Dominique Karreman (student)

Advies rapport techstack

Table of Contents

[Webapplicatie vs Native app 2](#_Toc192368563)

[1. Beveiliging 2](#_Toc192368564)

[Scores 2](#_Toc192368565)

[2. Privacy 2](#_Toc192368566)

[Scores 2](#_Toc192368567)

[3. Prestaties 3](#_Toc192368568)

[Scores 3](#_Toc192368569)

[4. Toegang tot native functies 3](#_Toc192368570)

[Scores 3](#_Toc192368571)

[5. Documentatie & Community 4](#_Toc192368572)

[Scores 4](#_Toc192368573)

[6. Beschikbaarheid van Plug-ins 4](#_Toc192368574)

[Scores 4](#_Toc192368575)

[7. Compatibiliteit 4](#_Toc192368576)

[Scores 5](#_Toc192368577)

[Scores en onderbouwing 5](#_Toc192368578)

[Database 6](#_Toc192368579)

[Backend 6](#_Toc192368580)

[Bronnenlijst 7](#_Toc192368581)

# Webapplicatie vs Native app

## 1. Beveiliging

Native macOS-applicaties die met SwiftUI worden ontwikkeld, profiteren van de sterke beveiligingsmaatregelen die door macOS worden aangeboden. Ze draaien in een sandbox omgeving, waardoor ze beperkte toegang hebben tot systeembronnen en de impact van eventuele beveiligingslekken wordt beperkt. Bovendien moeten deze apps strenge controles doorlopen voordat ze worden goedgekeurd voor distributie via de Apple App Store, wat de kans op schadelijke software sterk reduceert.

• Bron: [Flare Compare - Native vs Web Apps Security](https://flarecompare.com/Cybersecurity/Native%20app%20vs%20web%20app%20%20Which%20one%20is%20more%20secure/)

• Bron: [GeeksforGeeks - Difference between Native and Web Apps](https://www.geeksforgeeks.org/difference-between-native-apps-and-web-apps/)

Webapplicaties daarentegen zijn toegankelijk via webbrowsers, wat betekent dat ze blootgesteld worden aan een meerdere soorten aanvallen zoals cross-site scripting (XSS) en cross-site request forgery (CSRF). Moderne frameworks zoals React, Vue en Angular bevatten wel beveiligingsmechanismen, maar de verantwoordelijkheid blijft grotendeels bij de ontwikkelaar om veilige code te implementeren. Daarnaast zijn webapplicaties afhankelijk van de beveiliging van de server waarop ze worden gehost, wat extra aandacht vereist voor server-side beveiligingspraktijken.

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| Beveiliging | 3 | 2 |

## 2. Privacy

Bij native apps kan de opslag van gegevens vaak lokaal plaatsvinden, wat gebruikers meer controle geeft over hun persoonlijke informatie. Gebruikers hebben de mogelijkheid specifieke toestemmingen te verlenen of te weigeren voor toegang tot bepaalde apparaatfuncties, wat bijdraagt aan een verbeterde privacy. Echter, studies tonen aan dat zowel iOS- als Android-apps soms gebruikmaken van tracking door derden, wat een minpunt kan zijn.

• Bron: [Flare Compare - Privacy in Native Apps](https://flarecompare.com/Cybersecurity/Native%20app%20vs%20web%20app%20%20Which%20one%20is%20more%20secure/)

• Bron: [CitizenSide - Native Apps vs Web Apps](https://citizenside.com/technology/native-apps-vs-web-apps/)

Webapplicaties slaan vaak gegevens op externe servers op, wat kan leiden tot privacy problemen doordat gebruikers minder controle hebben over de verzameling en het gebruik van hun data. Transparantie en naleving van privacywetten (zoals AVG/GDPR) zijn essentieel, maar het ontbreken van lokale opslag betekent dat er altijd een risico is dat gegevens extern worden beheerd.

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| **Privacy** | 2 | 1 |

## 3. Prestaties

SwiftUI-applicaties hebben directe toegang tot systeembronnen op macOS, wat resulteert in uitstekende prestaties en energie-efficiëntie. Ze zijn geoptimaliseerd voor de hardware en software van Apple-apparaten, wat een soepele en responsieve gebruikerservaring biedt, vooral bij grafisch intensieve taken of zware berekeningen.

• Bron: [arXiv - Performance Analysis of Native vs Web Apps](https://arxiv.org/abs/2308.16734)

Webapplicaties daarentegen draaien in een browser, wat vaak leidt tot hogere CPU- en geheugengebruik. Dit kan resulteren in een lagere algehele performance, vooral op apparaten met beperkte hardware of bij intensieve taken. Hoewel moderne JavaScript-frameworks de prestaties verbeteren, blijven ze meestal achter bij native apps.

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| **Prestaties** | 3 | 1 |

## 4. Toegang tot native functies

Native apps gebouwd met SwiftUI hebben volledige toegang tot alle macOS-specifieke functies en hardwarecomponenten. Dit maakt het mogelijk om applicaties te ontwikkelen die diep geïntegreerd zijn met het besturingssysteem, zoals gebruikmaken van geavanceerde grafische rendering, toegang tot systeem-API’s en integratie met andere native applicaties.

• Bron: [Coffee IT - Verschil tussen Hybride, Native en Web Apps](https://coffeeit.nl/blog/verschil-tussen-hybride-native-en-web-apps)

Webapplicaties hebben daarentegen beperkte toegang tot de hardware. Hoewel technologieën zoals Progressive Web Apps (PWA’s) enige toegang bieden tot functies zoals offline opslag en pushmeldingen, blijft de mogelijkheid om diep in het besturingssysteem te integreren beperkt in vergelijking met native apps.

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| Toegang tot native functies | 3 | 1 |

## 5. Documentatie & Community

Apple biedt uitgebreide en kwalitatief sterke documentatie voor SwiftUI, wat het leren en implementeren van native applicaties versimpeld. De community voor SwiftUI is misschien kleiner dan die voor populaire webframeworks, maar de officiële bronnen en ondersteuning zijn van hoge kwaliteit.

• Bron: [Apple Developer Documentation](https://developer.apple.com/documentation/swiftui/)

Webapplicaties profiteren van zeer grote en actieve communities. Frameworks zoals React, Angular en Vue beschikken over een overvloed aan tutorials, plug-ins en externe bibliotheken. Deze uitgebreide documentatie en de actieve community ondersteunen snelle probleemoplossing en continue verbetering van de frameworks.

• Bron: [Supablog - Vergelijking React, Angular en Vue](https://www.supablog.nl/articles/vergelijking-react-angular-vue?utm_source=chatgpt.com)

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| **Documentatie & community** | 2 | 3 |

## 6. Beschikbaarheid van Plug-ins

Voor SwiftUI is het aanbod van plug-ins en bibliotheken aan het groeien, maar blijft het aanbod relatief beperkt in vergelijking met webframeworks. Dit betekent dat ontwikkelaars soms zelf oplossingen moeten ontwikkelen voor specifieke functionaliteiten die niet standaard beschikbaar zijn.

• Bron: [Supablog - Vergelijking React, Angular en Vue](https://www.supablog.nl/articles/vergelijking-react-angular-vue?utm_source=chatgpt.com)

Webapplicaties hebben een enorm aanbod aan plug-ins en bibliotheken, vooral via platforms zoals NPM. Dit maakt het mogelijk om bestaande oplossingen gemakkelijk te integreren en zo de ontwikkeltijd te verkorten. De grote variëteit aan beschikbare plug-ins dekt vrijwel elk aspect van de ontwikkeling, van UI-componenten tot complexe state management oplossingen.

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| **Beschikbaarheid van Plug-ins** | 2 | 3 |

## 7. Compatibiliteit

SwiftUI is specifiek ontworpen voor het Apple-ecosysteem en draait native op macOS, iOS, en andere Apple-platforms. Dit betekent dat er aparte ontwikkeling nodig is voor andere platforms zoals Windows of Android, wat de ontwikkelingskosten kan verhogen als multi-platform ondersteuning vereist is.

• Bron: [Frankwatching - Webapp vs. Native App vs. PWA](https://www.frankwatching.com/archive/2020/12/14/webapp-native-app-progressive-web-app/)

Webapplicaties daarentegen zijn platformonafhankelijk. Ze draaien in elke moderne webbrowser, waardoor ze toegankelijk zijn op vrijwel elk apparaat en besturingssysteem. Dit maakt webapplicaties zeer schaalbaar en breed inzetbaar, wat vooral voordelig is als een applicatie een groot, divers publiek moet bereiken.

### Scores

|  |  |  |
| --- | --- | --- |
|  | Swift (Native) | Webapplicatie |
| **Compatibiliteit** | 1 | 3 |

# Scores en onderbouwing

Niet elke categorie weegt even zwaar mee. Er zijn verschillende dingen die ik erg belangrijk vind tijdens het programmeren maar ook voor het product als eindresultaat. Die dingen wegen zwaarder mee dan categorieën die ik minder belangrijk vind. Daarom geef ik elke categorie zijn eigen multiplier. De gegeven score van de categorie wordt vermenigvuldigd met de multiplier. Alle scores bij elkaar opgeteld geeft dus een goede indicatie voor welke route we zullen nemen voor de tech stack.

1. Beveiliging – multiplier: x10
   1. Ik vind beveiliging een belangrijk onderwerp wanneer het komt tot AI en persoonlijke data. De gesprekken tussen de AI en de gebruiker moeten ten alle tijden veilig zijn en mogen dus niet gelekt worden.
2. Privacy – multiplier: x10
   1. Net zoals beveiliging speelt privacy een grootte rol. We willen niet dat de gebruikers gesprekken met de AI op straat komen te liggen want er kan sensitieve data in voorkomen.
3. Prestaties – multiplier: x8
   1. Met een systeem zoals deze is performance ook erg belangrijk. Je wilt niet minuten moeten wachten op antwoorden of acties. Het is erg fijn als je direct acties ziet.
4. Toegang tot native features: x8
   1. De nature van deze applicatie richt zich ook op een aantal native features zoals het kunnen sorteren en aanmaken van bestanden. Hoewel niet al deze features exclusief aan een native app zijn. Zijn ze wel vaak efficiënter en simpeler.
5. Documentatie & community: x5
   1. Tijdens het ontwikkelen is het erg fijn als er een grote community is van de stack die je gebruikt zodat er genoeg materiaal online te vinden is over errors en andere zaken. Ook is officiële documentatie heel erg belangrijk voor het leren van de taal en features.
6. Plugins – multiplier: x5
   1. Plugins kunnen het leven veel makkelijker maken en geven je meer mogelijkheden tijdens het developen. Ze zijn dus ook een belangrijk onderdeel van het process
7. Compatibiliteit – multiplier: 3x
   1. Omdat dit product gericht is op eigenlijk het verbeteren van de interactie tussen mensen en operating systems is er gelijk al een splitsing tussen de operating systems. Het is mogelijk om een tool te maken die voor alle operating systems werkt, maar de vraag is dan. Werkt hij optimaal en of zijn er veel features of word je tegen gehouden? Ik vind het dus niet heel belangrijk dat IEDEREEN mee kan genieten van een basic product, ik heb veel liever dat een bepaalde doelgroep (in dit geval mac gebruikers) een revolutionaire ervaring hebben met het product. En als er genoeg positieve reacties zijn kan ik wellicht in de toekomst (na dit project) uitbreiden naar windows of linux.

De totale score wordt berekend door elke categorie te vermenigvuldigen met hun multiplier. Als we dat doen komen we voor Swift op een totale score van: 121. Voor de webapplicaties komt dat neer op een totale score van: 91.

Deze vergelijking laat zien dat voor het doeleinde dat ik wil realiseren met dit product, webapplicaties niet de beste vorm zijn.

# Database

Voor het kiezen van een database heb je meerdere keuzes, dingen zoals Postgress, mysql, mariadb, nosql databases. Maar ook voor services zoals apple’s cloudkit.

Aangezien we erachter zijn gekomen dat een native applicatie met Swift beter zal passen bij dit programma dacht ik dat het misschien verstandig zou zijn dat we alle informatie opslaan in cloudkit. Cloudkit is een service die apple aanbiedt om data van applicaties op te slaan in de cloud en bied verder ook de mogelijkheid om deze data te synchroniseren tussen devices met hetzelfde iCloud account.

Het voordeel is dat cloudkit heel goed geïntegreerd is binnen swift en dat je dat dus niet allemaal hoeft te regelen in de AOI laag. Ook is Cloudkit heel erg schaalbaar, het kan miljoenen gebruikers ondersteunen zonder dat je enkele aanpassingen hoeft te maken in de implementatie van de code.

Een nadeel aan cloudkit is dat het onderdeel is van de apple developer program. Het betaalde programma van apple wat je nodig hebt om het volle potentieel van swift te kunnen unlocken.

# Backend

Nu moet ik alleen nog erachter komen welke taal ik wil gebruiken voor het AI-gedeelte van de backend. Als men het heeft over AI is de taal waar in het gedaan wordt meestal in python. En dat komt omdat er veel frameworks en bestaande libraries zijn die ondersteund worden in python.

Ik heb toevallig ook al ervaring in het programmeren van AI-producten in python. Een van de meest populaire libraries is Ollama. Deze library wordt gebruikt om open source large language models in te kunnen laden en lokaal te kunnen runnen.

Er is veel documentatie te vinden over deze modellen en over hoe je het gebruikt met ollama in python.

De bedoeling is dat er een python API komt die de applicatie aanstuurt met AI-antwoorden afkomstig uit de Ollama modellen. Deze API laag ben ik van plan om te doen met Flask API. Omdat er veel plug-ins zijn voor het Flask framework en omdat ik jaren ervaring heb met deze techniek.

# Bronnenlijst

1. Native app vs web app - Which one is more secure? (z.d.). Flare Compare. <https://flarecompare.com/Cybersecurity/Native%20app%20vs%20web%20app%20%20Which%20one%20is%20more%20secure/>
2. GeeksforGeeks. (2023, 14 februari). Difference between Native Apps and Web Apps. GeeksforGeeks. <https://www.geeksforgeeks.org/difference-between-native-apps-and-web-apps/>
3. Native Apps vs. Web Apps | CitizenSide. (2023, 10 september). CitizenSide. <https://citizenside.com/technology/native-apps-vs-web-apps/>
4. Horn, R., Lahnaoui, A., Reinoso, E., Peng, S., Isakov, V., Islam, T., & Malavolta, I. (2023). Native vs Web Apps: Comparing the Energy Consumption and Performance of Android Apps and their Web Counterparts (pp. 44–54). <https://doi.org/10.1109/mobilsoft59058.2023.00013>
5. *Wat is het verschil tussen hybride, native en web apps? | Blog | Coffee IT*. (z.d.). <https://coffeeit.nl/blog/verschil-tussen-hybride-native-en-web-apps>
6. SwiftUI | Apple Developer Documentation. (z.d.). Apple Developer Documentation. <https://developer.apple.com/documentation/swiftui/>
7. Supablog. (2024b, juli 26). Het Juiste Framework Kiezen: Vergelijking van React, Angular en Vue. Het Juiste Framework Kiezen: Vergelijking van React, Angular en Vue. <https://www.supablog.nl/articles/vergelijking-react-angular-vue>
8. Nooijen, L. (2020, 14 december). Een webapp, native app of progressive web app: wat kies je? Frankwatching. <https://www.frankwatching.com/archive/2020/12/14/webapp-native-app-progressive-web-app/>