Research report

well NETWORKED
top CONNECTED

business models • smart services • strategies • solutions

Kremer, Yordi Y.C.T.J.

MABS4.0 | DATUM: 09/01/2024

Table of Contents

Samenvatting

StageBedrijf

Mabs4.0 is een klantgericht en innovatief next gen ICT professional services bedrijf met de mogelijkheden en ambitie om simply the best / world class te zijn/worden.

Opdracht gever

Move4Vitality is een bedrijf actief in Nederland wat mensen, kennis, processen en slimmen systemen samen brengt. Het biedt een platform voor fysiotherapeuten bestaande uit een aantal digitale tools en diensten. Een van de tools is het all-in-one digitaal beweegprogramma op maat voor mensen die in behandeling zijn voor bijvoorbeeld chronische klachten zoals COPD, etalagebenen (claudicatio), artrose of Long COVID. Maar ook voor mensen die hun levensstijl willen veranderen of na een operatie moeten realiseren om weer zo vitaal mogelijk te worden of blijven.

Dit beweegprogramma is onderdeel van het Move4Vitality platform en is gebaseerd op de grondmotrische eigenschappen en belastingvariabelen waarbij patientdata gerelateerd aan het beweegprogramma continue inzichtelijk zijn. Daarbij wordt er voorzien:

1. Een speciale app voor de patiënt
2. Een dashboard voor de fysiotherapeut
3. De techniek om automatisch data te interpreteren en verwerken, in die support voor dagelijkse fysiotherapeutische behandeling en dossiervorming.

Move4Vitality heeft als missie de vitaliteit van de medemens in zijn woon- en werkomgeving continue te verbeteren. Met vitaliteit in de breedste zin van het woord, zowel fysiek als geestelijk. En met expliciet aandacht voor bewegen, eten, slapen, stress en leren.

Move4Vitality ziet het als haar maatschappelijke rol om een gezonde levensstijl voor iedereen toegankelijk te maken. Op deze manier dragen wij bij aan het welzijn van onze samenleving.

De alsmaar stijgende zorgkosten, toename van het aantal mensen met een chronische ziekte, toename van vergrijzing en een groeiend tekort aan zorgprofessionals (zoals fysiotherapeuten), zorgen ervoor dat het voorkomen van gezondheid gerelateerde problemen steeds belangrijker wordt. De Nederlandse Zorgautoriteit heeft niet voor niets als een van haar speerpunten het realiseren van passende zorg - de juiste zorg, op de juiste plek, op het juiste moment. De traditionele fysieke zorg gaat de komende jaren dan ook een drastische verandering doormaken waarbij E-health het vergroten van eigen regie (in lijn met het gedachtengoed van Positieve Gezondheid [Institute for Positive Health, 2011]) een belangrijke rol gaat spelen. Door gebruik te maken van Move4Vitality ben je in staat om meer mensen te helpen en meer aandacht te geven aan mensen die het harder nodig hebben.
De traditionele fysieke zorg gaat de komende jaren dan ook een drastische verandering doormaken. E-health gaat hierbij een grote rol spelen.

Onderzoeksplan

Doel van het project

Momenteel wordt door Move4Vitality alleen gebruik gemaakt van Garmin activiteiten trackers. Data die gegenereerd wordt door iWatch, fitbit, google health en apple health worden nog niet gebruikt. Om zo'n groot mogelijk bereik te creeren is het gewenst om Google Health en Apple Health te koppelen aan het platform. Zodoende is Move4Vitality onafhankelijk van de activiteiten tracker die een deelnemer heeft.

Het gewenste eindresultaat is een Technische realisatie van de koppeling met Apple- en Google Health. De data moet 24/7 opgehaald en gepresenteerd kunnen worden in het platform en de app. Ook moet gekeken worden naar de voorwaarden waarop dit kan/dient te gebeuren.

Hoofdvraag

Hoe kan een integratie van third-party gezonheidsdata op een veilige manier in de huidige bedrijfssoftware gerealiseerd worden?

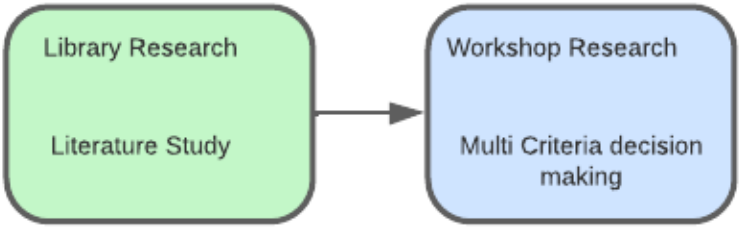
Deelvragen

1. Welke activiteiten apps hebben de meeste integraties van populaire smartwatches?
2. Hoe kan ik de API's van Apple Health en Google Health aanroepen?
3. Welke technische en functionele vereisten zijn nodig voor het ontwikkelen van een systeem dat in staat is om gegevens van externe API's te verzamelen en deze gegevens weer te geven?
4. Hoe moet omgegaan worden (uit technisch en regelgevings oogpunt) met privé/gezondheidsdata?

Onderzoeksstrategie

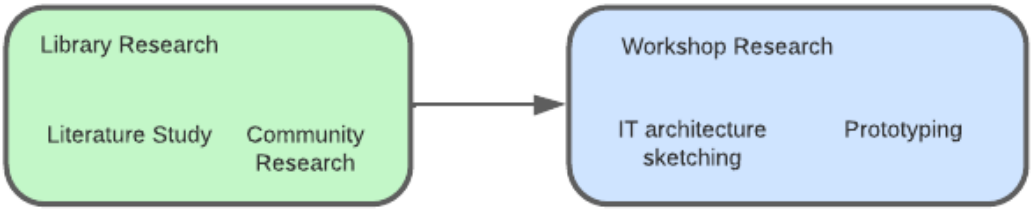
Welke activiteiten apps hebben de meeste integraties van populaire smartwatches?

Ik ga een onderzoek uitvoeren om de meest populaire smartwatches op de markt te identificeren. Vervolgens zal ik deze informatie visualiseren in een diagram en de integraties van deze smartwatches met populaire gezondheidsapps onderzoeken. Mijn doel is om te bepalen welke gezondheidsapps de meeste integraties met deze populaire smartwatches hebben en ze met elkaar te vergelijken.



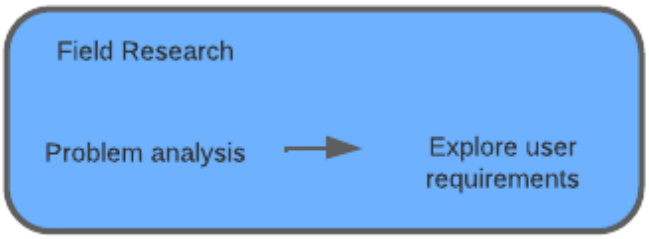
Hoe kan ik de API's van Apple Health en Google Health aanroepen?

Ik dien te onderzoeken hoe ik een externe API kan integreren. Wat zijn de vereiste toestemmingen en welke regelgeving moet mijn app naleven om deze API's te kunnen gebruiken? Daarnaast ben ik van plan een onderzoeksworkshop uit te voeren om te bepalen hoe ik dit in de praktijk kan brengen, door middel van het ontwerpen en maken van prototypes.



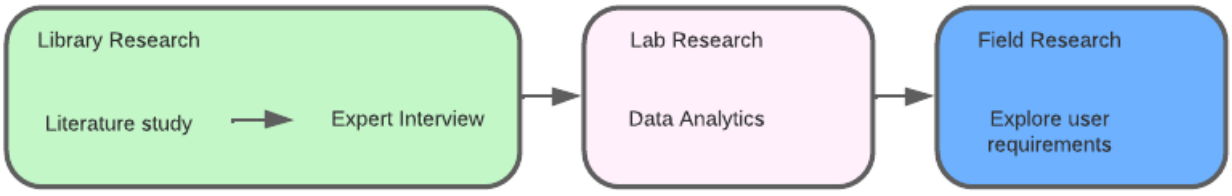
Welke technische en functionele vereisten zijn nodig voor het ontwikkelen van een systeem dat in staat is om gegevens van externe API's te verzamelen en deze gegevens weer te geven?

Om deze onderzoeksvraag te kunnen beantwoorden ga ik een moscow maken met de eisen van het project, ook ga ik een technisch en een functioneel ontwerp opleveren zodat voor mij en de stakeholders duidelijk is waar de prioriteiten liggen en wat ik ga realiseren.



Hoe moet omgegaan worden (uit technisch en regelgevings oogpunt) met privé/gezondheidsdata?

Ik plan hiervoor veldonderzoek uit te voeren en, indien mogelijk, contact op te nemen met fysiotherapeuten om te bepalen welke gegevens van Google Health en Apple Health relevant zijn voor hun praktijk. Welke informatie moet ik presenteren? Daarnast zal ik de de huidige integratie met Garmin activiteitentracking onderzoeken. Bovendien zal ik de documentatie van Google doornemen om te begrijpen aan welke beveiligingseisen mijn app moet voldoen om de data veilig te bewaren.



Welke activiteiten apps hebben de meeste integraties met smartwatches?

De eerste stap was een marktonderzoek naar beschikbare smartwatches. Na het doorzoeken van diverse websites, heb ik 8 smartwatches gevonden die het meeste voorkomen bij bronnen. Deze ga ik vergelijken met activiteiten-apps. Voor dit onderzoek heb ik specifiek de bijbehorende apps van de smartwatches gebruikt en standaard gezondheids-apps op iOS en Android zoals Apple Health, Google fit en Health Connect.

Na raadpleging van meerdere bronnen heb ik een lijst van 13 apps samengesteld. Om inzicht te verkrijgen in de beste integraties met gezondheids-apps voor de Move4vitality app, heb ik een diagram opgesteld waarin ik de 8 smartwatches vergelijk met de 13 gezondheids-apps. Dit onderzoek richt zich op het identificeren van de integraties op Android en iOS die de meest uitgebreide connectiviteit bieden met smartwatches. Het diagram focust uitsluitend op de directe verbindingen met de smartwatch-apps.

Na overleg met Roy, zijn we tot de conclusie gekomen dat Apple Health de meest uitgebreide dekking biedt op iOS, maar helaas geen ondersteuning mogelijk maakt voor Fitbit op iOS. Voor Android was de situatie complexer. Google Fit bood de meest uitgebreide dekking, maar ondersteunde geen Samsung, Fitbit en Garmin. Aangezien Garmin al geïmplementeerd is, vervalt dit nadeel. Bovendien is de Google Fit Android API verouderd en wordt deze naar verwachting eind 2024 stopgezet ondanks dat hebben we besloten Google Fit toch te implementeren om de meeste dekking te krijgen. Google werkt momenteel aan Health Connect op Android, dat tot op heden verbinding biedt met Fitbit en Samsung Health, wat precies ontbreekt in Google Fit.

De voornaamste smartwatches die lastiger zijn voor integratie zijn Fitbit, Samsung Watches en Garmin Watches, omdat ze alleen connectiviteit toestaan met Apple Health of Google Health. Dit zou later mogelijk problemen kunnen veroorzaken. De overige horloges in de vergelijking ondersteunen connectiviteit met zowel Google als Apple Health.

Smart Watch diagram directe koppelingen

Er is een diagram gemaakt om goed in te kunnen zien welke gezondheids apps welke smartwatch integraties ondersteunen.

Watches Apps	FIT BIT	SAMSUNG	APPLE	GARMIN	POLAR	AMAZFIT	FOSSIL	MOBVOI
Apple health (IOS)	X	V	V	V	V	V	V	V
Google fit (BOTH)	X	X	V	X	V	V	V	V
Samsung health (BOTH)	X	V	V	X	X	X	X	X
Health Connect (ANDROID)	V	V	X	X	X	X	X	X
Health Connect : Health tracker (IOS)	X	X	V	X	X	X	X	X
Fit Bit app (BOTH)	V	X	X	X	X	X	X	X
Garmin connect (BOTH)	X	X	V	V	X	X	X	X
Strava (BOTH)	V	V	V	V	V	V	V	V
Polar flow (BOTH)	X	X	V	X	V	X	X	X
Zepp (BOTH)	X	X	V	X	X	V	X	X
Fossil Smartwachtes (BOTH)	X	X	V	X	X	X	V	X
Mobvoi App (BOTH)	X	X	V	X	X	X	X	V
Healthsync (ANDROID)	V	V	X	V	V	X	X	X

Gekozen implimentatie

Voor de iOS-app is momenteel uitsluitend de integratie met Apple Health beschikbaar, met uitzondering van FitBit. De Android-app omvat integraties met Google Fit, Health Connect en Garmin. De Google Fit Android API word verwijderd eind 2024 dus voor deze implementatie is gekozen voor Google Fit Rest API om een uitgebreide dekking te waarborgen, in anticipatie op verdere uitbreiding van de Health Connect-app door Google. Op deze wijze kunnen alle noodzakelijke connecties worden ondersteund.

Watches Apps	FIT BIT	SAMSUNG	APPLE	GARMIN	POLAR	AMAZFIT	FOSSIL	MOBVOI
IOS(Apple health)	X	V	V	V	V	V	V	V
ANDROID(Google Fit/Health Connect/Garmin Connections)	V	V	V	V	V	V	V	V

Hoe kan ik de API'S van Apple Health en Google Health aanroepen?

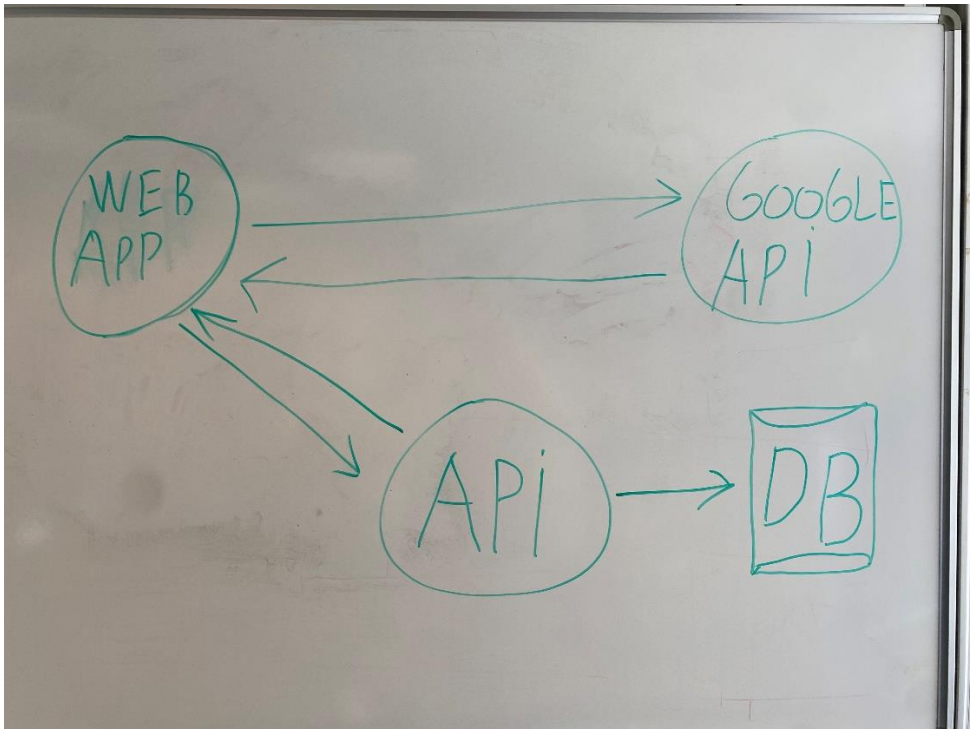
Als eerste ben ik gaan onderzoeken wat apple en google health inhoud, bij apple heb je de gezondheids app en bij google heb je de google fit app. Deze apps houden data bij zoals:

- Activiteiten
- Blood glucose
- Bloeddruk
- Lichaamsafmetingen
- Lichaamstemperatuur
- Hartslag
- Locatie
- Voeding
- Zuurstofverzadiging
- Slaap

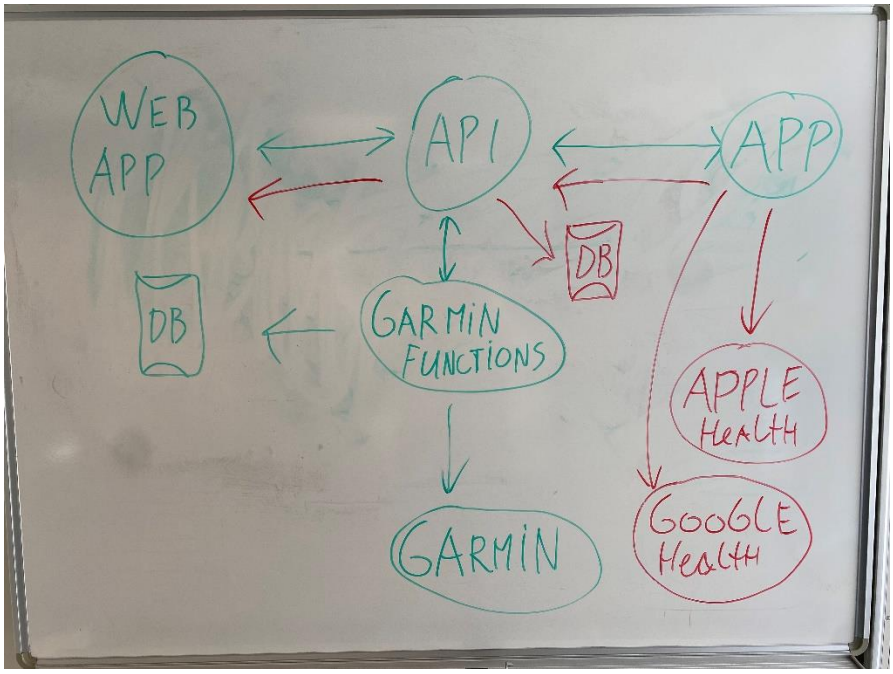
Ik begon met het bestuderen van de documentatie van de Google Fit API en de Apple Health API. Aangezien de integratie van deze API's aanzienlijk verschilt, en omdat Apple doorgaans beperkte toegang verleent tot hun software, was het duidelijk dat de Apple Health-integratie waarschijnlijk uitdagender zou zijn. Ik heb deze kwestie besproken met mijn stagebegeleider en we hebben besloten om ons in eerste instantie te richten op de Google Fit API.

Om een helder beeld te krijgen van de reikwijdte van het project, heb ik schetsen gemaakt op een whiteboard. Dit heeft me geholpen om duidelijkheid te krijgen over de aspecten waaraan ik zal werken, en het stelt mijn stagebegeleider ook in staat te zien dat ik het project begrijp.

Dit is het systeem dat ik zal ontwikkelen in een testomgeving.



Dit is de gewenste uiteindelijke werking van het systeem.



Mijn testomgeving zal dienen als een high-fidelity prototype waarin ik de mogelijkheden van de integratie kan verkennen en testen. Zodra ik tevreden ben met de staat van het prototype, zal ik het implementeren in het daadwerkelijke project.

Ik ben vervolgens de stappen gaan volgen zoals beschreven in de Google Fit Rest API-handleiding. Ik heb een OAuth 2.0-client-ID aangevraagd en een project aangemaakt waarin ik de Fitness API heb ingeschakeld. Tijdens dit proces kwam ik erachter wat 'scopes' zijn, aangezien je bij het aanmaken van je project specifieke toegangsniveaus moet definiëren voor de gegevens die je project van een Google-account wil gebruiken. Ik heb alle Fit-gegevens die ik nodig heb geselecteerd.

Het is belangrijk op te merken dat deze scopes beperkt zijn, wat betekent dat ze toegang verlenen tot gevoelige informatie, en ik niet zomaar toestemming krijg om deze gegevens te gebruiken. Daarom zal ik bezoekers van de webpagina moeten vragen om toegang tot deze gegevens te verlenen. Ik heb besloten om Google Sign-In te gebruiken, aangezien dit automatisch het autorisatieverzoek voor je afhandelt. Daarnaast heb ik de Google Fit-app gedownload om alvast mijn eigen gegevens te verzamelen, zodat ik later kan testen of de gegevens die ik ophaal daadwerkelijk correct zijn.

Als eerste moet je Google sign in installeren in je react native project

```
npm i @react-native-google-signin/google-signin
```

Ook moet je in de bestanden waar je de Google sign in wilt gebruiken het importeren.

```
import { GoogleSignin } from '@react-native-google-signin/google-signin';
```

Dan kun je de package die je geïmporteerd hebt aanroepen en je scopes definiëren.

```
export const googleAuthentication = async () => {
    //set the permissions to authorize access to.
    GoogleSignin.configure({
        scopes: ['https://www.googleapis.com/auth/fitness.activity.read',
'https://www.googleapis.com/auth/fitness.location.read']
      });
      GoogleSignin.signIn().then((result => {
        console.log(result);
      })).catch((error => {
        console.log(error);
      }))
}
```

Op dit moment kan ik succesvol inloggen met mijn Google-account en ontvang ik de profielinformatie. Nu sta ik voor de uitdaging om toegang te krijgen tot gegevens van de beperkte 'scopes'. Na een succesvolle login ontving ik een accesstoken van de ingelogde gebruiker. Nu wil ik graag weten hoe ik deze accesstoken met een POST-verzoek naar de Google Fit API kan sturen, zodat ik de benodigde gegevens kan verkrijgen.

Ik heb geprobeerd informatie hierover te vinden, maar helaas kon ik niet veel relevante documentatie vinden. Daarom heb ik mijn eigen vraag op Stack Overflow geplaatst. Helaas heb ik tot nu toe weinig reacties op mijn vraag ontvangen. Na uitgebreid zoeken ben ik echter gestuit op een vraag van iemand anders waarin nuttige informatie stond. Met deze informatie als leidraad ben ik begonnen met het maken van een GET-request naar de Google API.

'me' in de url verwijst naar de ingelogde gebruiker via Google. Vervolgens geef ik aan dat ik de gegevensbronnen van deze gebruiker wil bekijken, specifiek de geschatte stappentelling, en aan het einde geef ik met behulp van die nummers aan binnen welk tijdsbestek in milliseconden ik deze gegevens wil zien

```
axios.get("https://www.googleapis.com/fitness/v1/users/me/dataSources/derived:com.google.step_count.delta:com.google.android.gms:estimated_steps/datasets/1694124000000000000-1694210400000000000")
```

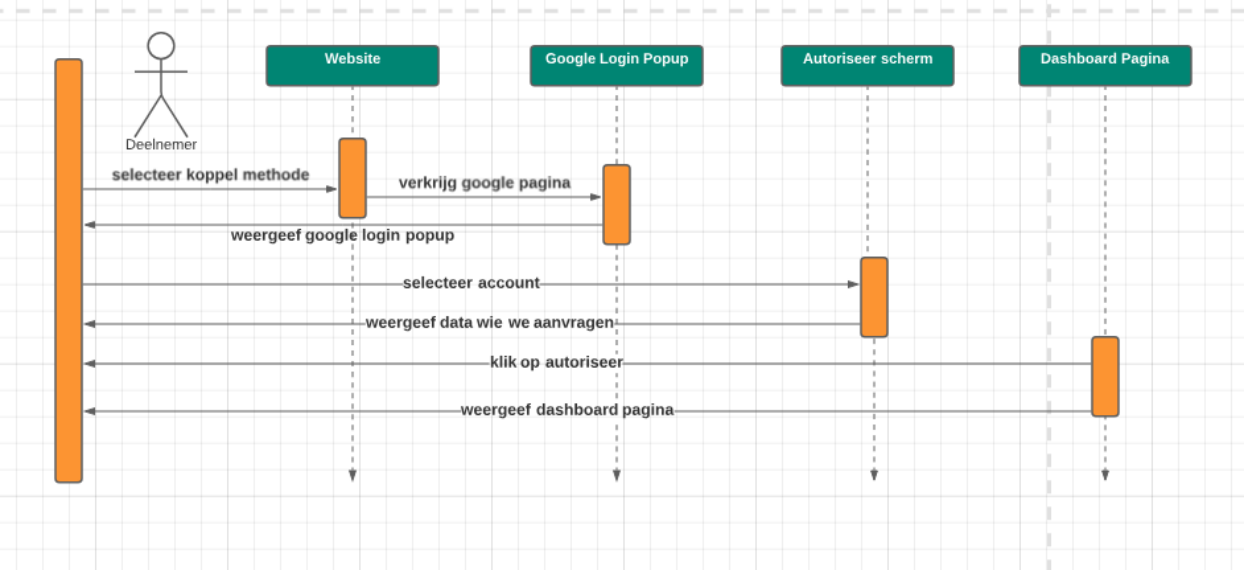
Ik bleef een '401 Unauthorized'-foutmelding ontvangen omdat ik geen token meestuurde. Toen herinnerde ik me dat ik tijdens mijn project in het derde semester van React een JWT-token moest meesturen, meestal via een 'Authorization'-header. Tot mijn verbazing werkte dit ook in dit geval.

```
    function scopesRequest(){
        axios.get("https://www.googleapis.com/fitness/v1/users/me/dataS
ources/derived:com.google.step_count.delta:com.google.android.gms:estim
ated_steps/datasets/1694124000000000000-1694210400000000000", {
            headers: { Authorization: 'Bearer ' + token }
        })
        .then(function (response) {
          setStepRecords(response.data.point)
          calculateDailySteps();
        });
    }
```

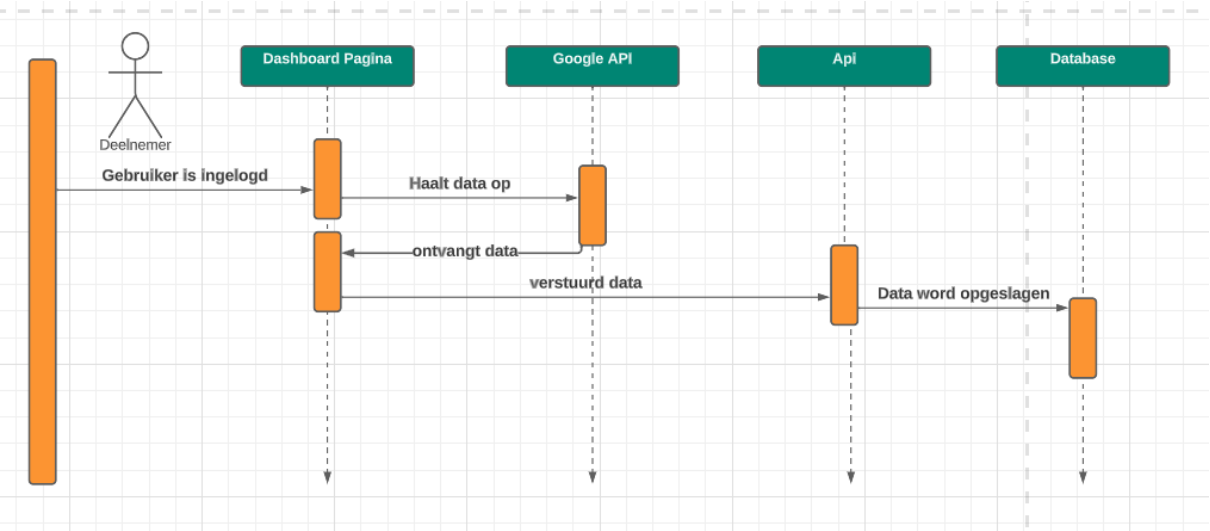
Ik ontving nu succesvol de data van de externe Google Fit API en is het me gelukt om deze correct aan te roepen. Ik heb besloten om deze werking nog te illustreren in een Sequence Diagram voor duidelijkheid.

Sequence Diagram

Het bijgevoegde diagram illustreert de stappen van de Google-verbinding. De gebruiker kiest de koppelingsmethode, waarna de Google-inlogpagina wordt geopend om in te loggen op hun account. Hierop volgend wordt een lijst met gegevens weergegeven die de applicatie wil inzien van de gebruiker. De gebruiker heeft de mogelijkheid om al dan niet toestemming te verlenen voor deze gegevens.



Vervolgens stuurt de applicatie een verzoek naar de Google API, inclusief de authenticatietoken van de ingelogde gebruiker. De data verkregen van de Google API wordt vervolgens doorgestuurd naar onze REST API. Deze gegevens worden verwerkt en opgeslagen om vervolgens te worden weergeven op onze website.

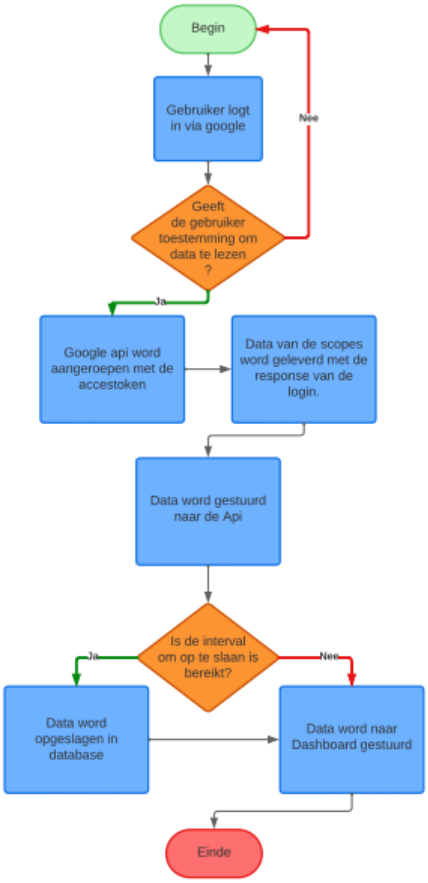


Welke technische en functionele vereisten zijn nodig voor het ontwikkelen van een systeem dat in staat is om gegevens van externe API's te verzamelen en deze gegevens weer te geven?

Ik heb overleg gevoerd met mijn stagebegeleider en collega Ruben, die actief betrokken is geweest bij het Move4Vitality-project. Gedurende dit gesprek hebben zij mij op de hoogte gebracht van de operationele werking van het bestaande systeem en potentiële verbeterpunten voor mijn toegewezen project. Ruben heeft specifiek de technische vereisten van het project doorgenomen, waaronder de benodigde programmeertalen en software.

Functionele vereisten

Om erachter te komen wat een deelnemer in de applicatie moet kunnen doen heb ik een flowchart opgesteld hierdoor is het overzichtelijk hoe de deelnemer door de applicatie heenloopt.



Doormiddel van deze flowchart heb ik een lijst van gebruikersverhalen opgesteld om de taken van elke gebruiker te definiëren. Hierop volgend heb ik Planning Poker toegepast om een schatting te maken van de complexiteit van deze functionaliteiten, zodat ik ze kon integreren in mijn projectplanning. Bovendien heb ik use cases ontwikkeld voor mijn gebruikersverhalen, om beter inzicht te verkrijgen in hoe gebruikers in aanraking komen met deze functionaliteiten.

Moscow Diagram

Onderstaand is het MoSCoW-diagram weergegeven, waarin de functionele eisen van het project zijn geordend op basis van prioriteit. Dit diagram is tot stand gekomen na overleg met mijn bedrijfsbegeleider om de prioriteiten van het project duidelijk te definiëren. Dit helpt mij om te begrijpen waar de focus van mijn werkzaamheden dient te liggen.

Must have	**Should have**	**Could have**	**Won't have**
Als deelnemer moet ik mijn google fit kunnen koppelen aan de app	Als fysiotherapeut wil ik gezondheidsdata kunnen inzien via een dashboard.	Als fysiotherapeut wil ik meerdere grafiek opties bij het weergeven van klant data	Als fysiotherapeut kan ik andere data inzien van Health Connect
Als deelnemer moet ik mijn apple gezondheid kunnen koppelen aan de app op ios			Als fysiotherapeut kan ik andere data inzien van Google Fit
Als deelnemer moet ik mijn health connect kunnen koppelen aan de app op android.			Als fysiotherapeut kan ik andere data inzien van Apple health
Als fysiotherapeut wil ik de dagelijkse stappen kunnen inzien			
Als fysiotherapeut wil ik de afstand afgelegd door een deelnemer kunnen inzien.			
Als deelnemer moet toestemming kunnen geven om mijn gegevens te delen.			

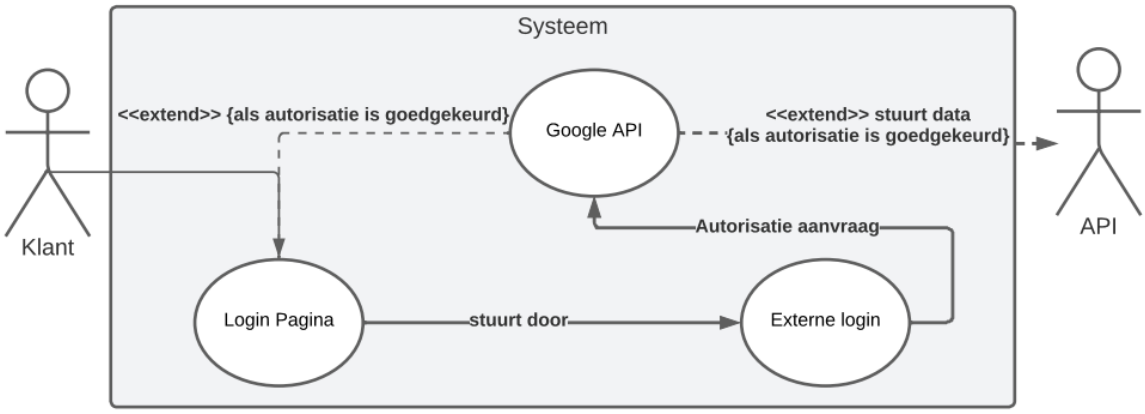
Gebruikersverhalen

Ik heb vanuit de MoSCoW-functionaliteiten gebruikersverhalen afgeleid, waardoor ik mijn sprints efficiënter kan plannen met meer specifieke functionaliteiten. Bovendien heb ik Planning Poker toegepast om een indicatie te verkrijgen van de complexiteit bij deze gebruikersverhalen.

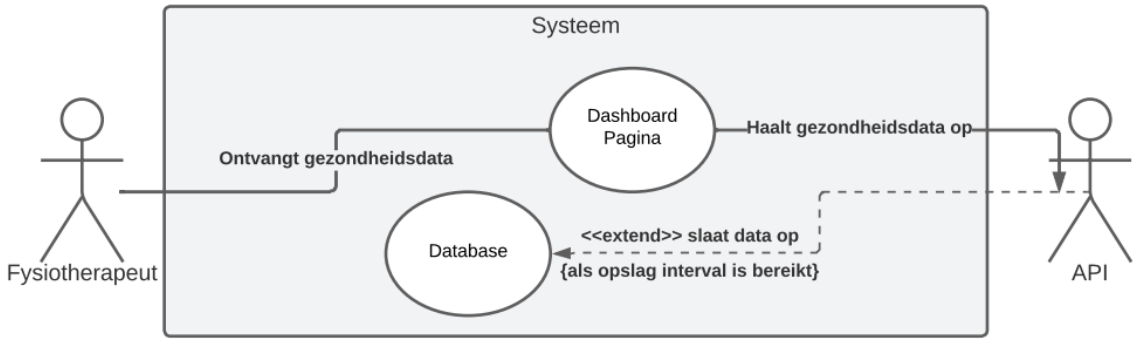
Functionaliteit	**Complexiteit**
Als deelnemer moet ik mijn apple gezondheid kunnen koppelen aan de app op ios	12
Als deelnemer moet ik mijn google fit kunnen koppelen aan de app	12
Als deelnemer moet ik mijn health connect kunnen koppelen aan de app op android.	12
Als deelnemer moet ik toestemming kunnen geven om mijn gegevens te delen.	5
Als fysiotherapeut wil ik gezondheidsdata kunnen inzien via een dashboard.	13
Als fysiotherapeut wil ik dat de dagelijkse stappen verzameld en verwerkt worden.	5
Als fysiotherapeut wil ik dat de afstand afgelegd door een gebruiker verzameld en verwerkt word	5
Als fysiotherapeut wil ik meerdere grafiek opties bij het weergeven van klant data	13
Als fysiotherapeut wil ik dat gegevens van meerdere koppelingen samen te voegen in één overzichtelijk dashboard.	20

Use-Cases

Dit is het Use-Case diagram van de google login zodat de gebruiker toestemming kan geven voor het verlenen van data.

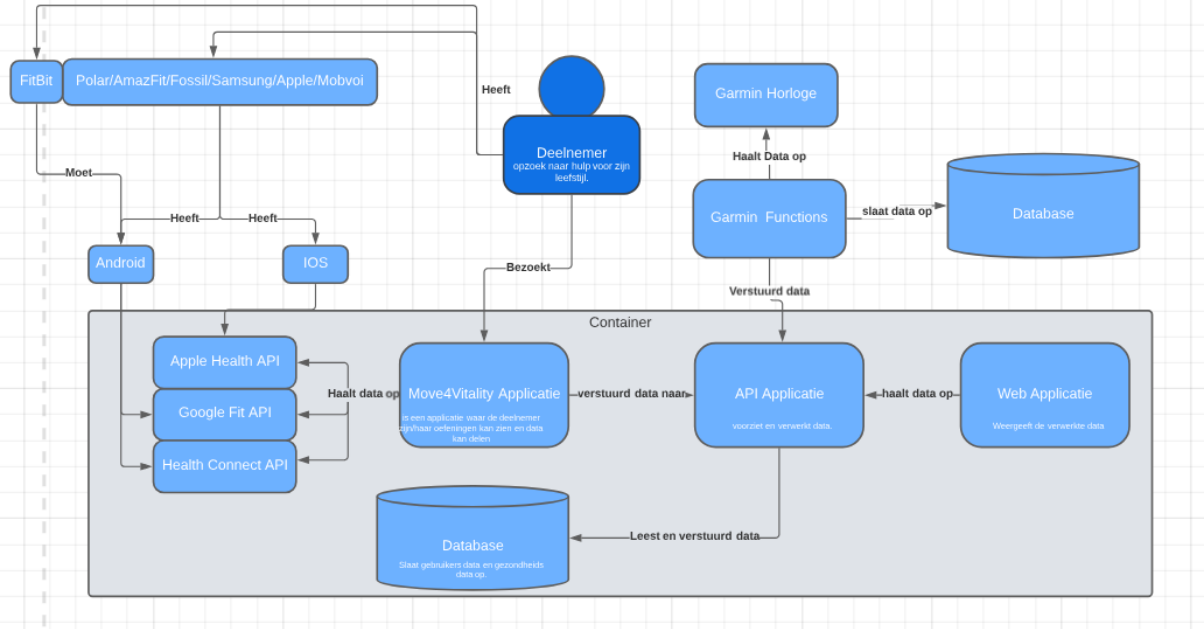


Dit is het Use-Case diagram van hoe gezondheids data verwerkt word en verzonden word naar de frontend voor de fysiotherapeut.



Technische vereisten

Om een volledig beeld te verkrijgen van de technische vereisten van het project, heb ik een C2-diagram opgesteld. Dit diagram bevat de benodigdheden voor het realiseren van het project. Dit overzicht is besproken met mijn bedrijfsbegeleider en Ruben om te verifiëren of dit de gewenste opstelling is voor het project.



React Native App

De applicatie biedt de mogelijkheid voor de deelnemer om in te loggen met hun eigen account. Na inloggen kan de gebruiker de app koppelen aan Apple Health op iOS of aan Health Connect en/of Google Fit op Android. De app haalt gezondheids- en activiteitsgegevens op van deze koppelingen en stuurt deze vervolgens naar mijn API. Ik maak gebruik van Expo Go om de applicatie te testen. Echter, vanwege het ontbreken van ondersteuning voor native code in Expo Go, vereist het bouwen van de applicatie aparte procedures.

Android

Om deze applicatie op android te kunnen bouwen is een Windows omgeving of een Mac omgeving nodig hiervoor gebruik ik mijn persoonlijke laptop. Om deze applicatie te kunnen testen gebruik ik de android telefoon die ik verkregen heb van mijn stagebedrijf.

IOS

Om deze applicatie op IOS te kunnen bouwen is een Mac omgeving nodig hiervoor ben ik met Roy in gesprek gegaan over de opties en er is nog een Mac book beschikbaar van het bedrijf waarbij de accu vervangen moet worden die ik kan gebruiken zodra dit gereed is. Om deze applicatie te kunnen testen gebuik ik mijn eigen iphone.

React Web Applicatie

De webapplicatie is essentieel voor het tonen van activiteits- en gezondheidsgegevens aan fysiotherapeuten. Hier hebben fysiotherapeuten de mogelijkheid om deelnemers te selecteren en relevante gegevens op te vragen. Deze data wordt opgehaald vanuit mijn API.

REST Web API

De API van Move4Vitality is een C# API die werkt volgens Clean Architecture. Deze architectuur volgt het Dependency Inversion Principle en Domain Driven Design. Clean Architecture plaatst de business

logica en het applicatiemodel in het midden van de applicatie. Infrastructuur- en implementatiedetails zijn afhankelijk van de Application Core, bereikt door het benoemen van abstracties en interfaces in de infrastructuurlaag. De structuur van de backend ziet er uit als volgend:

- API bevat de webgerelateerde onderdelen: controllers, web specifieke services, configuratie, enz.
- Applicatie: De applicatielaag bevat de bedrijfslaag hierin staat alle logica.
- Gemeenschappelijk: Bevat hulpprogrammaklassen en -services en is de enige assembly waarnaar verwezen kan worden door alle andere assemblies (en mag geen afhankelijkheid hebben van een andere projectassembly!).
- Domein: Bevat domeinentiteiten (EF POCO's).
- Infrastructuur: Implementaties voor externe afhankelijkheden (Auth0, Garmin, enz.).
- Persistentie: Bevat EF-configuratie en definitie van de DB-context."

deze API ontvangt activiteiten en gezondheids data vanuit mijn React Native App. Deze data word vervolgens verwerkt en opgeslagen in een database.

Databases

NoSQL

Als database voor dit project word gebruik gemaakt van een NoSQL database voor het opslaan van alle Garmin gezondheidsdata die Move4Vitality ontvangt. De reden hiervoor is dat er veel data opgeslagen kan worden, wat je gaat hebben als een activiteitentracker constant data doorstuurt. Voor mijn Proof of concept heb ik alleen een Cosmos database gebruikt omdat Ruben mij dit had aangeraden en hun hadden er nog een beschikbaar.

SQL

Ook word er gebruik gemaakt van een SQL database in het platform dit om het totaal aantal dagelijkse stappen en stappen per activiteit op te slaan. De gegevens in de SQL database zijn cruciaal voor de functionaliteit van het domein. Binnen het domein zijn bepaalde relaties van belang waardoor er gekozen is voor een SQL database.

Hoe moet omgegaan worden (uit technisch en regelgevings oogpunt) met privé/gezondheidsdata?

In eerste instantie heb ik onderzocht naar de wijze waarop ik toegang kan verkrijgen tot de beveiligde gegevens van een gebruiker via een van de externe gezondheids-API's. Dit proces omvat het aanvragen van 'scopes'. Gezondheidsdata van een gebruiker wordt geclassificeerd als 'restricted scope', wat impliceert dat de autorisatie van de gebruiker vereist is om deze data te kunnen raadplegen.

Ik heb een lijst samengesteld van alle beschikbare activiteits- en gezondheidsgegevens die door Apple en Google worden aangeboden. Vervolgens heb ik overleg gepleegd met mijn stagebegeleider om te bepalen welke gegevens daadwerkelijk worden geïmplementeerd in de applicatie, met als doel het aantal benodigde 'scopes' te beperken tot strikt noodzakelijke informatie.

Tevens heb ik contact opgenomen met Ruben om toegang te verkrijgen tot de gegevens van de bestaande koppeling met Garmin. Dit biedt inzicht in het huidige gebruik van gegevens bij het tonen aan fysiotherapeuten en stelt mij in staat te overwegen hoe deze gegevens mogelijk in de toekomst geïntegreerd kunnen worden.

Voor verdere verduidelijking van hun plannen met betrekking tot een dergelijke applicatie zal ik in gesprek gaan met mijn stagebegeleider, Ruben. Dit zal me assisteren bij het bepalen van de optimale wijze waarop ik de gegevens wil verwerken en presenteren aan de fysiotherapeut.

Tijdens een gesprek met Ruben heb ik mijn zorgen geuit over de beveiliging van gezondheidsgegevens, zoals ik had aangegeven tijdens mijn tussentijdse presentatie. Ruben heeft me gerustgesteld door te vertellen dat de databases in Azure zijn versleuteld, waardoor ik me geen zorgen hoef te maken. Ik vroeg ook of het nodig was om gezondheidsdata te versleutelen bij het versturen naar de API, en Ruben verzekerde me dat HTTPS voldoende is om dataverlies te voorkomen

Technische oogpunten

Dit zijn de punten waar je rekening mee moet houden wil je een technisch veilige applicatie hebben voor gezondheidsgegevens.

1. **Encryptie en beveiliging:** Gebruik sterke encryptiemethoden voor zowel data in rust (opgeslagen gegevens) als data in transit (verzonden gegevens). Beveilig de opslag en transmissie van deze gegevens.
2. **Toegangscontrole:** Implementeer strenge controles voor wie toegang heeft tot de gegevens, zowel binnen de app als op de servers. Gebruik methoden zoals authenticatie, autorisatie en logging.
3. **Pseudonimisering of anonimisering:** Overweeg technieken zoals het verwijderen of vervangen van direct identificeerbare informatie om de privacy van individuen te beschermen terwijl de gegevens bruikbaar blijven voor analyse.
4. **Minimalisatie van gegevens:** Verzamel en bewaar alleen de benodigde gegevens. Vermijd het vastleggen van overbodige persoonlijke of gezondheidsgerelateerde informatie.
5. **Beveiligde ontwikkelingspraktijken:** Volg best practices voor software- en app-ontwikkeling om kwetsbaarheden te minimaliseren. Voer regelmatig beveiligingstests en audits uit.

In de React Native applicatie vraag ik de deelnemer voor toestemming voor de data die ik wil lezen van de deelnemer zodra de deelnemer de applicatie opent en op de knop klikt voor zijn koppeling word het autorisatie scherm geopend waar de deelnemer moet selecteren welke data hij/zij wil toestaan om uit te laten lezen.

Dit is waar toestemming word gevraagd aan de deelnemer en waar je toegang kan vragen voor meer data.

```
    await requestPermission([
        {
          accessType: 'read',
          recordType: 'Steps',
        },
        {
          accessType: 'read',
          recordType: 'Distance'
        }
      ]);
```

Vervolgens word op deze manier data opgehaald van de externe API zoals het aantal stappen.

```
 //retrieves steps from health connect using the filter.
          const steps = await readRecords('Steps', { timeRangeFilter });
          const totalSteps = steps.reduce((sum, cur) => sum + cur.count, 0);
          setSteps(totalSteps);
```

Deze word data word vervolgens doorgestuurd naar de API.

```
export const postTrackerData = async (args: postTrackerDataArgs) => {
  const {calendarDate, steps, distanceInMeters} = args;
  //gets jwt token
  const jwt = await getCredentials();
  return DashboardApi.post(`/GoogleHealth/SaveTrackerResults`, {
    calendarDate,
    steps,
    distanceInMeters
  },
  {
    headers: {
        Authorization: `Bearer ${jwt?.accessToken}`
    }
  }).then((result) => {
    console.log(result);
  }).catch((err) => {
    console.log(err);
  });
};
```

Mijn api ontvangt een request met de volgende variabelen.

```
public required string CalendarDate { get; init; }

public required int Steps { get; init; }

public required int DistanceInMeters { get; init; }
```

De controller veranderd dit vervolgens naar een SaveTrackerResultsCommand.

```
[HttpPost("SaveTrackerResults")]
[AuthorizeRole(RoleName.Client)]
public Task SaveTrackerResults(SaveTrackerResultsBody body) =>
    Mediator.Send(
        new SaveTrackerResultsCommand
        {
            CalendarDate = body.CalendarDate,
            Steps = body.Steps,
            DistanceInMeters = body.DistanceInMeters,
        });
```

Vervolgens controleert de Handler van SaveTrackerResultsCommand of de TrackerResult geupdate moet worden of dat er een nieuwe moet worden opgeslagen, en of de deelnemer een abonnement heeft zonder tracker want alleen dan mag de deelnemer gebruik maken van Google Fit of Health Connect.

```
private async Task CreateOrUpdateTrackerDay(SaveTrackerResultsCommand command,
CancellationToken ct)
        {
            var calendarDate = command.CalendarDate.ToCalendarDate();
           var client = await TryGetClientAndSingleTrackerDay(_caller.Id,
      calendarDate, ct);

            if (client.UtcSubscriptionIncludesTrackerSince != null)
             throw new BadRequestException("Client should have subscription
             excluding tracker.");

            var trackerDay = client.TrackerDays.SingleOrDefault();
            if (trackerDay == null)
            {
                trackerDay = new TrackerDay(client, calendarDate);
                client.TrackerDays.Add(trackerDay);
            }

            var utcNow = _dateService.UtcNow;
            trackerDay.UpdateSteps(command.Steps, utcNow);
            trackerDay.UpdateDistance(command.DistanceInMeters, utcNow);
            client.UtcTrackerSynchronised = utcNow;
            await _db.SaveChangesAsync(ct);
        }
```

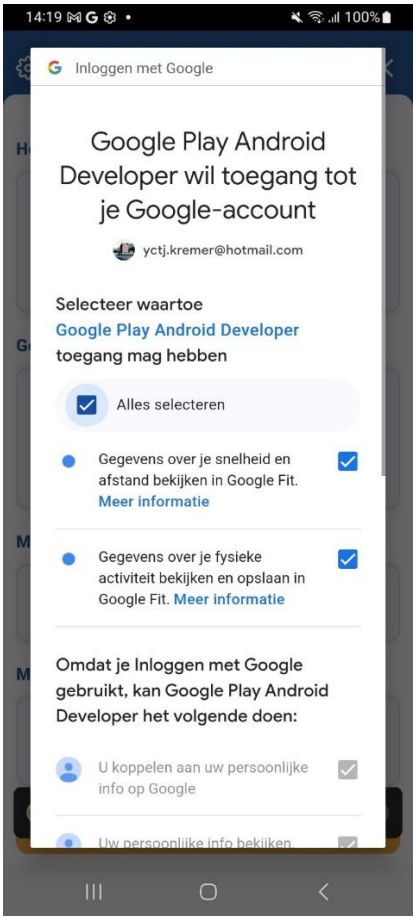
Regelgevings oogpunten

Dit zijn de punten waar je rekening mee moet houden wil je een applicatie hebben die aan alle regelementen voldoet.

1. **Compliance met wetgeving:** Zorg ervoor dat je voldoet aan regelgeving zoals HIPAA, GDPR of andere wetten en voorschriften die van toepassing zijn op de locaties waar je app wordt gebruikt.
2. **Toestemming en transparantie:** Verkrijg expliciete toestemming van gebruikers om hun gegevens te gebruiken. Zorg ervoor dat gebruikers duidelijk geïnformeerd zijn over welke gegevens worden verzameld, hoe ze worden gebruikt en met wie ze worden gedeeld.
3. **Gegevensbewaartermijnen:** Houd rekening met de regelgeving met betrekking tot hoe lang bepaalde gegevens moeten worden bewaard. Minimaliseer de bewaartijd van gegevens wanneer mogelijk.
4. **Meldplicht bij datalekken:** Zorg voor procedures om datalekken te detecteren, melden en aan te pakken volgens de vereisten van relevante regelgeving.
5. **Gegevensportabiliteit en verwijdering:** Bied gebruikers de mogelijkheid om hun gegevens te exporteren en verwijderen, in overeenstemming met de regelgeving zoals het recht om vergeten te worden onder de GDPR.

Health Connect/Google Fit

In de bestaande Move4Vitality app is grotendeels van deze oogpunten al gerealiseerd. Het stukje Toestemming en transparantie heb ik gerealiseerd, ik laat de deelnemer zien welke data data de app wilt lezen van andere apps. Een scherm komt een beeld waar de deelnemer toestemming kan verlenen aan de test app zodat wij data mogen lezen van externe gezondheidsapps.



Conclusie

Move4Vitality dient verdere integraties te implementeren in hun app voor third-party gezondheidsdata, waardoor ze minder afhankelijk worden van de activiteitentracker die een deelnemer gebruikt.

De geplande integraties voor de Move4Vitality app omvatten Health Connect en Google Fit voor Android, evenals Apple Health voor iOS. De bestaande Garmin-integratie in de Move4Vitality app zal behouden blijven, aangezien de integraties Health Connect en Google Fit op Android geen ondersteuning bieden voor Garmin.

Voor het ophalen van gezondheidsdata van third-party apps is het noodzakelijk om een pakket te installeren in de React Native App voor elke integratie. Vervolgens moeten scopes worden ingevoerd om aan te geven welke data gelezen dient te worden van de third-party apps.

Wanneer de deelnemer in de app op de gewenste koppeling klikt, verschijnt er een pop-up scherm met de naam van onze app en de naam van de app waarvan we data willen ophalen. Op dit pop-up scherm worden tevens alle gegevens weergegeven die de app wenst te lezen. Hier kan de deelnemer selecteren welke data hij/zij wilt toestaan om uit te laten lezen.

Na het verlenen van toestemming aan de app kan het geïnstalleerde pakket worden gebruikt om functies aan te roepen, zoals readSteps(). Bij deze functie moet een startdatum en einddatum worden opgegeven, en het resultaat moet worden opgeteld om het totaal te verkrijgen.

De algemene functionaliteiten voor dit project omvatten het koppelen met externe third-party apps zoals Apple Health, Health Connect en Google Fit, het ophalen en verwerken van data, het weergeven van data aan fysiotherapeuten, de mogelijkheid voor deelnemers om toestemming te verlenen voor het lezen van data, en het opslaan van data in de database.

Extra functionaliteiten die kunnen worden toegevoegd, omvatten een dashboard waarmee fysiotherapeuten gegevens in grafieken kunnen bekijken en verschillende weergavevormen voor de data.

De huidige Move4Vitality app voldoet reeds aan veel regelgevingsvereisten. Voor mijn deel van het project heb ik toestemming en transparantie geïmplementeerd, zodat de deelnemer een duidelijk inzicht krijgt in hoe wij omgaan met hun gezondheidsdata en welke gegevens precies worden uitgelezen.

References

ANYTHING INFOTECH PVT LTD. (2023, Maart 15). *beginners-guide-developing-healthcare-app-anything-infotech-pvt-ltd*. Retrieved from linkedin: https://www.linkedin.com/pulse/beginners-guide-developing-healthcare-app-anything-infotech-pvt-ltd/

Apple. (2024). *setting_up_healthkit*. Retrieved from developer.apple: https://developer.apple.com/documentation/healthkit/setting_up_healthkit

CHOKKATTU, J. (2023, October 14). *best-smartwatches*. Retrieved from wired: https://www.wired.com/gallery/best-smartwatches/

Digitale Zorg Gids. (2024). *toelichting-privacy-kenmerken*. Retrieved from digitalezorggids: https://www.digitalezorggids.nl/toelichting-privacy-kenmerken/

Google . (2024). *activity apps*. Retrieved from google play store: https://play.google.com/store/apps/collection/cluster?gsr=SnBqGEM5TnIwV0ZkS1Btd1NBRmM2S3pIdVE9PcICUwojCh9jb20uZ29vZ2xlLmFuZHJvaWQuYXBwcy5maXRuZXNzEAcYCDABOABKJggBGh5Hb29nbGUgRml0OiBBY3Rpdml0ZWl0dHJhY2tpbmcgACgA:S:ANO1ljJe09Q&hl=nl&gl=US

Google . (2024). *developers.google.com*. Retrieved from google-sign-in: https://developers.google.com/identity/#google-sign-in

Google. (2024). *android developer*. Retrieved from support.google: https://support.google.com/googleplay/android-developer/answer/12991134?hl=en#:~:text=Apps%20reading%20and%2For%20writing,only%20with%20the%20user's%20consent.

Google. (2024). *get-started*. Retrieved from developers.google: https://developers.google.com/fit/rest/v1/get-started

HINDY, J. (2023, August 26). *every-major-smartwatch-brand-ranked-worst-best*. Retrieved from slashgear: https://www.slashgear.com/1372604/every-major-smartwatch-brand-ranked-worst-best/

macelai, r. (2023, Juli). *react-native-health?activeTab=readme*. Retrieved from npmjs: https://www.npmjs.com/package/react-native-health?activeTab=readme

matinzd. (2023, September 12). *react-native-health-connect*. Retrieved from github: https://github.com/matinzd/react-native-health-connect

Miller, M. (2023, October 17). *best-smartwatch*. Retrieved from zdnet: https://www.zdnet.com/article/best-smartwatch/

Open AI. (2022). Retrieved from chat.openai: https://chat.openai.com/

Peterse, J. (2022, November 25). *beste-fitness-app*. Retrieved from ID: https://id.nl/huis-en-entertainment/computer-en-gaming/software/beste-fitness-app-check-deze-20-toppers

smartwatches top 10. (2024). Retrieved from coolblue: https://www.coolblue.nl/en/smartwatches/top-10

stasdoskalenko, a. (2023, November 14). *react-native-google-fit*. Retrieved from npmjs: https://www.npmjs.com/package/react-native-google-fit