FACULTEIT INGENIEURSWETENSCHAPPEN
EN ARCHITECTUUR

Sporters Connect

TEAM 2

Hannes Beheydt
Mathis Bossuyt
Gust Debaere
Stan Lein

UNIVERSITEIT
GENT

2023-2024 Industrieel ingenieur informatica

INHOUD

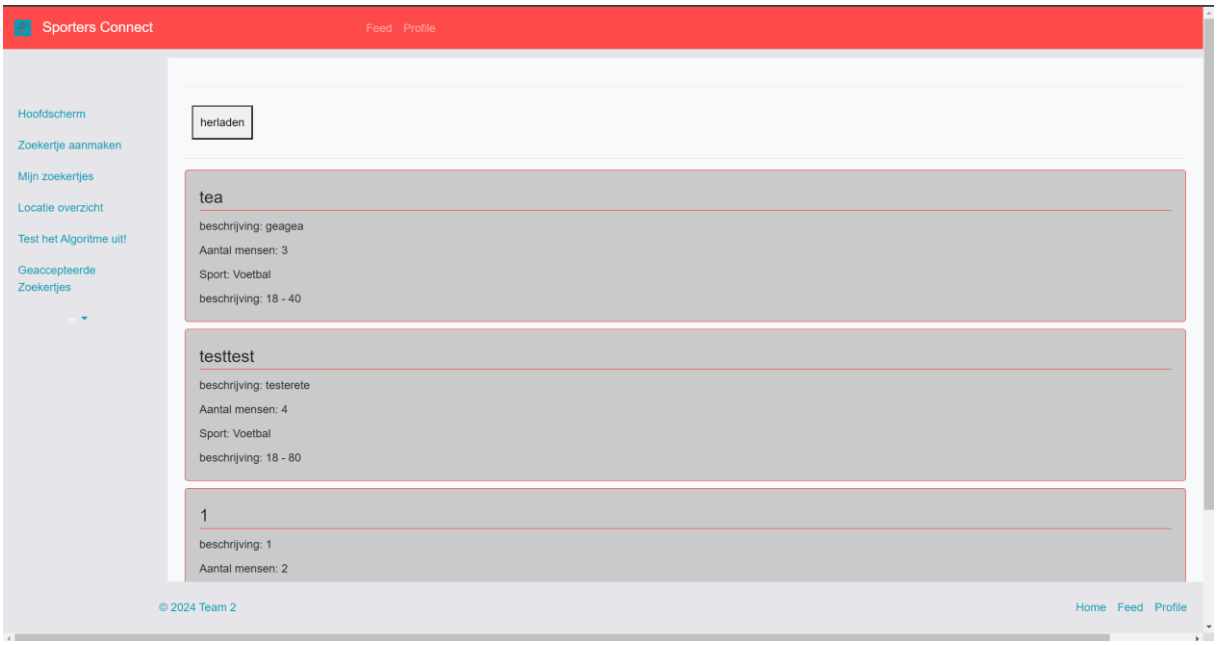
1. INLEIDING WEBSITE

De opdracht vereist het maken van een website die mensen met elkaar verbindt. Bij het creëren van een interactieve website moet worden gekozen tussen het gebruik van JavaScript en HTML of Angular.

De website Sporters Connect richt zich op het bij elkaar brengen van sporters. Om de website te gebruiken, moet een account worden aangemaakt. Bij het aanmaken van een account moet de leeftijd, de sporten die worden beoefend, het niveau en de woonplaats worden opgegeven. Deze informatie wordt gebruikt om het account te koppelen aan relevante zoekertjes. Een zoekertjes wordt aangemaakt door een gebruiker van de site.

Bij het aanmaken van een zoekertje wordt volgende informatie meegegeven welke sport, het aantal personen, leeftijd en locatie. Op deze manier kunnen sporters elkaar vinden gebaseerd op locatie, niveau, soort sport en leeftijd.

In figuur 1 ziet u een screenshot van het hoofdscherm van de website.



Figuur 1: Hoofdscherm website

2. ANALYSE EN ONTWERP

2.1 WEGWIJS IN HET PROJECT

Het project bestaat uit verschillende HTML-, CSS- en JavaScript-bestanden die samen een webapplicatie creëren met functionaliteiten zoals gebruikersprofielbeheer, het creëren en onderhouden van zoekertjes, Google Maps-integratie en algoritmische matching. Deze bestanden zijn ook weergegeven in figuur 2, figuur 3 en figuur 4. Hieronder volgt een gedetailleerde uitleg van de rollen en functionaliteiten van deze bestanden:

- Firebase Configuratie en Data Ophalen

  - “FetchProfileData.js”: Dit script is verantwoordelijk voor het ophalen van gebruikersprofielgegevens van Firebase, waardoor de applicatie de benodigde gebruikersinformatie kan verkrijgen.

  - “firebaseConfig.js”: Dit bestand bevat de configuratie-instellingen voor de verbinding van de applicatie met Firebase. Het omvat API-sleutels en andere instellingen die nodig zijn om Firebase-diensten te initialiseren.

- Zoekertjes

  - “createZoekertje.js, CreateZoekertje.html”: Deze bestanden beheren het aanmaken van nieuwe zoekertjes. Gebruikers kunnen de benodigde details invullen om een nieuw zoekertje aan te maken.

  - “acceptedZoekertjes.js, acceptedZoekertjes.html”: Deze bestanden beheren de zoekertjes die zijn geaccepteerd door de gebruiker.

  - “zoekertje.js, Zoekertje.html”: Deze bestanden worden gebruikt voor het weergeven van individuele zoekertjes en hun details.

  - “hoofdscherm.js, hoofscherm.html “: Deze bestanden beheren alles wat gebeurt op het hoofdscherm, dit is vooral het weergeven van beschikbare zoekertjes die matchen met de voorkeuren van de gebruiker.

- Google Maps Integratie

  - “mapLocatieKiezen.js”: Dit script stelt gebruikers in staat om een locatie op de Google Map te kiezen.

  - “mapLocaties.js, locatiesZoekertjes.html”: Dit script beheert de weergave van meerdere locaties op de kaart die zich bevindt in het html bestand.

  - “mapToonLocatie.js”: Dit script wordt gebruikt om een specifieke locatie op de kaart weer te geven.

- Gebruikersauthenticatie en Profielbeheer

  - “login.js, login.html, login.css”: Deze bestanden beheren het inlogproces van de gebruiker, zodat gebruikers kunnen inloggen op hun accounts.

  - “signup.js, signup.html, signup.css”: Deze bestanden beheren de gebruikersregistratie, waardoor nieuwe gebruikers een account kunnen aanmaken.

  - “EditProfile.js, EditProfile.html, EditProfile.css”: Deze bestanden stellen gebruikers in staat om hun profielinformatie te bewerken.

- “profiel.js, profiel.html, profiel.css”: Deze bestanden worden gebruikt voor het weergeven en beheren van het gebruikersprofiel.

- “profielAnderePersoon.js, profielAnderepersoon.html”: Deze bestanden zorgen ervoor dat er bij de details van een zoekertje doorgeklikt kan worden naar de maker ervan. Op deze manier kan de profiel informatie van de maker bekeken worden.

- Algoritme en Hulpscripts

  - “graaf.js, graaf.html”: Deze bestanden maken deel uit van het algoritme dat verantwoordelijk is voor het vinden van de beste matches voor gebruikers. De hoofdfunctionaliteit `findBesteMatches()` is hier geïmplementeerd.

  - “GraafConfig.js”: Dit script bevat configuratie-instellingen voor het matching-algoritme, o.a. de initialisatie van de graaf klasse gebeurt hier.

  - “RandomProfileGenerator.js”: Dit script genereert willekeurige gebruikersprofielen, dit werd gebruikt voor testdoeleinden.

- Overige Bestanden

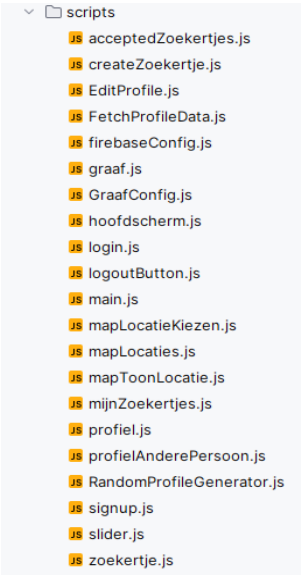
  - “index.html”: Het beginscherm van de webapplicatie.

  - “main.css, zoekertje.css, hoofdscherm.css “”: Deze bestanden bevatten de belangrijkste stijlregels voor de applicatie.

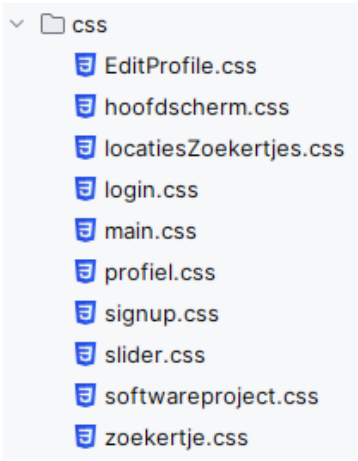
  - “slider.js”: dit bestand beheert een schuifcomponent, deze wordt gebruikt voor het kiezen van een leeftijdsinterval bij het creëren van een zoekertje.

  - “TermsandConditions.pdf”: Dit bestand bevat de algemene voorwaarden voor het gebruik van de applicatie.

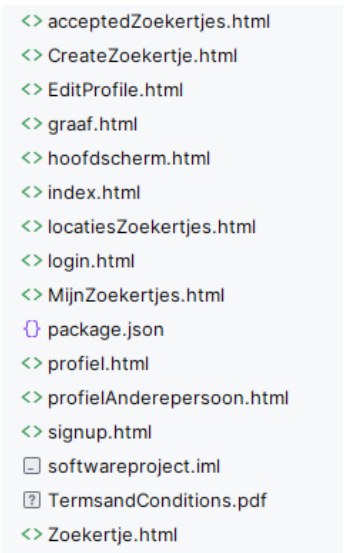
De webapplicatie is gestructureerd met een duidelijke scheiding van verantwoordelijkheden, waarbij taken zijn verdeeld over verschillende HTML-, CSS- en JavaScript-bestanden. Firebase wordt gebruikt voor backend-diensten, waaronder gebruikersgegevensbeheer en authenticatie.



Figuur 2: javascript bestanden



Figuur 3: css bestanden

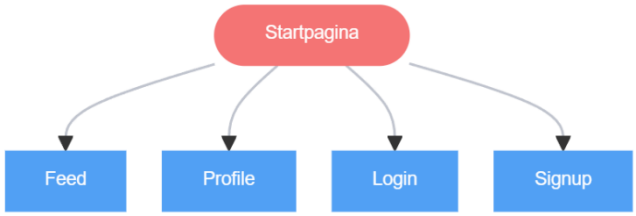


Figuur 4: html bestanden

2.2 DIAGRAMMEN

2.2.1 STARTPAGINA MOGELIJKHEDEN

Wanneer een gebruiker op de start pagina van het project komt zijn er zoals in onderstaande *figuur 5* te zien 4 mogelijkheden, namelijk: naar de pagina gaan waar alle zoekertjes staan die bij de gebruiker passen (feed), naar zijn profielpagina waar hij zijn gegevens kan overlopen en mogelijks aanpassen (profile), de gebruiker kan ook een account aanmaken (sign up) en de gebruiker heeft als laatste mogelijkheid ook de optie om in te loggen op zijn account (login).

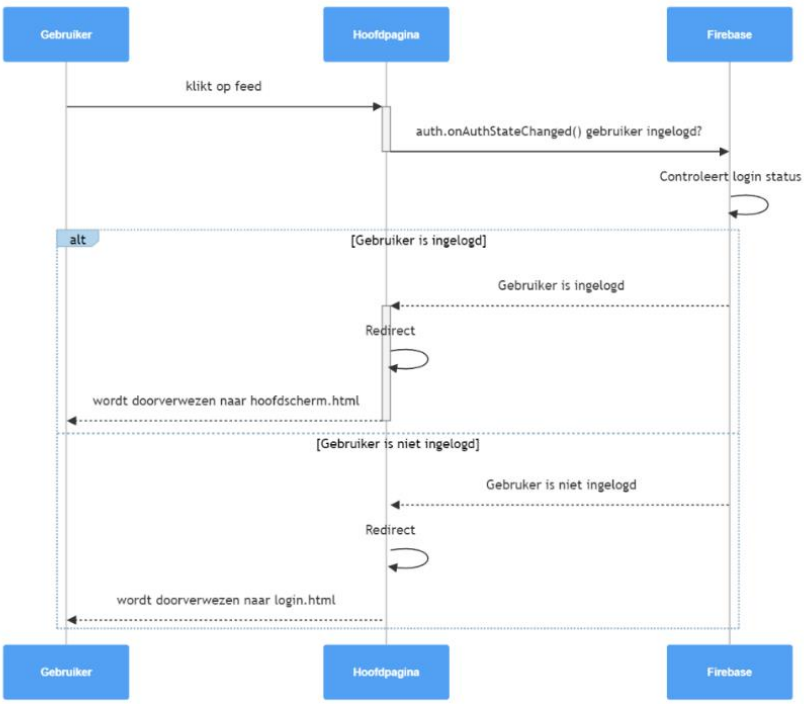


Figuur 5: : gebruiksmogelijkheden op de startpagina

2.2.2 KNOPPEN FEED EN PROFILE

Indien de gebruiker nog niet zou ingelogd zou zijn op de sportersconnectpagina zou het niet mogelijk mogen zijn om naar de pagina's feed en profile te navigeren. Daarvoor zijn er obstakels ingebouwd in de site zodat dit ongewenst gedrag niet voorkomt.

Figuur 6 toont het voorbeeld dat een gebruiker vanaf de startpagina naar de hoofdpagina wil navigeren, dan schiet er een controlemechanisme ingang dat gaat controleren of de gebruiker wel effectief ingelogd is. Dit mechanisme maakt gebruikt van de functie auth.onAuthStateChanged() van firebase, zie ook 3.1.1 voor meer uitleg over FireBase platform. Indien de gebruiker ingelogd is, wordt de gebruiker zonder enige hinder doorgestuurd naar de hoofdpagina, anders wordt de gebruiker doorverwezen naar de login pagina.

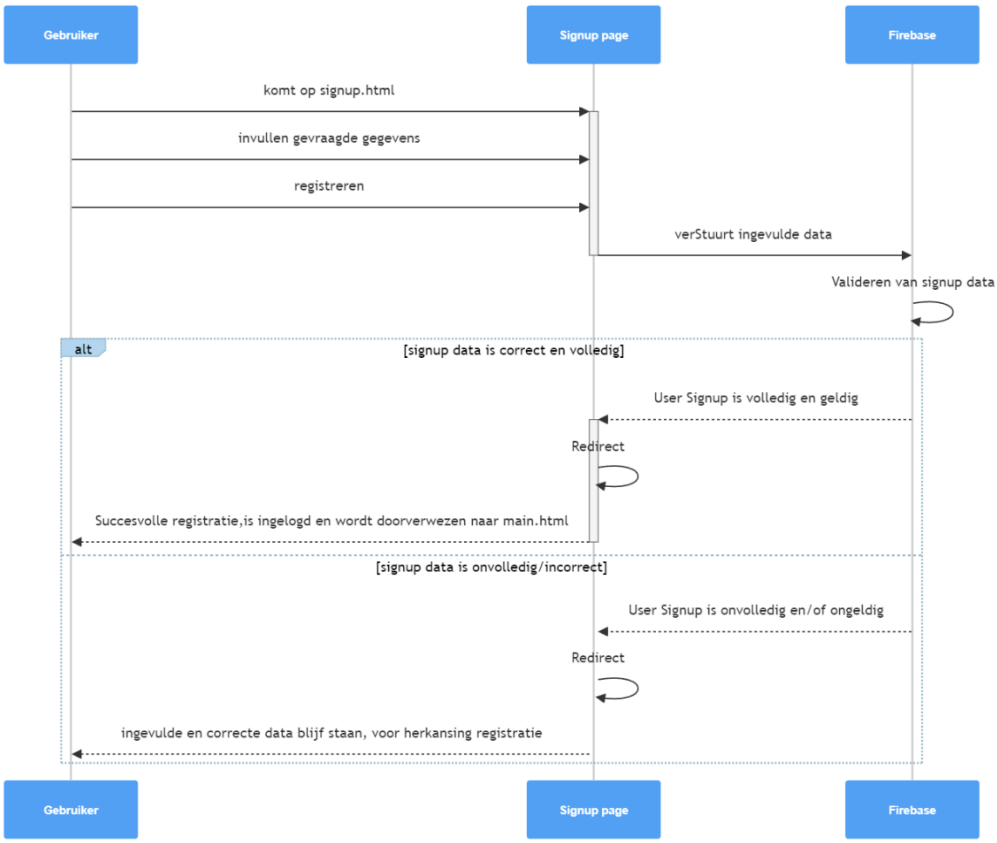


Figuur 6: Controlemechanisme bij het navigeren naar de feed

Eenzelfde controle vindt plaats wanneer de gebruiker wil navigeren naar de profiel pagina, alsook wanneer de gebruiker zich naar de login of sign up begeeft terwijl deze al ingelogd is. De gebruiker wordt dan doorgestuurd naar de hoofdpagina. Deze figuren worden niet opgenomen in dit verslag doordat *Figuur 6* de werking van deze controles voldoende toont.

2.2.3 REGISTRATIE EN INLOGGEN VAN EEN GEBRUIKER

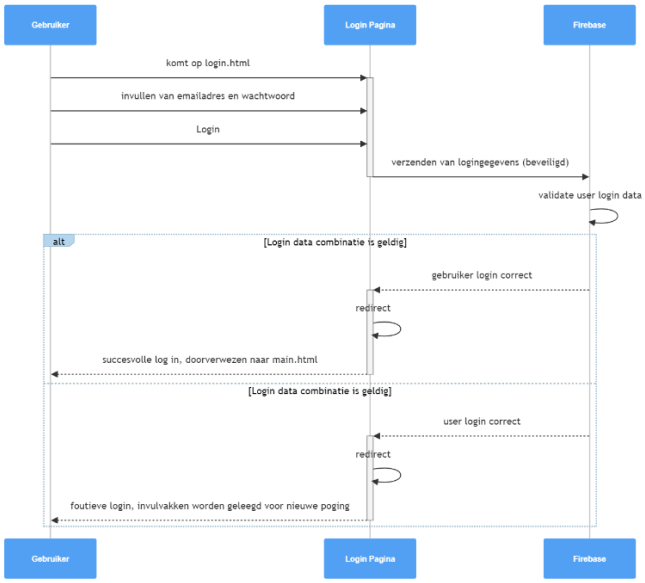
Wanneer Er een nieuwe gebruiker op de SportersConnect pagina komt, begeeft de gebruiker zich naar de registratie pagina (sign up). Daar vult de gebruiker alle benodigde gegevens in, deze gegevens worden dan verstuurd naar de firebase waar er per gebruiker data opgeslagen wordt met de nodige ingestelde veiligheden en controles, zie ook 3.1.1. Als de data door de gebruiker correct en volledig ingevuld is wordt er een account aangemaakt en wordt de gebruiker ingelogd en doorverwezen naar zijn hoofdpagina. Indien dit niet het geval is zal de gebruiker, zoals te zien is in *figuur 7*, een herkansing krijgen om zijn gegevens verder aan te vullen of aan te passen.



Figuur 7: registratie van een gebruiker

Wanneer de pagina te maken heeft met een terugkerende gebruiker moet er geen registratie meer plaatsvinden de gebruiker moet zich enkel nog inloggen, indien dit nog niet gebeurd is (zie bovenstaande). De gebruiker vult zijn emailadres en wachtwoord in, deze worden op een beveiligde manier verstuurd naar de firebase waar ze gecontroleerd worden. Indien deze combinatie incorrect is,

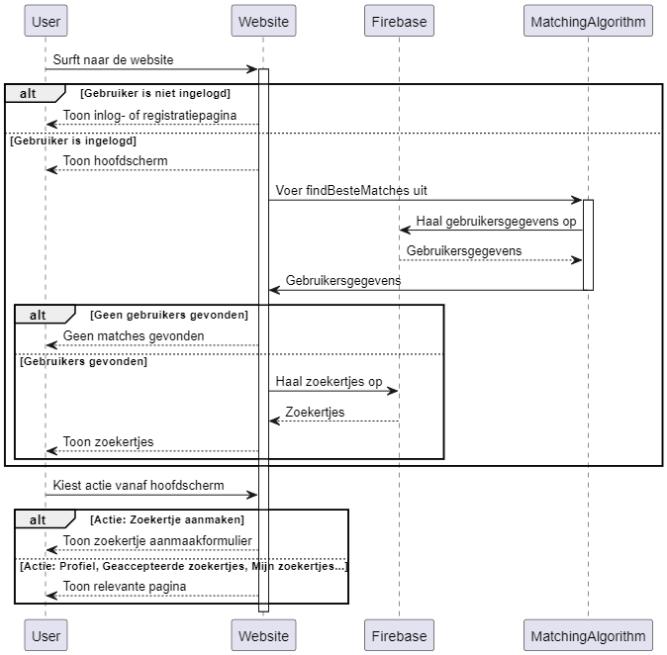
worden de invulvakken leeggemaakt voor een nieuwe poging zoals te zien is in *figuur 8*. Wanneer deze wel correct is, wordt de gebruiker ingelogd en doorgestuurd naar de hoofdpagina.



Figuur 8: Inloggen van een gebruiker

2.2.4 MOGELIJKHEDEN OP DE HOOFDPAGINA MET BETREKKING TOT ZOEKERTJES

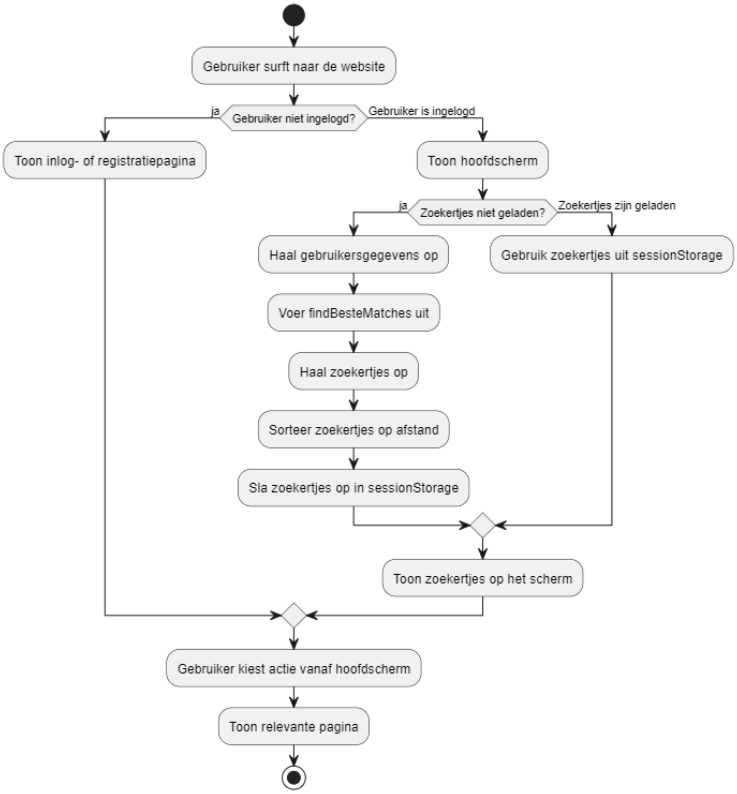
Wanneer een gebruiker op de hoofdpagina terecht komt, krijgt de gebruiker een aantal zoekertjes te zien die passen bij de gegevens die hij invulde bij registratie. Welke Zoekertjes er exact getoond worden, hangt af van het matchingalgoritme. Deze koppelt de kenmerken van een gebruiker aan de zoekertjes van andere gebruikers. Hoe dit algoritme exact ineen zit wordt uitgelegd in deelhoofdstuk 3.2. Wanneer er matchende zoekertjes zijn (en dus ook andere gebruikers) worden de nodige gebruikersdata en zoekertjesdata, zoals getoond op *figuur 9,* opgehaald uit de firebase om de zoekertjes te tonen aan de gebruiker.



Figuur 9: sequentie diagram, tonen van Zoekertjes aan gebruiker

Figuur 10 toont meer in detail hoe de zoekertjes worden getoond. Eerst wordt er gecontroleerd of de gebruiker al aangemeld is, zoals hierboven besproken. Daarna zal de pagina controleren of de zoekertjes al eerder ingeladen werden, als dit het geval is worden de zoekertjes direct aan de gebruiker getoond. Dit is vaak niet het geval, dan worden de gegevens van de gebruiker uit de Firebase gehaald om de functie findBesteMatches() uit te voeren. Om zo geschikte zoekertjes voor de gebruiker te vinden.

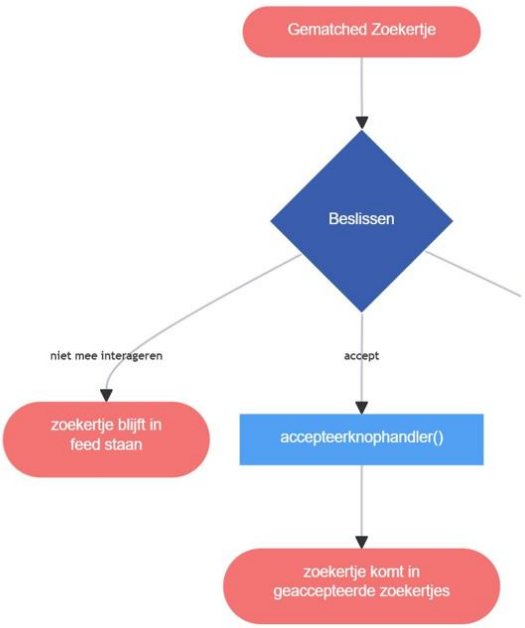
Als deze gevonden worden, worden deze ook opgehaald uit de firebase, waarna ze gesorteerd worden op basis van de afstand tussen de gebruiker en het zoekertje. Deze zoekertjes worden dan opgeslagen in de sessionstorage van de browser van de gebruiker. Dit laat toe om de data te behouden tot de browser wordt gesloten, zo moet de data niet telkens opnieuw opgehaald worden bij het herladen van de pagina. Deze zoekertjes worden dan getoond aan de gebruiker die mogelijke acties kan ondernemen.



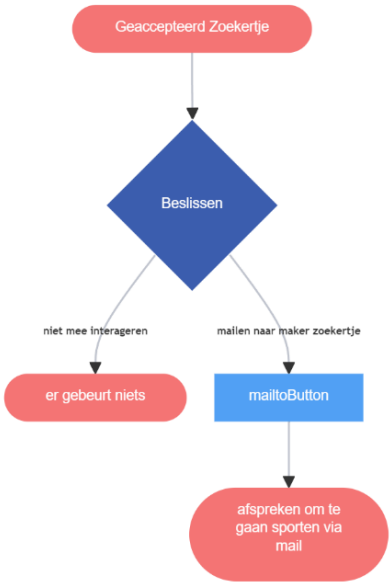
Figuur 10: activiteiten diagram hoofdscherm

2.2.5 GEVOLGEN VAN GEACCEPTEERDE ZOEKERTJES

Met de zichtbare zoekertjes kunnen de gebruikers een aantal zaken doen zoals te zien is in *figuur 11*. Zo kan de gebruiker het zoekertje gewoon laten staan om hier later op terug te komen. Het zoekertje kan ook geaccepteerd worden dan zal dat zoekertje zich niet meer in het hoofdscherm bevinden maar wordt dit geplaatst in de pagina met de geaccepteerde zoekertjes. Op deze pagina kunnen de details van het zoekertje nog eens bekeken worden, of kan er naar de maker van het zoekertje gemaild worden zoals aangegeven in *figuur 12*.



Figuur 12: Mogelijkheden met een zoekertje in de feed



Figuur 11: Mogelijkheden geaccepteerde zoekertjes

3. COMPLEXE STUKKEN CODE

3.1 DATA EN FIREBASE

Alle data die moet worden bijgehouden wordt opgeslagen in google Firebase. Dit is een ontwikkelplatform gemaakt door Google die ondersteuning biedt in meerdere onderdelen van mobiele- en web-apps.

Ten eerste moet de firebase correct opgezet worden.

Bij het opzetten van een Google Firebase worden er aan die firebase specifieke sleutels meegegeven. Dit zijn zaken zoals de apikey, het projectID, de storageBucket en nog enkele andere waarden. Deze waarden zijn van groot belang want zonder de juiste waarden voor deze sleutels kan de firebase niet worden bekeken. Al deze sleutels worden in een bestand genaamd firebasconfig.js gestoken, zodat alle sleutels maar eenmalig in de code moeten worden opgeschreven. Het is belangrijk dat deze waarden nergens gedeeld worden, anders kunnen onbevoegde gebruikers ook aan de firebase.

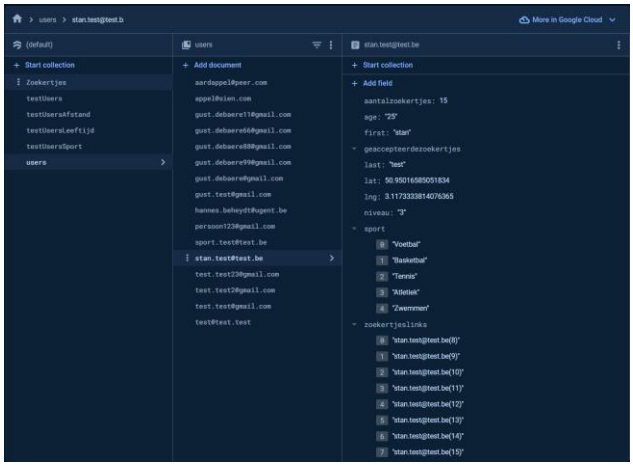
Na de setup zijn er erg veel verschillende hulpmiddelen die kunnen geïmplementeerd worden naar de vereisten van de applicatie. In de website wordt er gebruikgemaakt van 3 onderdelen.

3.1.1 FIRESTORE DATABASE

Het eerste onderdeel waar gebruik van wordt gemaakt is de Firestore Database. Het is een NoSQL Database, dat wil zeggen dat er geen gebruik gemaakt wordt van tabellen of rijen. In plaats daarvan wordt data opgeslagen in documenten, die georganiseerd worden in collecties. Er wordt data opgeslagen in 2 hoofdcollecties voor echte gebruikers, en in 4 testcollecties die bedoeld zijn om testen uit te voeren. Dit wordt weergegeven in figuur 13.

De eerste hoofdcollectie is de “users” collectie. Hierin wordt per gebruiker een document aangemaakt die alle info bevat over de gebruiker. Het document krijgt als naam de email van de gebruiker. In het document wordt volgende data bewaard:

- De voor- en achternaam van de gebruiker
- De leeftijd van de Gebruiker
- coördinaten van de locatie van de gebruiker
- het niveau van de sporter
- het aantal zoekertjes dat een account al gemaakt heeft.
- een lijst met titels van zoekertjes die de gebruiker zelf gemaakt heeft
- een lijst met titels van zoekertjes die de gebruiker geaccepteerd heeft.



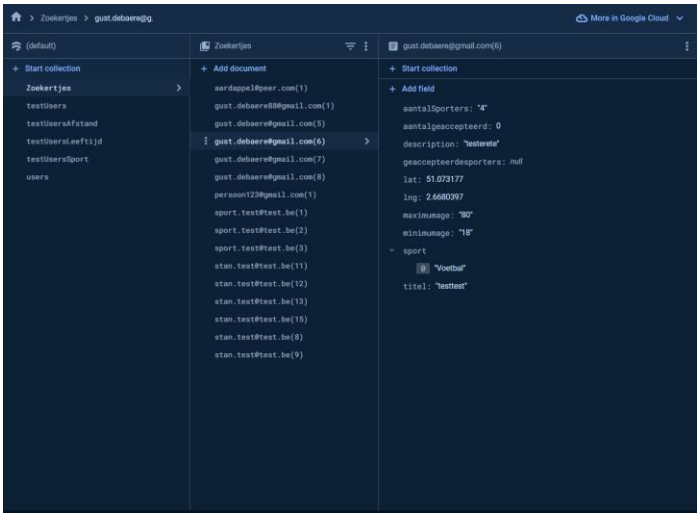
Figuur 13: indeling Firebase collectie users

Om deze data gemakkelijk op te halen, werd een bestand gemaakt genaamd fetchProfileData.js, dat alle data ophaalt van Firebase om gedupliceerde code te vermijden.

De tweede hoofdcollectie is de “Zoekertjes” collectie. Elk document in deze collectie stelt een zoekertje voor. Wanneer een zoekertje gemaakt wordt zal de waarde van het aantal zoekertjes dat een account al gemaakt heeft verhogen met 1. Dit wordt het specifieke zoekertjesnummer. De titel van het document van het nieuwe zoekertje wordt dan het emailadres van de gebruiker met tussen haakjes het zoekertjesnummer. In de documenten van elk zoekertje wordt volgende data bijgehouden:

- De titel en de beschrijving van het zoekertje
- de sport waarvoor mensen gezocht worden
- het aantal gezochte sporters
- het aantal sporters die al geaccepteerd hebben
- de coördinaten van waar de sport beoefend zal worden
- de leeftijdsrange waar de maker van het zoekertje mensen uit zoekt

Dit wordt ook weergegeven in figuur 14.



Figuur 14: indeling firebase collectie zoekertjes

3.1.2 GOOGLE FIREBASE STORAGE

Het tweede onderdeel waar de website gebruik van maakt is Google Firebase Storage.

Dit doet ongeveer hetzelfde als de Firestore Database, op de manier dat het ook data bijhoudt en opslaat, maar deze storage kan ook gebruikt worden om andere bestandstypes bij te houden, zoals afbeeldingen en video's. De website gebruikt dit om alle profielfoto's bij te houden van de gebruikers. Bij het aanmaken van het profiel wordt er een bestand aangemaakt met de profielfoto van de gebruiker in deze storage. Als de gebruiker geen profielfoto geselecteerd heeft, word een standaard *placeholder* gebruikt. De afbeeldingen in deze storage kunnen van het type "jpg" of "png" zijn.

3.1.3 GOOGLE FIREBASE AUTHENTICATION

Het laatste onderdeel waar de website gebruik van maakt is Google Firebase Authentication.

Google Firebase biedt een eenvoudig te implementeren authenticatiesysteem. Er zijn verschillende opties beschikbaar om gebruikers in te laten loggen. Voor dit project werd er gekozen om in te loggen met email en passwoord, maar er kan ook gekozen worden voor inloggen met facebook, Microsoft, GitHub, etc... Alle paswoorden worden versleuteld door google Firebase zodat er een lager beveiligingsrisico is. De makers van de Firebase kunnen zelf de wachtwoorden van de gebruikers ook niet zien.

Het implementeren van de Authenticatie gaat als volgt:

Er wordt een pagina gemaakt waar gebruikers hun informatie kunnen invullen. De informatie over de gebruiker zelf wordt naar de firestore Database geschreven. Het emailadres en het wachtwoord van de gebruiker worden gebruikt om een nieuwe user in de Authentication aan te maken. Wanneer er dan een login pagina gemaakt wordt en de juiste combinatie van naam en wachtwoord ingevoerd worden, herkent de firebase de gebruiker en kan er ingelogd worden.

Er moet aan het begin van de loginpagina de juiste authenticatie persistentie ingesteld worden. Dit bepaalt hoelang de gebruiker ingelogd blijft. Er zijn 3 opties. De eerste optie is dat een gebruiker ingelogd blijft enkel op deze pagina, wanneer de pagina herlaadt of doorgestuurd wordt naar een andere pagina, wordt de gebruiker terug uitgelogd. De tweede optie is dat de gebruiker ingelogd blijft tot dat de browser gesloten wordt. De derde optie is dat de gebruiker altijd ingelogd blijft tenzij dat er zelf voor wordt gekozen om uit te loggen. Er wordt in het project gebruik gemaakt van de tweede optie, de gebruiker blijft ingelogd tijdens het gebruik van de site, zolang de browser niet gesloten wordt.

Op iedere pagina wordt er aan het begin gecontroleerd of de gebruiker is ingelogd. Als dit niet het geval is wordt de gebruiker naar het loginscherm gestuurd. Dit gebeurt om te vermijden dat een gebruiker op een scherm komt waar normaal informatie weergegeven wordt over de gebruiker, zonder ingelogd te zijn.

3.2 PROFIEL MATCHING ALGORITME

Het ontworpen algoritme maakt gebruik van verschillende manieren om profielen te matchen met elkaar. Als eerste is er een Graafstructuur, deze houdt de locatie van de verschillende profielen bij met relatieve afstand ten opzichte van elkaar. Daarnaast is er ook een simpeler systeem dat de verschillende eigenschappen van een profiel in een getal zal omzetten. Hoe hoger dit getal is hoe beter de match tussen de twee profielen.

Als eerste wordt er wat beter gekeken naar de graafstructuur. De graafstructuur wordt opgeslagen in een klasse, deze klasse bevat dan ook de graaf. De graaf wordt geïnitialiseerd met behulp van coördinaten. Dit kan doordat telkens er een nieuw profiel aangemaakt wordt, de persoon een locatie op de kaart moet aanduiden. Deze locatie wordt samen met de rest van de gegevens van het profiel opgeslagen in de firestore database.

Telkens wanneer de website ingeladen wordt, wordt er gekeken naar de firestore database. Hieruit worden alle opgeslagen locaties van de profielen opgehaald en op deze manier wordt de graaf terug geïnitialiseerd.

Om te bepalen hoe ver twee profielen zich van elkaar bevinden, kan een hulpfunctie worden gebruikt om de afstand tussen de twee coördinaten te berekenen.

Als tweede wordt de gewogen matchmaking onder de loep genomen. Deze werkt op basis van de verschillende profiel eigenschappen die worden doorgegeven aan de firestore database wanneer een persoon een nieuw profiel aanmaakt. Dit gaat dan onder andere over leeftijd, sportinteresses en sportniveau.

De graafklasse die hierboven al werd aangehaald bevat ook manieren om al deze eigenschappen bij te houden. Dus wanneer de website geladen wordt, zullen al deze eigenschappen van de bestaande profielen ook geherinitialiseerd worden zodat deze later veel sneller kunnen worden gebruikt.

De gewogen matchmaking wordt gedaan aan de hand van een functie. Deze functie kijkt voor het profiel dat ingelogd is op de website welke andere profielen het best matchen.

Als eerste wordt binnen deze functie het profiel gezocht dat het verst gelegen is van het huidige profiel, deze afstand wordt later gebruikt om normalisatie te kunnen toepassen. Deze afstanden kunnen berekend worden dankzij de volledig geïnitialiseerde graaf.

Daarna zal elk profiel dat opgeslagen zit in de graaf overlopen worden. De afstand tussen de profielen wordt berekend, er wordt gecontroleerd of er minstens 1 gemeenschappelijke sport is die tot de interesses behoord. Als dit niet het geval is wordt het profiel overgeslagen. De afstand wordt ook genormaliseerd aan de hand van de eerder berekende maximale afstand. Door deze normalisatie wordt een representatief beeld verkregen van de verschillende afstanden.

Verder worden ook de andere eigenschappen vergeleken met elkaar en omgezet naar getallen. Met deze getallen kan een score berekent worden die aantoont hoe goed de twee profielen matchen. Bij deze omzetting kan er gekeken worden welk van de eigenschappen een grotere of minder grote invloed moet hebben dan de rest. Zo heeft de locatie een grotere doorweegfactor dan het leeftijdsverschil.

Als laatste worden score en profiel opgeslagen. Wanneer alle profielen uit de graaf aan bod zijn geweest worden deze gegevens geordend zodat het profiel met de hoogste score eerst staat. Hier kan er dan ook beslist worden hoeveel van de beste matches er gebruikt moeten worden voor volgende stappen.

3.3 PROFIEL ALGORITME IN COMBINATIE MET ZOEKERTJES

Wanneer een gebruiker de website opent is het eerste dat verschijnt het hoofdscherm met een aantal zoekertjes van andere profielen. Deze zichtbare zoekertjes zijn niet zomaar willekeurig gekozen, achter de schermen gebeurt namelijk een complexe berekening die kiest welke zoekertjes geschikt zijn voor welke gebruiker.

Het eerste dat gebeurt, werd hierboven al besproken. Vanaf dat een nieuw profiel wordt aangemaakt wordt deze opgeslagen in de database. Als de gebruiker van dit profiel naar het hoofdscherm surft treedt het profiel matching algoritme in werking, de beste matches voor de huidige gebruiker worden bijgehouden. Wanneer de pagina herladen wordt, gebeurt de matching opnieuw om rekening te houden met nieuw toegevoegde gebruikers.

Daarna treedt automatisch een tweede ordening proces in werking. De profielen die het beste matchen met de huidige gebruiker worden elk overlopen om te kijken of deze openstaande zoekertjes hebben. Als dit het geval is wordt gekeken of het zoekertje gaat over een sport die ook tot de interesses van de huidige gebruiker behoord. Daarna wordt het zoekertje of wel toegevoegd aan de wachtrij die op het hoofdscherm mag worden weergegeven, ofwel wordt het overgeslagen omdat het niet correspondeert met de huidige gebruiker.

De zoekertjes die dit proces doorstaan worden daarna weergegeven op het hoofdscherm zodat de gebruiker deze kan bekijken.

3.4 ZOEKERTJES

Zoekertjes zijn een zeer belangrijk deel van de site. In bijna iedere pagina wordt er wel gebruik van gemaakt. Een zoekertje kan worden aangemaakt door een gebruiker van de site, met als doel andere gebruikers te vinden die dezelfde interesse hebben. Wanneer een andere gebruiker een zoekertje ziet dat bij zich past, kan deze persoon het accepteren. Dan komt het tussen zijn lijst van geaccepteerde zoekertjes terecht. Ook wordt bij het zoekertje zelf dan het aantal geaccepteerde sporters aangepast. Het profiel van de gebruiker die het geaccepteerd heeft wordt getoond bij de weergave van het zoekertje.

3.4.1 HET AANMAKEN VAN EEN ZOEKERTJE

Bij de pagina genaamd "Zoekertje aanmaken" kan een gebruiker een zoekertje creëren. Hier geeft de gebruiker alle informatie op zodat alles aan de gestelde eisen voldoet en creëert het zoekertje. Op dat moment wordt er in de Firestore Database een nieuw document aangemaakt in de collectie "zoekertjes" met die informatie. Het aantal geaccepteerde sporters wordt standaard op nul gezet.

3.4.2 HET TONEN VAN DE ZOEKERTJESLIJST

Op het hoofdscherm, bij de feed, worden alle zoekertjes weergegeven die overeenstemmen met het account van de gebruiker. Er wordt gecontroleerd of de leeftijd van de gebruiker binnen het leeftijdsinterval van het zoekertje valt, of de sport bij het zoekertje een van de sporten van de gebruiker is, enzovoort. Eigen zoekertjes worden niet weergegeven in de feed.

De zoekertjes die weergegeven worden, worden ook gesorteerd op hoe goed de accounts van de gebruikers bij elkaar horen. Dit wordt gedaan op basis van bovengenoemd algoritme. Zoekertjes in de feed tonen enkel de titel en de beschrijving. Op elk zoekertje kan geklikt worden. Dan wordt de gebruiker doorgestuurd naar de gedetailleerde weergave van het zoekertje.

3.4.3 DE GEDETAILEERDE WEERGAVE VAN EEN ZOEKERTJE

Na het klikken op een zoekertje wordt de gedetailleerde weergave van een zoekertje getoond.

Hier wordt opnieuw de titel en de beschrijving getoond, maar ook alle andere informatie. Het leeftijdsinterval, het gezocht aantal sporters en de locatie worden hier getoond. Er is ook een afdeling voorzien die de informatie over de maker van het zoekertje toont. Dit is ook een link naar het account van de gebruiker. Hetzelfde wordt gedaan voor iedere geaccepteerde gebruiker, als die er zijn.

Afhankelijk of het zoekertje van de gebruiker is of niet worden er ook enkele knoppen voorzien. Als het zoekertje van de actieve gebruiker is, krijgt deze de mogelijkheid om het zoekertje te verwijderen. Indien het van een andere gebruiker is kan het zoekertje geaccepteerd worden.

Als een zoekertje geaccepteerd is door een gebruiker, zal de gebruiker het zoekertje niet meer zien staan tussen zijn feed maar wel in zijn lijst met geaccepteerde zoekertjes.

4. SUSTAINABLE DEVELOPMENT GOALS

In het ontwikkelen van een website is het van groot belang om rekening te houden met duurzaamheidsdoelstellingen (SDG's). Een strategie die hierbij kan helpen, is het implementeren van validatieprocessen voor formulieren en externe invoer.

Allereerst, door foutvalidatie in real-time en bij het indienen van formulieren toe te passen, kunnen gebruikers geholpen worden om fouten te corrigeren voordat deze leiden tot problemen.

Ten tweede, door duidelijk gelabelde vereiste elementen te hebben, verbetert de toegankelijkheid van de website voor alle gebruikers, inclusief diegenen die afhankelijk zijn van schermlezers en virtuele assistenten.

In het project moet er vaak een formulier ingevuld worden, dus is het belangrijk dat er hier rekening mee wordt gehouden om problemen te voorkomen. Voor dit te realiseren wordt er gebruik gemaakt van de authenticatie die in Firebase zelf zit en limieten opstellen voor de numerieke waarden die moeten ingegeven worden.

Een 2de sdg waar er rekening mee gehouden werd is Het minimaliseren van HTML, CSS en JavaScript. Dat is een goede stap in web ontwikkeling om de prestaties van een website te optimaliseren. Door het verwijderen van overbodige witruimte en opmaak uit de broncode kunnen aanzienlijke gegevensbesparingen worden gerealiseerd, wat resulteert in snellere laadtijden van webpagina's.

Deze strategie omvat het comprimeren van alle broncode, inclusief *inline* code, tijdens de compilatie. Dit minimaliseert de omvang van de bestanden die naar de gebruikers worden verzonden, wat leidt tot verminderde laadtijden. Hoewel dit op zichzelf geen direct ecologisch voordeel heeft omdat witruimte wordt genegeerd door rendermachines, draagt het bij aan duurzaamheidsdoelen door verbeteringen voor bezoekers in termen van laadtijden.

In het project werd er vooral op gelet dat er geen onnodige witruimtes of ongebruikte code aanwezig was.

De 3de sdg waar er rekening mee werd gehouden is het gebruik van gunstige JavaScript en zijn API's. Dat is een belangrijk aspect van web ontwikkeling, gericht op het verbeteren van de gebruikerservaring en het bevorderen van duurzaamheid

In het project werd er gebruik gemaakt van 2 API’s die de werking van de website een stuk verbeterden. De firebase is een handig backend-end platform waar gegevens opgeslagen kunnen worden, maar het biedt nog veel meer functionaliteit aan dan dat. De google maps API is heel praktisch om een kaart te integreren op de website en heeft ook heel wat functionaliteiten ter beschikking om toe te passen op deze mappen.

Als websiteontwikkelaars is het van essentieel belang om deze principes te omarmen en te integreren in het project. Door deze strategieën toe te passen, verbetert de gebruikerservaring niet alleen, maar wordt er ook bijgedragen aan een meer duurzame en inclusieve wereld.

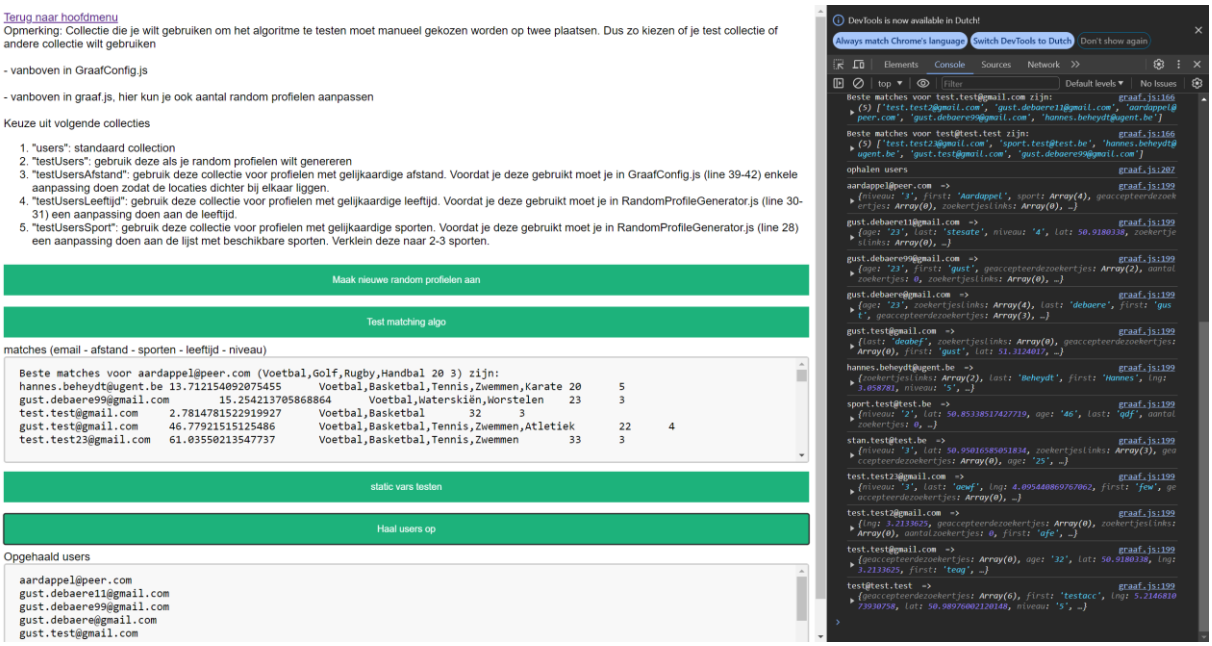
5. TESTEN

Tijdens het creatieproces van het algoritme dat profielen aan elkaar moet koppelen, was voortdurend testen en observeren van groot belang. Een van de methoden die hiervoor werd gebruikt was een random profielgenerator. Met deze generator konden de kenmerken ingesteld worden die de gecreëerde profielen moesten hebben. Op deze manier kon het gedrag van het algoritme worden geobserveerd en geanalyseerd.

Door herhaaldelijk gebruik te maken van de random profielgenerator kon het algoritme steeds verder worden aangepast en verfijnd. Het doel was om uiteindelijk een versie te ontwikkelen die in staat was om profielen nauwkeurig en correct te matchen in alle mogelijke scenario's.

Na meerdere pogingen en aanpassingen werd uiteindelijk een algoritme verkregen dat voldeed aan de verwachtingen. Dit algoritme was in staat om profielen op een effectieve manier te ordenen op basis van hun kenmerken. Het resultaat was een systeem dat in staat was om potentiële matches te vinden die goed pasten bij de voorkeuren en eigenschappen van de gebruikers.

Om zelf aan de slag te gaan met het testen van het algoritme is er een menu-item voorzien op het hoofdscherm dat de gebruiker naar het scherm van figuur 15 brengt. Hier is extra informatie gegeven over hoe de gebruiker zelf aan de slag kan gaan om op verschillende manieren het profiel matching algoritme uit te proberen. Hiervoor moeten telkens wel een paar aanpassingen in de code worden gedaan.



Figuur 15: testomgeving algoritme

6. BESLUIT

Nu dat het project gedaan is zijn er enkele zaken die heel goed gegaan zijn en andere waar er iets te weinig tijd voor was. Er is door alle groepsleden veel bijgeleerd over het werken met een Database en met javascript. Nog niemand had hiervoor met google firebase gewerkt, maar nu dat iedereen ermee kan werken, blijkt het een heel handige tool te zijn. Moest er nog meer tijd geweest zijn ging er nog een chatfunctie geimplementeerd worden. Momenteel kunnen de gebruikers via mail communiceren maar het zou eenvoudiger en beter zijn moest er in de website een ingebouwde chat gebruikt kunnen worden.

7. DOCUMENTATIE

Google firebase documentatie:

Google. "Firebase Documentatie." Geraadpleegd op 19 mei 2024. https://firebase.google.com/docs/

Slider voor het maken van de leeftijdsrange:

Davidovic, Predrag. "Native Dual Range Slider HTML, CSS, JavaScript." Medium, 2 april 2022. https://medium.com/@predragdavidovic10/native-dual-range-slider-html-css-javascript-91e778134816

Wikipedia over Grafentheorie:

"Wikipedia: Grafentheorie." Geraadpleegd op 12 april 2024. https://nl.wikipedia.org/wiki/Grafentheorie

foto's startpagina webapp:

Miller, Luke. "A Person Jumping a Snow Board in the Air." Unsplash.https://unsplash.com/photos/a-person-jumping-a-snow-board-in-the-air-E9WXfIYbo3c?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Bystrov, Viktor. "Mens Black Bike Helmet." Unsplash.https://unsplash.com/photos/mens-black-bike-helmet-Gi0OMNguFaw?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Veer, KaLisa. "Man Surfing on Waves." Unsplash.https://unsplash.com/photos/man-surfing-on-waves-gRx74OSJTG8?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Google Maps documentatie:

Google. (z.d.). Google Maps Platform documentatie. Geraadpleegd op 23 april 2024, van https://developers.google.com/maps/documentation

Bronnen van SDG's:

"Sustainable Web Design Guidelines: Gebruik nuttige JavaScript en zijn API's." Geraadpleegd op 15 mei 2024. https://sustainablewebdesign.org/guidelines/3-15-use-beneficial-javascript-and-its-apis/

"Sustainable Web Design Guidelines: Minimaliseer uw HTML, CSS en JavaScript." Geraadpleegd op 15 mei 2024. https://sustainablewebdesign.org/guidelines/3-2-minify-your-html-css-and-javascript/

"Sustainable Web Design Guidelines: Valideer formulierfouten en externe invoer." Geraadpleegd op 15 mei 2024. https://sustainablewebdesign.org/guidelines/3-11-validate-form-errors-and-external-input/

Chatgpt:

OpenAI. ChatGPT. Versie GPT-3.5. Persoonlijke communicatie, geraadpleegd op 19 mei 2024. https://www.openai.com/chatgpt