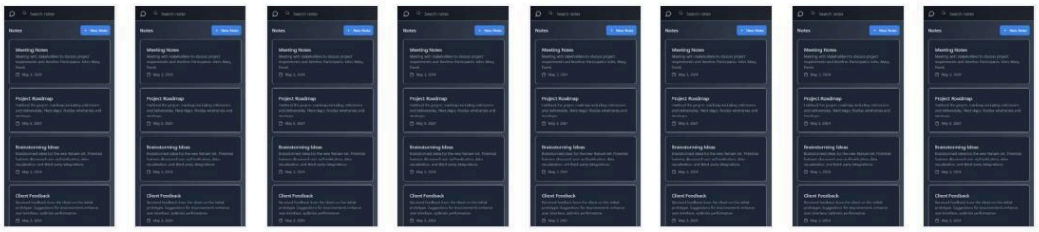
A. **Google Lighthouse Rapport MemoSphere**

B. **Google Lighthouse Rapport Tinkerstal 'T Boesehof**

Research Project Toegepaste Informatica

A. Google Lighthouse Rapport MemoSphere





Show audits relevant to: All FCP LCP TBT

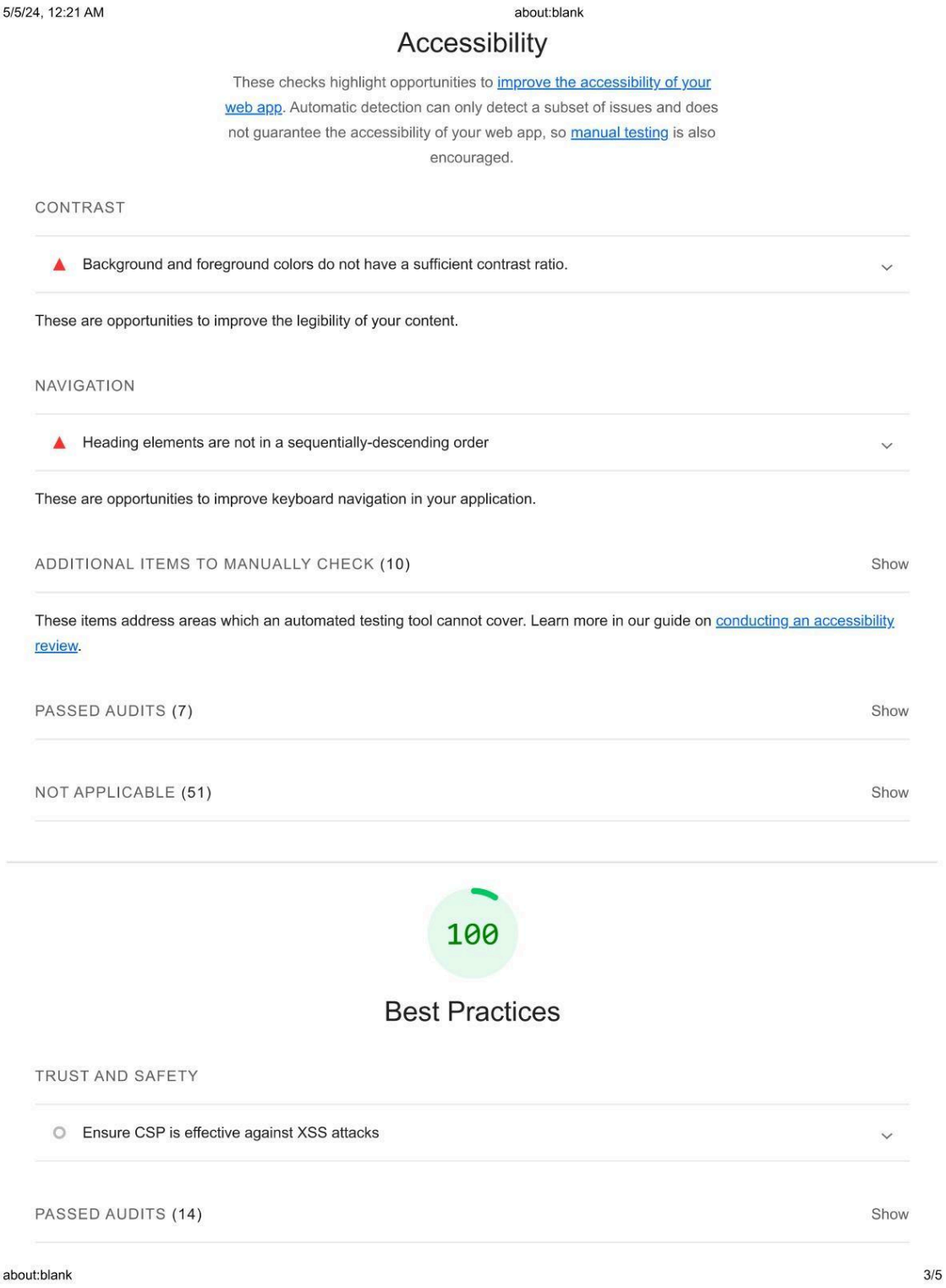
DIAGNOSTICS

- ▲ Reduce unused JavaScript — Potential savings of 28 KiB
- Minify JavaScript — Potential savings of 95 KiB
- Serve static assets with an efficient cache policy — 3 resources found
- Avoid serving legacy JavaScript to modern browsers — Potential savings of 0 KiB
- ○ Initial server response time was short — Root document took 60 ms
- ○ Avoids enormous network payloads — Total size was 77 KiB
- ○ Avoids an excessive DOM size — 195 elements
- ○ Avoid chaining critical requests — 3 chains found
- ○ JavaScript execution time — 0.1 s
- ○ Minimizes main-thread work — 0.3 s
- ○ Largest Contentful Paint element — 2,100 ms

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS (28) Show

85



5/5/24, 12:21 AM about:blank

Accessibility

These checks highlight opportunities to improve the accessibility of your web app. Automatic detection can only detect a subset of issues and does not guarantee the accessibility of your web app, so manual testing is also encouraged.

CONTRAST

▲ Background and foreground colors do not have a sufficient contrast ratio.

These are opportunities to improve the legibility of your content.

NAVIGATION

▲ Heading elements are not in a sequentially-descending order

These are opportunities to improve keyboard navigation in your application.

ADDITIONAL ITEMS TO MANUALLY CHECK (10) Show

These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review.

PASSED AUDITS (7) Show

NOT APPLICABLE (51) Show

100

Best Practices

TRUST AND SAFETY

○ Ensure CSP is effective against XSS attacks

PASSED AUDITS (14) Show

about:blank 3/5


5/5/24, 12:21 AM about:blank
NOT APPLICABLE (2) Show
100
SEO
These checks ensure that your page is following basic search engine optimization advice. There are many additional factors Lighthouse does not score here that may affect your search ranking, including performance on Core Web Vitals. Learn more about Google Search Essentials.
ADDITIONAL ITEMS TO MANUALLY CHECK (1) Show
Run these additional validators on your site to check additional SEO best practices.
PASSED AUDITS (12) Show
NOT APPLICABLE (2) Show
As per Chrome's updated Installability Criteria, Lighthouse will be deprecating the PWA category in a future release. Please refer to the updated PWA documentation for future PWA testing.
PWA
These checks validate the aspects of a Progressive Web App. Learn what makes a good Progressive Web App.
INSTALLABLE
Web app manifest and service worker meet the installability requirements
PWA OPTIMIZED
about:blank 4/5

Configured for a custom splash screen

Sets a theme color for the address bar.

Content is sized correctly for the viewport

Has a `<meta name="viewport">` tag with `width` or `initial-scale`

Manifest has a maskable icon

ADDITIONAL ITEMS TO MANUALLY CHECK (3) Show

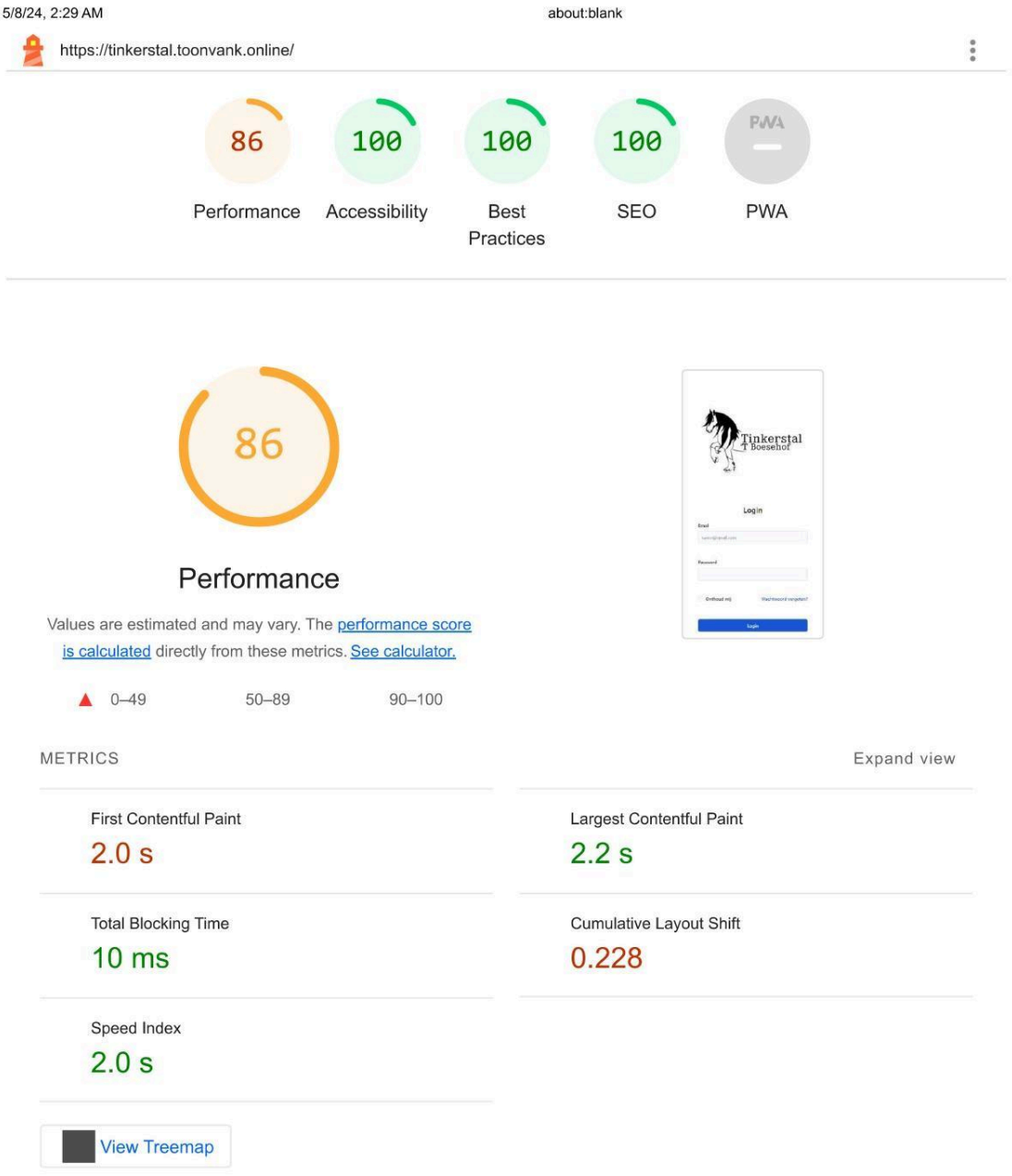
These checks are required by the baseline PWA Checklist but are not automatically checked by Lighthouse. They do not affect your score but it's important that you verify them manually.

- Captured at May 5, 2024, 12:20 AM GMT+2
- Initial page load
- Emulated Moto G Power with Lighthouse 11.6.0
- Slow 4G throttling
- Single page session
- Using Chromium 124.0.0.0 with devtools

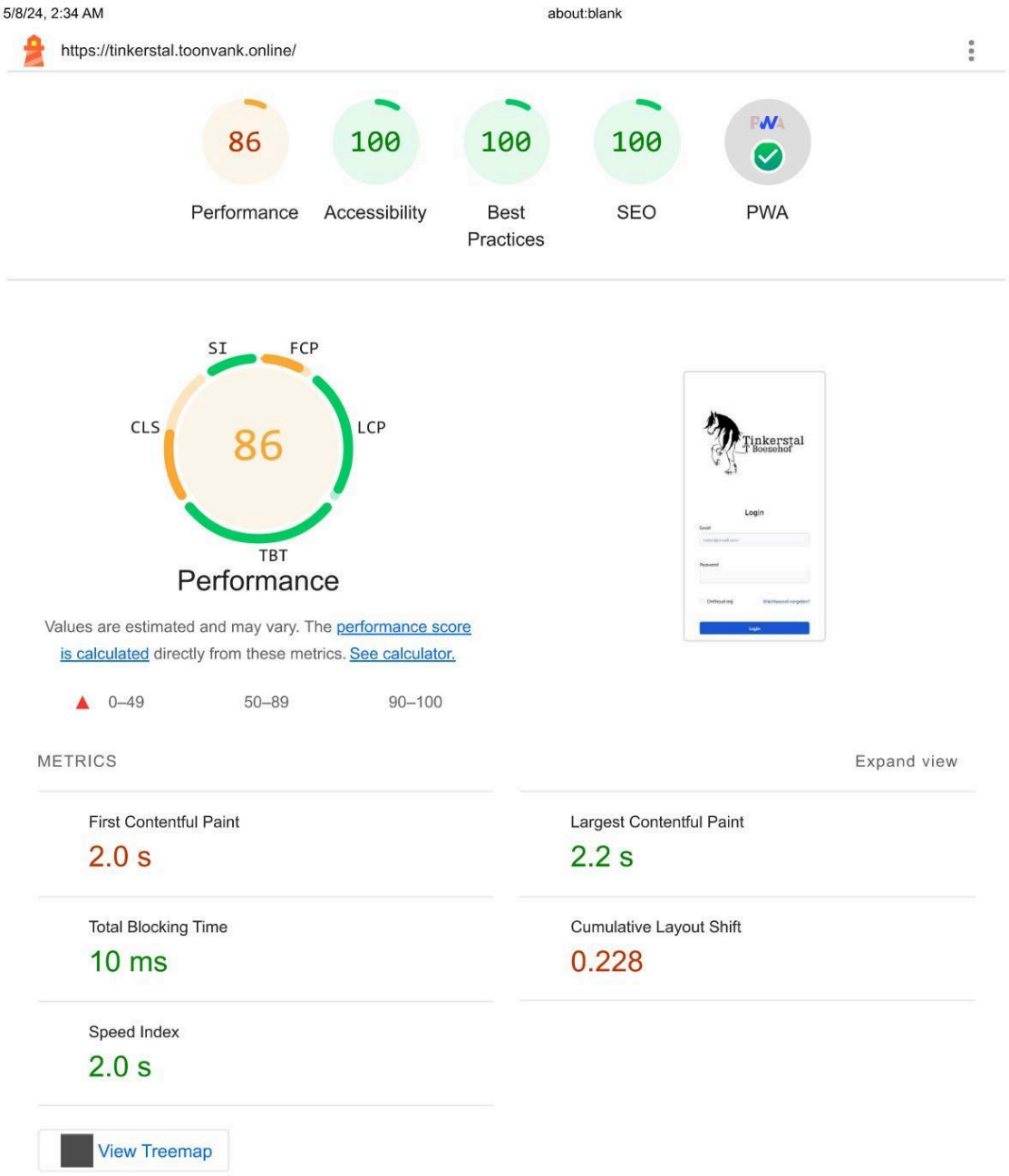
Generated by **Lighthouse** 11.6.0 | File an issue

Research Project Toegepaste Informatica

B. Google Lighthouse Rapport Tinkerstal 'T Boesehof



Geen PWA implementatie



PWA implementatie

Research Project Toegepaste Informatica