**ThijmenOS architectuur**

Project: ThijmenOS

Teamleden: Thijmen Brand (480490)

Opdrachtgever: Thijmen Brand

Versie: 1.0

Versiedatum: 09-12-2022

Inhoud

[1. Inleiding 3](#_Toc121490686)

[2. Architectonische doelen 4](#_Toc121490687)

[3. Veiligheid 4](#_Toc121490688)

[2. Systeem context 5](#_Toc121490689)

[3. Containers en technologie keuze 6](#_Toc121490690)

[4. Componenten 7](#_Toc121490691)

[4.1 ThijmenOS – server 7](#_Toc121490692)

[4.2 ThijmenOS – Client 8](#_Toc121490693)

# 1. Inleiding

JavascriptOS is een namaak operating system wat draait in de browser. Met dit project probeer ik meer te weten te komen van operating systemen, Maar ook een diepere kennis te ontwikkelen van typescript en javascript.

Het doel is om een compleet werkend operating system te schrijven met features zoals een kernel, applicaties, notificaties etc.

In dit document zal ik het ontwerp proces van deze applicatie toelichten.

# 2. Architectonische doelen

Er bestaan binnen ThijmenOS een aantal (non-) functionele requirements welke de architectuur hebben beïnvloed

A1: ThijmenOS moet bij de gebruiker op de PC kunnen draaien

A44 t/m A46: Gebruikers moeten bestanden kunnen maken en manipuleren

R5 & Q8: Applicaties mogen enken via voor gedefinieerde methoden met het OS communiceren

# 3. Veiligheid

In de architectuur zijn twee significante componenten van veiligheid ingebouwd. Zoals requirement R5 beschrijft dient een applicatie enkel interactie met het OS te kunnen hebben door een voor gedefinieerde set aan methoden.

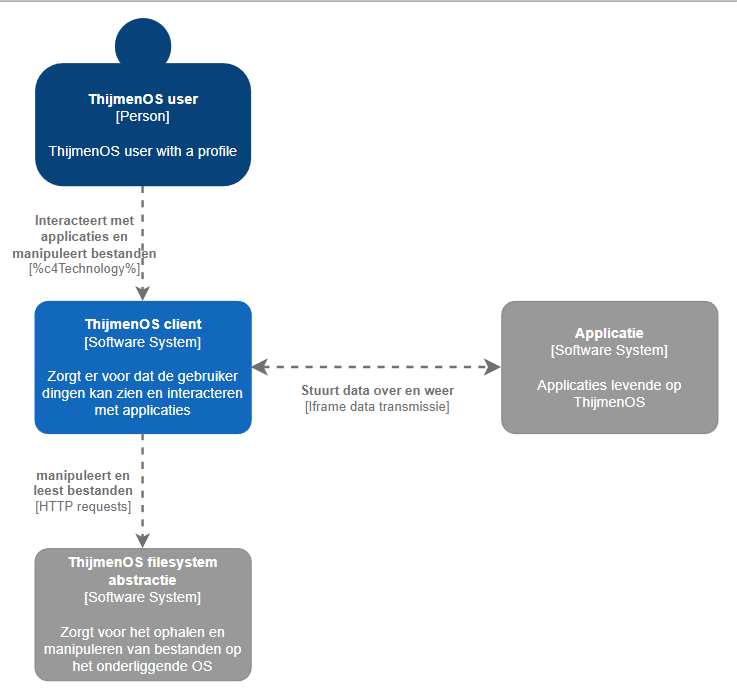
Op deze manier kan ieder stuk data waar een applicatie toegang tot krijg geverifieerd worden. Er kan gekeken worden naar de integriteit van de data en of de applicatie deze data überhaupt mag inzien of gebruiken. Dit sluit aan bij requirement A27 t/m A32 welke zeggen dat applicaties voor verschillende bevoegdheden toestemming dienen te vragen. Wanneer deze toestemming (nog) niet is verleent besluit het systeem dat de applicatie geen toegang heeft tot deze data.

Een ander belangrijk aspect, wat aansluit bij requirement R6 en welke zegt dat applicaties geschreven moeten kunnen worden in Html, Css en Javascript, is de mogelijkheid om systeem code te manipuleren met applicatie javascript code. Wanneer er bijvoorbeeld een applicatie direct in het OS word gedraaid kan de applicatie code botsen met de OS code of OS code kan aangepast en gemanipuleerd worden. De methode beschreven hierboven samen met de Iframe virtualisatie helpen bij het oplossen van dit probleem.

Het tweede veiligheidsaspect wat toegepast is binnen het ontwerp is gebruikers validatie. Er kunnen namelijk meerdere gebruikers worden toegevoegd aan het OS. Om deze gebruikers hun bestanden strikt gescheiden te houden zal er voor ieder server verzoek gevalideerd worden welke gebruiker er is ingelogd en tot welke bestanden deze gebruiker behoord.

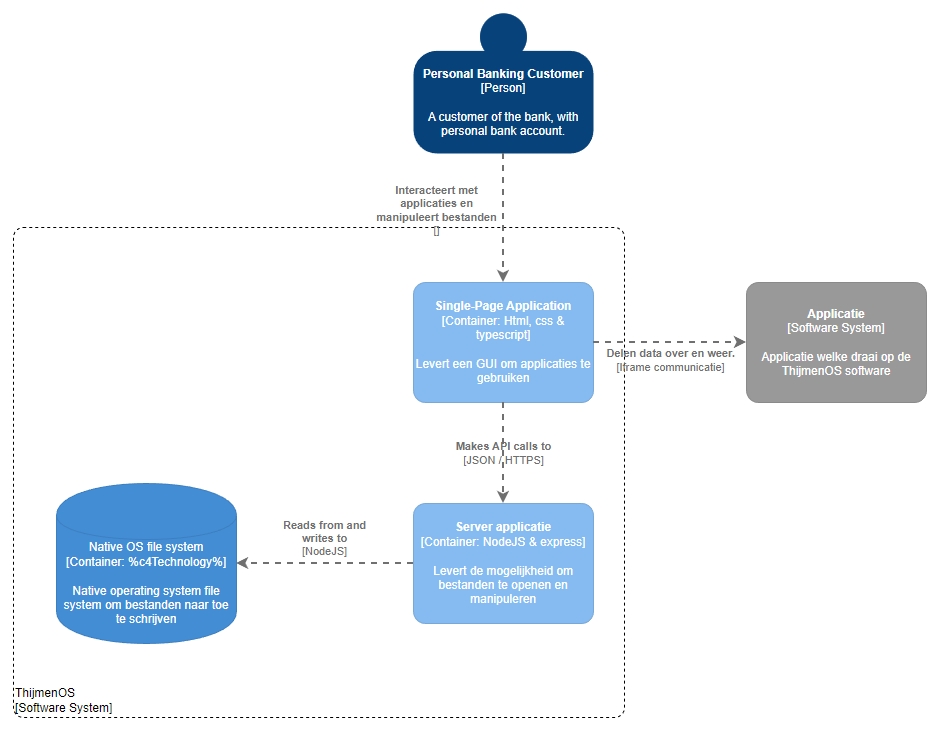
# 4. Systeem context

De systeem context van ThijmenOS is relatief simpel. Er zijn drie grote factoren. Ten eerste natuurlijk de gebruiker welke interactie heeft met de grafische interface van het operating system. Daarnaast zijn er applicaties die getoond kunnen worden. Maar deze applicatie kunnen ook weer acties uitvoeren op het operating system zoals setting veranderen en gebruik maken van de filesystem. En ten slotte dus het filesystem die kan lezen of schrijven naar bestanden.



# 5. Containers en technologie keuze

De huidige systeem architectuur is opgebouwd uit drie componenten. De client, de server en common. De client zorgt voor alles omtrent applicaties. Het visualiseren ervan, maar ook de processen daaromheen. De server zorgt vooral voor alles wat te maken heeft met het filesystem. Wanneer er bijvoorbeeld een bestand gelezen moet worden word er een API request naar de server gestuurd welke het bestand dan uitleest en naar de front-end stuurt. En slotte is er een common component welke gedeelde interfaces en variables beschikbaar stelt. Denk dan bijvoorbeeld aan een settings interface of configuratie opties.

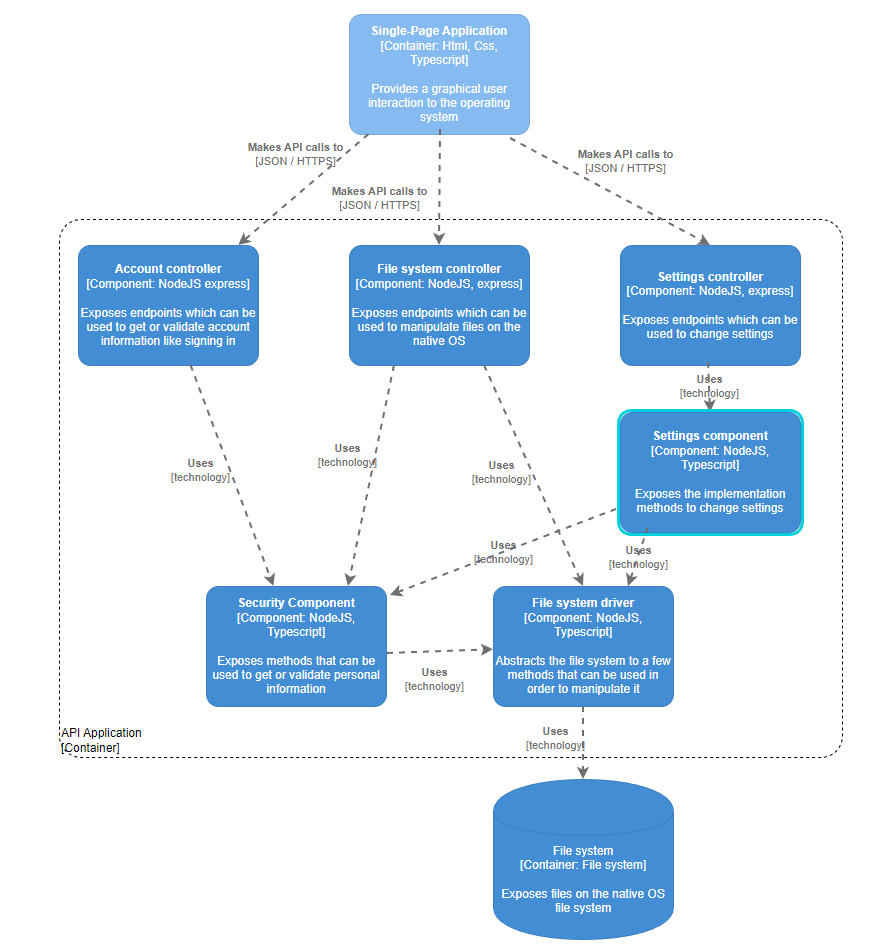


Mijn doel is om zoveel mogelijk client side te doen. Omdat je dan het gevoel krijgt dat het echt een operating system is wat op een fysiek stuk hardware draait. Maar omdat je natuurlijk geen bestanden kan gebruiken en opslaan aan de client side maak ik gebruik van een server. De server dient dan ook vooral om data wat langer bewaard moet worden op te slaan. Zoals bestanden of instellingen.

# 6. Componenten

## 6.1 ThijmenOS – server

De server applicatie van ThijmenOS bestaat uit een NodejS express applicatie welke verschillende endpoints heeft. Deze endpoints zijn onder te verdelen in Account end points, Zoals het inloggen of het veranderen van account data. File system endpoints; om bijvoorbeeld bestanden te lezen en manipuleren. En settings endpoints om instelling te veranderen. Deze endpoints hebben vervolgens onderliggende business layers om de implementatie te regelen.



## 6.2 ThijmenOS – Client

Voor de client zijn er momenteel twee belangrijke interactie punten voor de gebruiker. Hij kan interacteren met een bestands icoon en met de applicatie welke geopend word na het klikken van dat bestands icoon. Vervolgens kan een applicatie communiceren met het file system door gebruik te maken van Iframe communicatie. Deze communicatie word naar geluisterd door een luisteraar welke de verzoeken dan verwerkt en naar de applicatie manager stuurt om naar de desbetreffende applicatie terug te sturen