

analyse

THOMAS MORE - EURICOM

Wout Jacobs

THOMAS MORE | KLEINEHOEFSTRAAT 4, 2440 GEEL

Inhoud

De opdrachtgever(Euricom)	4
Teams	4
Project information.....	4
Probleemstelling.....	5
Projectafbakening en Risicoanalyse	5
Afbakening van verantwoordelijkheden	6
Risico's onder de verantwoordelijkheid van de opdrachtgever.....	6
Voorgestelde maatregelen om risico's te beperken	7
Requirements	7
Must Have	7
Should have.....	8
Would have	8
Game Logica	9
Game mode 1	9
Game mode 2	10
Elo-rating: Wat is het en hoe werkt het?.....	10
Nieuwe Elo-rating berekenen:	10
Aanpassingen voor de Gotcha Game Mode.....	11
Spelverloop en Mechanieken.....	11
Target Toewijzing: Dynamische Array Methode	12
Drempelwaarden voor de array:	12
Target-selectieproces:	12
Scoresysteem tijdens het Spel	12
Winnaar Bepalen	12
Samenvatting van de Game Mode.....	13
Front-end.....	13
Vue.....	13
Wanneer kies je voor Vue?	13
React	14
Wanneer kies je voor React?.....	14
Eind Conclusie	14
Back-end	15
.NET	15
Databank.....	16
Belangrijkste kenmerken	16

Wanneer kiezen voor Microsoft SQL Server?	17
Prototypes	18
Login	18
Home	18
Create Game	19
Geselecteerde game	19
Game rules	20
QR-Code page	20
Game Statistieken	21
Speler gegevens per game	21
Persoonlijke Statics	22
Admin Home Page	22
Admin Toon spelers in zijn game	23
Admin invite speler	23
Super Admin	24
Show all players	24
All Games	25
Add achievements	25
Add rules	26
Invite page	26
Achievements	27
Authentication / login	27
Login met Microsoft (Entra ID, SSO) via MSAL.js	27
Voordelen van deze aanpak:	27
Unieke QR-code	28
Hoe maken we de QR-code uniek?	28
Waarom kiezen voor qrcode.react?	28
QR-code scannen?	28
Clean code	29
Clean Architecture	29
De structuur van Clean Architecture bestaat doorgaans uit vier lagen:	29
3-Layer Programming	30
De structuur van 3-Layer Programming bestaat uit de volgende lagen:	30
Verschillen tussen Clean Architecture en 3-Layer Programming	31
Waarom kiezen voor Clean Architecture in jouw project?	31
PWA	31

Waarom kiezen we voor een PWA?	31
PWA implementeren met React	32
Waarom is een PWA ideaal voor dit project?	32
Grafieken	32
Recharts	32
Chart.js (via react-chartjs-2)	33
ApexCharts (via react-apexcharts)	33
Conclusie	33
Styling conclusie aanpassen met prime react en tailwind	34
Tailwind CSS	34
Material UI (MUI)	34
Bootstrap (via React-Bootstrap)	34
PrimeReact	35
Conclusie	35
Pipeline	36
Hoe kan ik een CI/CD Pipeline gebruiken voor dit project?	36
Waarom is een CI/CD Pipeline nuttig voor dit project?	36
Tools die je kunt gebruiken voor de CI/CD Pipeline:	37
Conclusie	37
Error handling	38
Waarom Application Insights gebruiken?	38
Welke gegevens kan je analyseren met Application Insights?	38
Hoe analyseer je gegevens in Application Insights?	39
Waarom is dit belangrijk?	39

De opdrachtgever(Euricom)

Euricom is een dynamisch en innovatief bedrijf dat zich richt op ICT-oplossingen en technologische dienstverlening. Ze leveren een breed scala aan diensten en producten, variërend van cloudoplossingen tot softwareontwikkeling en consultancy voor diverse industrieën. Met een stevige focus op klantgerichte oplossingen, streeft Euricom ernaar bedrijven te ondersteunen bij het realiseren van hun digitale transformatie.

Het bedrijf heeft een sterke reputatie opgebouwd door technologische expertise te combineren met praktische bedrijfskennis. Euricom werkt samen met bedrijven van verschillende groottes en sectoren, waarbij ze zich inzetten om efficiëntie te verbeteren, innovatie te stimuleren, en bedrijfsprocessen te optimaliseren door middel van technologie.

Euricom staat bekend om hun open bedrijfscultuur, waarin samenwerking en kennisdeling centraal staan. Ze bieden een omgeving waarin creativiteit en technische kennis worden gewaardeerd, wat het bedrijf een aantrekkelijke werkplek maakt voor technische professionals en ontwikkelaars.

Als opdrachtgever in dit project is Euricom gericht op het ontwikkelen van een game-achtige applicatie voor hun jaarlijkse DEV-Cruise. Dit project richt zich op het verbeteren van de teamdynamiek en samenwerking tijdens dit evenement, wat ook aansluit bij hun algemene focus op innovatie, teambuilding en het versterken van de bedrijfscommunity.

Teams

De succesvolle realisatie van de Gotcha Applicatie is afhankelijk van de samenwerking van verschillende gespecialiseerde teams, waarbij elk team een cruciale rol speelt in de ontwikkeling:

- Het team van stagiairs en mentoren: Dit team is verantwoordelijk voor de praktische uitvoering van de Gotcha applicatie. Jullie werken aan het ontwerpen, ontwikkelen en implementeren van de functionaliteiten, zoals het scannen van QR-codes voor eliminaties en het beheren van spelersdata. Daarnaast zorgen jullie ervoor dat de applicatie goed presteert, gebruiksvriendelijk is en voldoet aan de gestelde eisen, met begeleiding van de mentoren.
- Euricom (klant): Euricom is de opdrachtgever voor de Gotcha applicatie en zorgt voor de visie, doelen en vereisten van het project. Zij zorgen ervoor dat de applicatie aansluit bij hun specifieke wensen en eisen en begeleiden het team gedurende het ontwikkelproces.

Samen werken deze teams aan de ontwikkeling van de Gotcha Applicatie, waarbij ze de benodigde functionaliteit en prestaties leveren voor een succesvolle werking en een optimale gebruikerservaring.

Project information

Deze stageopdracht is gebaseerd op een spel dat tijdens de jaarlijkse DEV-Cruise van 2024 bij Euricom werd gespeeld. Het spel, dat veel enthousiasme opriep onder de deelnemers, draaide

om het elimineren van collega's met een Nerf-geweertje. Elke deelnemer ontving een mysterieuze enveloppe met de naam van een target die ze gedurende het weekend moesten uitschakelen door hem of haar te raken met een Nerf-pijltje. Om het spel eerlijk en overzichtelijk te houden, waren er enkele basisregels:

- Er mocht geen andere Euricommer binnen een straal van 3 meter zijn.
- Eliminaties waren niet toegestaan tijdens de sessies.
- Werd je uitgeschakeld, dan gaf je jouw target door aan degene die je had uitgeschakeld.

Hoewel het spel een groot succes was, kwamen er enkele verbeterpunten naar voren. Deze vormen de basis voor deze stageopdracht, waarin het doel is om een verbeterde versie van het spel te ontwikkelen.

Probleemstelling

Tijdens de eerste editie van het spel deden zich twee belangrijke problemen voor:

Zelftargeting en kleinere targetpool: Omdat de targets willekeurig werden verdeeld, werd de pool van beschikbare targets steeds kleiner. Hierdoor kregen sommige deelnemers uiteindelijk zichzelf als target toegewezen, ondanks dat er nog actieve spelers waren.

Vroege eliminatie: Sommige deelnemers werden erg snel uitgeschakeld – soms zelfs voordat ze hun eigen target kenden – waardoor zij nauwelijks betrokken waren bij het spel en de volledige ervaring misten.

Het doel van deze stageopdracht is om deze en andere verbeterpunten aan te pakken door een vernieuwde en verbeterde versie van het spel te ontwikkelen. Tijdens de stageperiode zal ik samen met mijn begeleiders werken aan oplossingen om het spel eerlijker, spannender en inclusiever te maken.

Projectafbakening en Risicoanalyse

Dit project heeft als doel een Progressive Web App (PWA) te ontwikkelen die het Gotcha-spel digitaliseert en optimaliseert. De applicatie automatiseert het toewijzen en volgen van targets, houdt eliminaties bij en biedt een live leaderboard, waardoor het spelproces efficiënter en gebruiksvriendelijker wordt.

Momenteel wordt het Gotcha-spel handmatig beheerd met enveloppen, wat leidt tot inefficiënties zoals dubbele targets en te snelle eliminaties. Er is geen digitale oplossing beschikbaar om dit proces te ondersteunen.

De PWA biedt een complete digitale ervaring en is succesvol afgerond wanneer deze:

- Targets willekeurig en zonder duplicaten of zelftoewijzing toewijst.
- Voor elke speler een unieke QR-code genereert voor eliminaties.
- Een live leaderboard toont met de status van alle actieve spelers.
- Notificaties verstuurt bij nieuwe targets en eliminaties.

- Offline toegankelijk is via caching (met behulp van een service worker).
- Veilig en betrouwbaar werkt via HTTPS.

Met deze innovatieve oplossing wordt het Gotcha-spel eenvoudiger te beheren, eerlijker en spannender voor alle deelnemers.

Afbakening van verantwoordelijkheden

Onder onze verantwoordelijkheid valt:

- Ontwerpen en ontwikkelen van de Gotcha PWA in React.
- Implementeren van de logica voor het genereren en scannen van QR-codes.
- Automatiseren van de targettoewijzing en voorkomen van dubbele of zelftoewijzing.
- Bouwen van een leaderboard dat real-time statusupdates toont.
- Inrichten van een service worker voor offline functionaliteit.
- Beveiliging van de applicatie via HTTPS.
- Documenteren van het systeem en opleveren van gebruiksinstructies.

Niet onder onze verantwoordelijkheid valt:

- Het bepalen van spelregels en wijzigingen tijdens het event.
- Hosting en het operationeel houden van de applicatie na oplevering.
- Inrichting en onderhoud van externe infrastructuur (zoals servers of CI/CD pipelines).
- Support en begeleiding tijdens het gebruik van de applicatie tijdens de DEV-Cruise.

Risico's onder de verantwoordelijkheid van de opdrachtgever

1. Onvoldoende infrastructuur of hostingomgeving:

- Risico: Als de opdrachtgever geen geschikte hostingomgeving biedt, is de applicatie niet toegankelijk voor gebruikers.
- Maatregel: De opdrachtgever zorgt voor een veilige, stabiele hostingomgeving met HTTPS-ondersteuning.

2. Gebrek aan gebruikersacceptatie:

- Risico: Als spelers de applicatie niet begrijpen of willen gebruiken, wordt het doel van het project niet behaald.
- Maatregel: De opdrachtgever organiseert een introductiemoment om de werking van de app uit te leggen aan de deelnemers.

3. Onverwachte technische problemen tijdens het event:

- Risico: Technische fouten of bugs kunnen het spel verstoren.

- Maatregel: De opdrachtgever zorgt voor een back-upplan (zoals handmatige targettoewijzing) als de applicatie faalt.

4. **Beheer na oplevering:**

- Risico: Zonder structureel onderhoud kunnen er bugs of beveiligingsrisico's ontstaan na verloop van tijd.
- Maatregel: De opdrachtgever is verantwoordelijk voor het onderhoud en de updates van de applicatie.

Voorgestelde maatregelen om risico's te beperken

- Regelmatige communicatie: Wekelijkse voortgangsbesprekingen om risico's tijdig te identificeren en te mitigeren.
- Teststrategie: Uitgebreide tests vooraf, inclusief gebruikerstests, om de stabiliteit van de applicatie te garanderen.
- Documentatie en overdracht: Duidelijke documentatie en overdracht van technische kennis aan de opdrachtgever.
- Fail-safe implementatie: Inbouwen van back-upmechanismen bij het uitvallen van kritieke functies zoals targettoewijzing.
- Flexibele logica: De targettoewijzing is dynamisch en kan tijdens het event worden aangepast indien nodig.

Deze aanpak zorgt ervoor dat de Gotcha-applicatie stabiel, gebruiksvriendelijk en schaalbaar is en minimaliseert de risico's voor zowel de ontwikkelaars als de opdrachtgever.

Requirements

Om het spel te verbeteren, zijn de volgende vereisten opgesteld. Deze zijn onderverdeeld in drie categorieën: 'Must Have', 'Should Have' en 'Would Have'.

Must Have

De Progressive Web App (PWA) voor het Gotcha-spel biedt een complete digitale ervaring die het spelbeheer automatiseert en optimaliseert. De applicatie is speciaal ontworpen om gebruiksvriendelijk, efficiënt en volledig afgestemd op de behoeften van Euricom te zijn.

Gebruikers kunnen de applicatie eenvoudig installeren en openen vanaf hun smartphone dankzij de PWA-functionaliteit. Het systeem bevat een automatische importfunctie waarmee alle Euricom-medewerkers moeiteloos als spelers toegevoegd kunnen worden. Voor veilige en snelle toegang loggen gebruikers in via Entra ID (Single Sign-On), waardoor aparte accounts overbodig zijn.

De interface is ontworpen in lijn met de Euricom-branding en biedt een logische, intuïtieve navigatie voor een prettige gebruikerservaring. In de standaard gamemode krijgt iedere speler willekeurig één target toegewezen. De achterliggende logica zorgt ervoor dat er uiteindelijk slechts één winnaar overblijft, wat de spanning tot het einde waarborgt.

Eliminaties worden geregistreerd door het scannen van een unieke QR-code van de target, wat zorgt voor een nauwkeurige en eerlijke voortgang van het spel. Een live leaderboard toont in real-time de status van alle spelers, inclusief overlevenden en scores, zodat iedereen de voortgang direct kan volgen.

Voor beheerders is er een uitgebreid adminpaneel beschikbaar waarmee het spel beheerd, spelers aangepast en indien nodig handmatig ingegrepen kan worden. Daarnaast bevat de applicatie duidelijke en overzichtelijke pagina's met de spelregels per gamemode, zodat deelnemers altijd de juiste informatie bij de hand hebben.

De Gotcha PWA is volledig gereed om ingezet te worden tijdens de volgende editie van de DEV-Cruise, waar het spel de deelnemers een spannende, gestroomlijnde en interactieve ervaring biedt.

Should have

De Gotcha PWA biedt diverse spelmodi en geavanceerde functies om een dynamische en meeslepende spelervaring te creëren. Spelers kunnen kiezen uit verschillende gamemodi, elk met unieke spelregels en uitdagingen.

In de Ranked-modus strijden spelers in een competitieve omgeving met een ELO-systeem, waarbij punten worden verdiend of verloren op basis van prestaties. De Free for All-modus biedt een alles-tegen-iedereen aanpak, waarin spelers zonder vaste targets strijden tot er één laatste overlevende is. Daarnaast is er een Custom-modus, waarin teams tegen elkaar kunnen spelen of een ladderstructuur gevolgd wordt voor meer flexibiliteit en maatwerk.

Het leaderboard biedt niet alleen een live-overzicht van de overgebleven spelers, maar toont ook uitgebreide statistieken zoals de Kill/Death/Assist-ratio (KDA), het aantal eliminaties en andere relevante prestaties. Aan het einde van elk spel ontvangen spelers een game roundup met leuke en interessante statistieken, zoals de snelste eliminatie, de langste overlevingsduur of de meeste gemiste kansen.

Beheerders hebben de mogelijkheid om custom spellen op te zetten met uitgebreide personalisatiemogelijkheden. Ze kunnen spelers uitnodigen voor specifieke spellen, eigen tijdslimieten instellen en extra regels toevoegen om de spelervaring verder te verfijnen.

Deze veelzijdige functies maken de Gotcha PWA niet alleen geschikt voor standaardspellen, maar ook voor aangepaste en competitieve formats, waardoor het spel altijd fris en uitdagend blijft.

Would have

De Gotcha PWA biedt een dynamische en meeslepende spelervaring met diverse extra functies die het spel spannender en interactiever maken.

Spelers ontvangen pushmeldingen op hun smartphone om op de hoogte te blijven van belangrijke gebeurtenissen. Of het nu gaat om het toewijzen van een nieuw target, de start van de finale, of wanneer ze geëlimineerd zijn—deze notificaties zorgen ervoor dat niemand iets mist.

Om het spel visueel aantrekkelijker te maken, bevat de applicatie animaties die belangrijke momenten extra benadrukken. Denk hierbij aan dynamische effecten bij eliminaties of een spectaculaire onthulling van de uiteindelijke winnaar, wat de spelbeleving nog intenser maakt.

Bij het opzetten van het spel zijn er diverse extra mogelijkheden beschikbaar om de spanning te verhogen en de spelervaring te verrijken. Zodra er nog 10 spelers over zijn, verdwijnt het overzicht van levende spelers, waardoor deelnemers niet meer kunnen zien wie er nog in het spel is. Dit voegt een extra laag mysterie en strategie toe in de eindfase.

Wanneer het aantal overlevenden een vooraf bepaald punt bereikt, wordt automatisch de finale aangekondigd. Deze fase introduceert aangepaste regels of extra uitdagingen om het einde van het spel nog intenser te maken en de strijd om de overwinning te verscherpen.

De applicatie is bovendien voorbereid op toekomstige uitbreidingen en dynamische regels, waardoor er eenvoudig nieuwe functies kunnen worden toegevoegd. Dit biedt de flexibiliteit om het spel continu te vernieuwen en aan te passen aan verschillende speelstijlen en evenementen.

Met deze geavanceerde functionaliteiten biedt de Gotcha PWA een innovatieve, interactieve en spannende spelervaring die zich steeds verder kan ontwikkelen.

Game Logica

Hier ga ik de verschillende game Modes beschrijven en wat deze precies inhouden:

Game mode 1

We moeten ervoor zorgen dat spelers in een cirkelvorm aan een target kunnen worden gekoppeld en dat er een maximale tijdslimiet voor de game wordt ingesteld. Ons idee is om bij het aanmaken van een game direct een begin- en eindtijd/datum in te voeren. Dit maakt het eenvoudiger om alles vooraf te plannen voor een evenement, in plaats van dit pas te regelen op het moment dat de game begint.

Deze tijdslimiet bepaalt de duur van het spel. Omdat er altijd onvoorziene omstandigheden kunnen zijn waardoor een game eerder moet eindigen, krijgt de admin de mogelijkheid om de game op elk moment te stoppen.

Om elke game vernieuwend te houden, willen we dit proces willekeurig maken. Elke keer dat een speler zijn target elimineert, krijgt hij de target van de geëlimineerde persoon. Hierbij kan echter een probleem ontstaan: een speler kan vóór het einde van het spel zijn eigen naam krijgen.

Om dit te voorkomen, hebben we het volgende bedacht:

Aan het begin nemen we alle gebruikers op in een unieke array, in de volgorde waarin we ze van de API ontvangen (waarschijnlijk op basis van hun werktijd bij Euricom). Om de verdeling volledig willekeurig te maken, vragen we eerst om een willekeurig getal tussen 1 en 10. Vervolgens gebruiken we dit getal in een for-loop. De reden hiervoor is om de willekeurigheid verder te vergroten.

Binnen deze for-loop krijgt elke speler een willekeurige plaats in de array. Dit wordt gedaan met de Fisher-Yates-Shuffle-methode.

Wanneer alle spelers op een willekeurige positie in de array zijn gezet, kennen we de targets toe. Dit gebeurt door simpelweg de volgende speler in de gerandomiseerde array als target te kiezen. Hierdoor is het nooit mogelijk om jezelf als target te krijgen, behalve wanneer het spel is afgelopen.

- Random generatie
 - Je kunt de random-functie van Math gebruiken, maar deze is niet volledig willekeurig en de mate van willekeur hangt af van de browser.
 - Random-js is een andere optie. Dit is een library die kan worden gebruikt om willekeurige getallen te genereren.
- Fisher-Yates-Shuffle-methode

Dit is een methode om een array op een willekeurige manier te schudden. Dit gebeurt door in een aflopende for-loop over elk element van de array te itereren en items op een willekeurige plek te plaatsen.

- Dit kan worden gedaan met de Math-library van React. Hiervoor heb je de random- en floor-functie nodig om alles correct en willekeurig te laten verlopen.
- Je kunt ook de Random-js-library gebruiken. Hiervoor heb je enkel de integer-functie van deze library nodig.

- Toekenning

Begin:	
1.	Bob
2.	Alice
3.	Leo
4.	Jef

Random array:	
1.	Leo
2.	Alice
3.	Jan
4.	Bob

Target:			
1.	Bob	→	Jef
2.	Alice	→	Jan
3.	Leo	→	Alice
4.	Jef	→	Leo

Game mode 2

Deze game mode draait om het **Elo-ratingsysteem**, waarbij spelers met een hogere rating vaker als doelwit worden gekozen dan spelers met een lagere rating. Dit systeem zorgt voor een gebalanceerde en uitdagende spelervaring waarin sterke spelers meer in de schijnwerpers staan, terwijl minder ervaren spelers minder vaak doelwit worden.

Elo-rating: Wat is het en hoe werkt het?

De **Elo-rating** is een getalsmatige aanduiding van de sterkte van een speler. Elke speler krijgt een score die hun prestaties weerspiegelt: hoe beter je speelt, hoe hoger je rating. Dit systeem maakt het mogelijk om spelers met elkaar te vergelijken en de beste deelnemers te identificeren.

Om de Elo-rating te berekenen, hanteren we de volgende formules:

Nieuwe Elo-rating berekenen:

$$R_{\text{New}} = R_{\text{Old}} + K(W - E) \quad R_{\text{New}} = R_{\text{Old}} + K(W - E)$$

- **RNew** = Nieuwe rating na de game
- **ROld** = Huidige rating vóór de game

- **K** = Correctiefactor die bepaalt hoe snel de rating verandert (in deze game mode ingesteld op **20**)
- **W** = Werkelijk resultaat (1 bij winst, 0.75 bij top 25%, 0.5 bij het midden, 0 bij de laagste 25%)
- **E** = Verwachte winstkans

2. Kans om te winnen berekenen:

$$E = \frac{1}{1 + 10^{\frac{(R2 - R1)}{400}}} \quad E = \frac{1}{1 + 10^{\frac{(R2 - R1)}{400}}}$$

- **E** = Verwachte kans om te winnen
- **R1** = Jouw Elo-rating
- **R2** = De hoogste Elo-rating van je tegenstanders

Aanpassingen voor de Gotcha Game Mode

Hoewel we de standaardformules hanteren, passen we de manier waarop waarden worden toegekend aan om het spel eerlijker en dynamischer te maken:

- **K-factor:** We gebruiken een **K-waarde van 20** om een gebalanceerde snelheid van ratingaanpassing te behouden. Dit kan indien nodig worden aangepast.
- **Werkelijk resultaat (W):**
 - **1.00** voor de winnaar (1e plaats)
 - **0.75** voor de top **25%** van de spelers
 - **0.50** voor spelers in het **middengebied**
 - **0.00** voor de laagste **25%**
- **Tegenstanders rating (R2):** In plaats van één specifieke tegenstander te gebruiken, nemen we de **hoogste rating** in het spel om de uitdaging te vergroten.

Spelverloop en Mechanieken

In deze game mode blijven alle spelers actief gedurende het hele spel, zelfs na eliminatie. Wanneer een speler een tegenstander uitschakelt, krijgt de eliminator een **nieuw target**, terwijl het oorspronkelijke target behouden blijft voor de geëlimineerde speler.

Om te voorkomen dat het spel eindeloos doorgaat, zijn er twee manieren om het spel te beëindigen:

1. **Tijdslimiet:** Bij het opzetten van de game kan de admin een tijdslimiet instellen (van enkele uren tot meerdere dagen).
2. **Handmatige beëindiging:** De admin heeft te allen tijde de mogelijkheid om de game handmatig te stoppen.

Daarnaast kan bij het aanmaken van de game een **startdatum** worden gekozen, waardoor deze mode perfect geschikt is voor geplande evenementen zoals de **DEV-Cruise**.

Target Toewijzing: Dynamische Array Methode

Bij de start van de game krijgt elke speler een **initiële Elo-rating van 1000**. Om de kans op het krijgen van een target te beïnvloeden, maken we gebruik van een **dynamische array**:

- Spelers met een hoge rating verschijnen **vaak** in de array, waardoor ze een grotere kans hebben om als target gekozen te worden.
- Spelers met een lage rating komen **minder vaak** in de array voor en worden dus minder snel als target geselecteerd.

Drempelwaarden voor de array:

- **< 700 rating:** Niet opgenomen in de array (niet beschikbaar als target).
- **700 – 1299 rating:** 1 keer in de array.
- **≥ 1300 rating:** 3 keer in de array.

Target-selectieproces:

1. De array wordt continu bijgewerkt op basis van de prestaties van de spelers.
2. Bij targettoewijzing wordt eerst het **eigen ID** uit de array gefilterd.
3. Vervolgens wordt een **willekeurig getal** gegenereerd om een target te kiezen uit de overgebleven array.

Scoresysteem tijdens het Spel

Elke speler begint de game met een **basis score van 10**. Gedurende het spel wordt deze score aangepast op basis van prestaties:

- **+2 punten** per eliminatie (kill).
- **-2 punten** bij eigen eliminatie (death).

Speciale score-aanpassingen:

- **Elke 6 punten stijging:** De speler verschijnt **extra** in de array (grotere kans om gekozen te worden).
- **Elke 6 punten daling:** De speler verschijnt **minder vaak** in de array.
- **Minimale score = 0:** Spelers kunnen geen negatieve score hebben. Bij **0 punten** worden ze volledig uit de array verwijderd.

Winnaar Bepalen

Aan het einde van de game wordt de winnaar bepaald op basis van de **Elo-rating** en de behaalde **score**. De uiteindelijke rangschikking wordt gebruikt om de nieuwe Elo-rating van elke speler te berekenen, waarbij de eerder besproken formules worden toegepast.

Deze aanpak zorgt ervoor dat het spel zowel competitief als eerlijk blijft, terwijl het dynamische karakter van de array ervoor zorgt dat sterke spelers een grotere uitdaging hebben en zwakkere spelers minder snel overbelast raken.

Samenvatting van de Game Mode

Het Elo-systeem bepaalt de sterkte van elke speler en past zich dynamisch aan op basis van hun prestaties. Door gebruik te maken van een dynamische array worden spelers met een hogere rating vaker als target gekozen, wat zorgt voor een extra uitdaging. Om te voorkomen dat het spel eindeloos doorgaat, biedt deze game mode een flexibele tijdslimiet en heeft de admin de mogelijkheid om de game op elk moment te stoppen. Het scoresysteem belooft spelers voor eliminaties en straft verliezen, waarbij de aanwezigheid in de array wordt aangepast bij elk veelvoud van zes punten. Uiteindelijk wordt de winnaar bepaald op basis van de Elo-rating en de behaalde prestaties aan het einde van het spel. Deze geavanceerde, Elo-gebaseerde Gotcha mode biedt een spannende, eerlijke en strategische spelervaring die zich voortdurend aanpast aan de vaardigheden van de spelers.

Front-end

vergelijken we **Vue** en **React**, twee populaire front-end frameworks, om de beste keuze te maken voor de ontwikkeling van onze gebruikersinterface. We gaan dieper in op hun kenmerken, voordelen en nadelen om te bepalen welk framework het beste past bij de behoeften van ons project.

Vue

Vue.js is een progressief JavaScript-framework dat wordt gebruikt voor het bouwen van gebruikersinterfaces. Dankzij de modulaire architectuur is het zowel geschikt voor kleine projecten als voor grootschalige applicaties. Het framework maakt gebruik van een declaratieve syntaxis, wat betekent dat je eenvoudige, op HTML gebaseerde sjablonen kunt schrijven. Door de reactieve benadering worden wijzigingen in data automatisch gesynchroniseerd met de interface, met behulp van concepten zoals `ref` en `reactive`. Vue is volledig component-gebaseerd, waardoor je applicaties kunt opbouwen uit herbruikbare en goed te onderhouden onderdelen.

Daarnaast biedt Vue officiële pakketten zoals Vue Router voor het beheren van routes en Vuex voor state management, wat helpt bij het structureren van complexe applicaties. Een ander voordeel van Vue is dat het lichtgewicht is, wat resulteert in kleinere bundelgroottes in vergelijking met sommige andere frameworks.

Voordelen:

- Makkelijk te leren, vooral voor mensen met HTML- en JavaScript-ervaring.
- Duidelijke documentatie en sterke community.
- Geschikt voor zowel SPA's (Single Page Applications) als kleinere integraties.

Nadelen:

- Minder ondersteuning bij zeer grote enterprise-applicaties vergeleken met React.
- Kleiner ecosysteem dan React.

Wanneer kies je voor Vue?

Vue is een uitstekende keuze wanneer je snel een prototype of MVP wilt ontwikkelen. Het is

ideaal voor kleinere tot middelgrote applicaties die eenvoud en gebruiksgemak vereisen. Dankzij de lage leercurve en flexibele architectuur biedt Vue een efficiënte en toegankelijke oplossing voor diverse projecten.

React

React is een JavaScript-bibliotheek, ontwikkeld door Facebook (nu Meta), die wordt gebruikt om dynamische en interactieve gebruikersinterfaces te bouwen. Het maakt gebruik van een component-based architectuur, waarbij applicaties worden opgebouwd uit herbruikbare componenten, en het Virtual DOM, wat zorgt voor efficiënte updates en optimale prestaties.

Een opvallend kenmerk van React is JSX, een syntactische uitbreiding die het mogelijk maakt om HTML-achtige structuren direct in JavaScript te schrijven. React biedt flexibele oplossingen voor state management, zoals de ingebouwde useState en useReducer hooks, of externe bibliotheken zoals Redux en Zustand. Voor client-side navigatie maakt React gebruik van het officiële React Router-pakket, wat het eenvoudig maakt om dynamische routes en views te beheren.

Voordelen:

- Flexibiliteit door uitgebreide bibliotheekondersteuning (bijv. Next.js voor SSR/SSG).
- Grote community en uitgebreide tooling.
- Zeer geschikt voor complexe en grootschalige applicaties.

Nadelen:

- Steilere leercurve, vooral door concepten zoals hooks, context en component-lifecycle.
- Boilerplate kan toenemen bij grotere applicaties.

Wanneer kies je voor React?

React is ideaal voor grote, complexe applicaties die schaalbaar moeten zijn en waarbij performance en componenthergebruik prioriteit hebben. Het is ook een goede keuze als je flexibiliteit wilt in je technologiekeuzes, zoals ondersteuning voor Server-Side Rendering (SSR), Static Site Generation (SSG), Single Page Applications (SPA) of hybride apps. Dankzij de krachtige architectuur en brede ondersteuning is React geschikt voor zowel kleine projecten als enterprise-oplossingen.

Eind Conclusie

Na het vergelijken van **Vue** en **React** voor dit project, heb ik ervoor gekozen om **React** te gebruiken. React biedt de flexibiliteit die we nodig hebben voor de complexiteit van de applicatie, vooral gezien de verschillende game modi, live updates van het leaderboard en de noodzaak voor schaalbaarheid en prestaties. De grote community en uitgebreide bibliotheken die React ondersteunt, zorgen ervoor dat we snel en efficiënt kunnen ontwikkelen. Daarnaast stelt de Virtual DOM React in staat om dynamische content snel en zonder vertraging weer te geven, wat essentieel is voor een interactieve game-ervaring zoals we voor ogen hebben.

Met de groeiende eisen van het project en de mogelijkheid om verschillende functionaliteiten toe te voegen (zoals custom games en notificaties), is React de beste keuze om de applicatie robuust en toekomstbestendig te maken.

Back-end

Voor de backend van deze applicatie hebben we gekozen voor **.NET**. .NET is een krachtig en veelzijdig framework ontwikkeld door Microsoft, ideaal voor het bouwen van schaalbare en veilige webapplicaties. Er zijn verschillende redenen waarom we voor .NET hebben gekozen voor dit project:

.NET

.NET is een uitstekende keuze voor de backend van deze applicatie vanwege de naadloze integratie met Microsoft-producten. Binnen de Euricom-omgeving is het cruciaal om te koppelen met tools zoals Entra ID voor Single Sign-On en andere Azure-diensten. Dankzij de uitgebreide ondersteuning van .NET voor deze technologieën kunnen we eenvoudig integreren met de bestaande infrastructuur en authenticatiemethoden, wat zorgt voor een veilige en efficiënte werkomgeving.

Daarnaast staat .NET bekend om zijn betrouwbaarheid en hoge prestaties, wat essentieel is voor een applicatie die live updates moet verwerken, zoals een dynamisch leaderboard en game-statistieken. De geoptimaliseerde architectuur maakt het mogelijk om de applicatie efficiënt te schalen naarmate het aantal gebruikers groeit, zonder in te boeten op snelheid of stabiliteit.

Beveiliging is een ander belangrijk aspect, vooral omdat de applicatie werkt met gevoelige gegevens zoals bedrijfsinformatie en persoonlijke gegevens van medewerkers. .NET biedt ingebouwde tools en best practices voor het implementeren van veilige verbindingen, authenticatie en autorisatie, waardoor de applicatie voldoet aan moderne beveiligingsstandaarden en veilig kan draaien in een productieomgeving.

Een ander voordeel is de kracht en veelzijdigheid van de programmeertaal C#, waarmee we zowel de game-logica als de administratieve functies, zoals het beheren van spelers en game-instellingen, efficiënt kunnen ontwikkelen. De robuustheid van C# verhoogt de productiviteit en maakt het mogelijk om complexe functionaliteiten op een gestructureerde en schaalbare manier te implementeren.

Bovendien profiteert .NET van uitgebreide ondersteuning en een actieve community. Microsoft biedt gedetailleerde documentatie, en de grote ontwikkelaarsgemeenschap draagt voortdurend bij aan het ecosysteem. Hierdoor hebben we altijd toegang tot de nieuwste resources, bibliotheken en best practices, wat de ontwikkeling versnelt en optimaliseert.

Kortom, .NET biedt een solide, betrouwbare en schaalbare basis voor de backend van deze applicatie. Het past perfect bij de bestaande technologische stack van Euricom en stelt ons in staat om zowel de gamefunctionaliteiten als de administratieve en beveiligingsvereisten van het project effectief te implementeren.

Databank

Microsoft SQL Server (MSSQL) is een krachtig relationeel databasebeheersysteem (RDBMS) ontwikkeld door Microsoft. Het wordt breed ingezet voor bedrijfsdatabases, datawarehousing en analytische toepassingen.

Belangrijkste kenmerken

SQL Server biedt krachtige en veelzijdige functies voor databeheer en -analyse. Een belangrijk onderdeel hiervan is T-SQL (Transact-SQL), een uitgebreide SQL-taal die ondersteuning biedt voor het uitvoeren van complexe bewerkingen via stored procedures, triggers en transacties. Deze database is ACID-compliant, wat betekent dat het voldoet aan de vier kernprincipes: atomicity, consistency, isolation en durability. Dit garandeert de betrouwbaarheid en integriteit van gegevens, zelfs bij fouten of systeemcrashes.

Daarnaast blinkt SQL Server uit in schaalbaarheid en prestaties door functies zoals indexering, partitionering en in-memory verwerking. Deze optimalisaties zorgen voor efficiënte data-analyse en snelle query-uitvoering, zelfs bij grote datasets.

Op het gebied van beveiliging biedt SQL Server geavanceerde functionaliteiten om gevoelige gegevens te beschermen. Functies zoals encryptie, Row-Level Security en Dynamic Data Masking zorgen ervoor dat alleen geautoriseerde gebruikers toegang hebben tot specifieke informatie en dat gevoelige data afgeschermd blijft.

Voor gegevensanalyse en rapportage integreert SQL Server naadloos met krachtige analysetools zoals Power BI, SQL Server Reporting Services (SSRS) en SQL Server Analysis Services (SSAS). Deze tools maken het mogelijk om diepgaande analyses uit te voeren en gedetailleerde rapporten te genereren, wat bijdraagt aan betere besluitvorming en datagedreven inzichten.

Voordelen

- Goede integratie met Microsoft-producten zoals Azure, Power BI en Active Directory.
- Geschikt voor zowel kleine als grote toepassingen door de schaalbaarheid en prestaties.
- Robuuste beveiligingsfuncties voor de bescherming van gevoelige data.
- Ondersteuning voor complexe queries en dataverwerking met geavanceerde functies zoals partitioning en in-memory tabellen.

Nadelen

- Licentiekosten kunnen hoog zijn, vooral bij gebruik in grootschalige omgevingen.
- Vereist aanzienlijke hardwarebronnen voor optimale prestaties bij grote datasets.
- Complexer in beheer en configuratie dan sommige open-source alternatieven.

Alternatieven

- MySQL: Open-source en geschikt voor kleinere tot middelgrote toepassingen, met name populair bij webapplicaties.
- PostgreSQL: Open-source, krachtig en geschikt voor complexe queries en geavanceerde datatypes.

- MongoDB: Een NoSQL-database die geschikt is voor ongestructureerde data en flexibele datamodellen.

Wanneer kiezen voor Microsoft SQL Server?

We kiezen voor Microsoft SQL Server omdat het naadloos integreert met Azure en andere Microsoft-diensten, wat het beheer en de implementatie binnen onze bestaande infrastructuur aanzienlijk vereenvoudigt. Aangezien we al gebruikmaken van Azure, biedt SQL Server ons directe ondersteuning voor cloudgebaseerde oplossingen zoals Azure SQL Database en Azure Data Services, wat schaalbaarheid en flexibiliteit vergroot.

Daarnaast staat SQL Server bekend om zijn betrouwbaarheid en prestaties, wat essentieel is voor toepassingen die grote hoeveelheden data verwerken en waarbij real-time verwerking vereist is. De geavanceerde beveiligingsfuncties, zoals encryptie, Row-Level Security en Dynamic Data Masking, zorgen ervoor dat gevoelige bedrijfsinformatie beschermd blijft en voldoen aan strikte compliance-eisen.

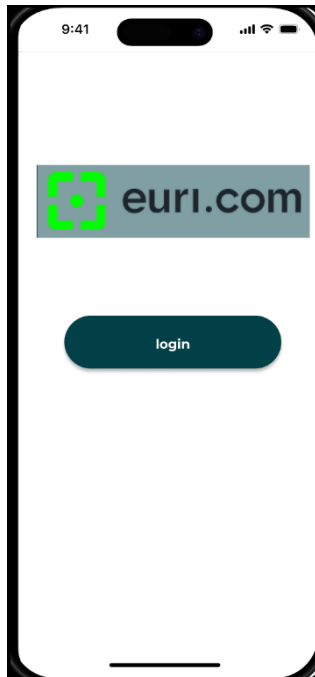
Een ander belangrijk voordeel is de uitgebreide ondersteuning en tooling. Microsoft biedt niet alleen uitstekende documentatie en technische ondersteuning, maar ook krachtige analysetools zoals SQL Server Management Studio (SSMS) en de integratie met Power BI voor diepgaande rapportage en datavisualisatie. Dit stelt ons in staat om efficiënter te werken en sneller inzichten te verkrijgen.

Kortom, SQL Server is voor ons de logische keuze vanwege de eenvoudige integratie met Azure, de hoge prestaties, de robuuste beveiliging en de uitgebreide ondersteuning, waardoor het perfect aansluit bij onze bestaande technologie en toekomstige groeibehoeften.

Prototypes

Hieronder worden de prototypes van de applicatie getoond. Deze prototypes geven een visueel overzicht van de belangrijkste functies en interacties binnen de applicatie, en dienen als referentiepunt voor verdere ontwikkeling en optimalisatie.

Login



De loginpagina bevat een eenvoudige knop met het label "Login". Wanneer medewerkers hierop klikken, worden ze doorgestuurd naar de Entra ID-authenticatiepagina van Microsoft voor verificatie. Na succesvolle inlog via Entra ID worden ze automatisch doorgestuurd naar de homepagina.

Home



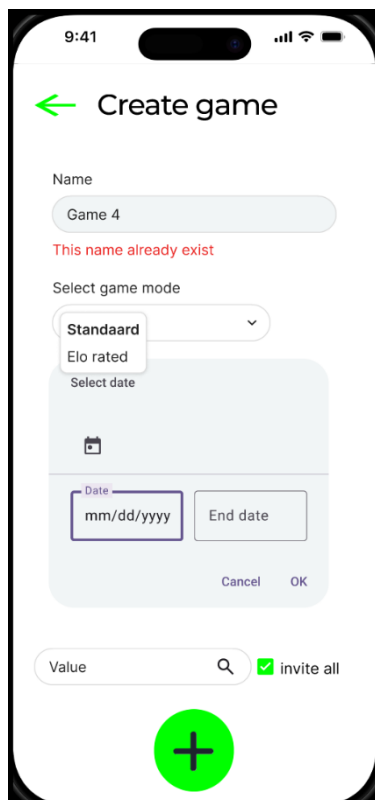
De eerste pagina na het inloggen toont een overzicht van alle games waaraan de gebruiker deelneemt of heeft deelgenomen. Elke game wordt weergegeven als een interactieve lijstitem met de naam en status van de game.

Wanneer de gebruiker op een game klikt, wordt hij doorgestuurd naar de detailpagina van die specifieke game.

Rechts onderaan bevindt zich een plusknop waarmee de gebruiker eenvoudig een nieuwe game kan aanmaken. Door hierop te klikken, wordt een nieuw creatieproces gestart waarin de gebruiker gegevens zoals de gamenaam en instellingen kan invoeren.

De interface is intuïtief en gericht op snelle navigatie tussen bestaande games en het opstarten van nieuwe games.

Create Game



Wanneer de gebruiker op het plusje (+) drukt, wordt hij doorgestuurd naar de pagina om een nieuwe game aan te maken.

Bovenaan deze pagina kan de gebruiker een gamemode selecteren uit een lijst met beschikbare speltypen. Hieronder zijn er invoervelden of een datumkiezer om de start- en einddatum van de game te bepalen.

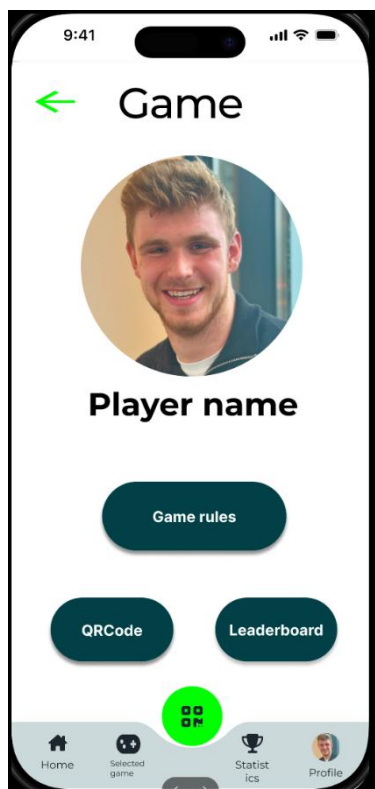
Onder deze velden is er een sectie om deelnemers uit te nodigen. Hier heeft de gebruiker twee opties:

Invite All-knop: Hiermee nodigt de gebruiker in één keer alle Euicommers uit.

Individuele zoekfunctie: De gebruiker kan specifieke collega's zoeken via een zoekbalk en hen handmatig selecteren om uit te nodigen.

Zodra alle keuzes zijn gemaakt, is er een Bevestigen-knop onderaan om de game definitief aan te maken en deelnemers uit te nodigen.

Geselecteerde game



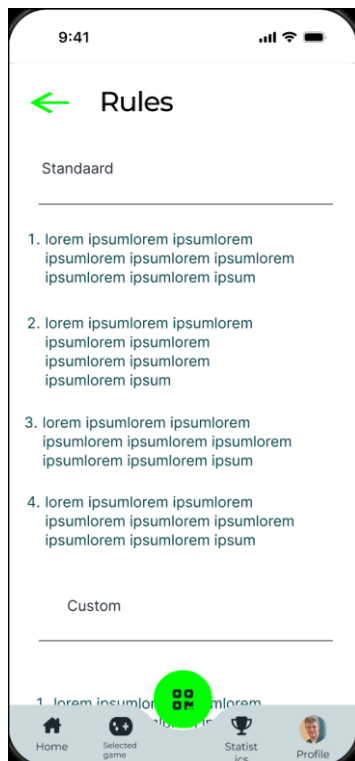
De geselecteerde game toont verdere details zoals de regels van de game, een QR-code en het leaderboard. Als je op de knop QR-Code drukt, ga je naar de pagina waar je jouw QR-code kunt laten zien, zodat je moordenaar deze kan scannen. Bij het leaderboard ga je naar de game statistics.

Onderaan zie je een navigatiebalk met vier tabs:

1. Home tab: Hiermee ga je terug naar de homepage.
2. Selected game tab: Dit is de huidige tab die je bekijkt.
3. Statistieken tab: Hier vind je de spelerstatistieken en de game statistieken.
4. Profile tab: Hier staan de beheerpagina's, achievements en invites.

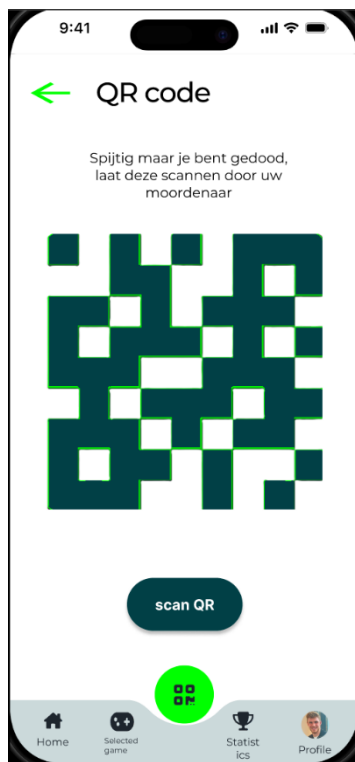
De groene knop leidt je snel naar de QR-code pagina, zodat deze makkelijk bereikbaar is.

Game rules

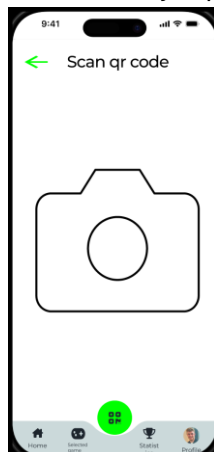


Hier worden de regels getoond van een bepaalde game.

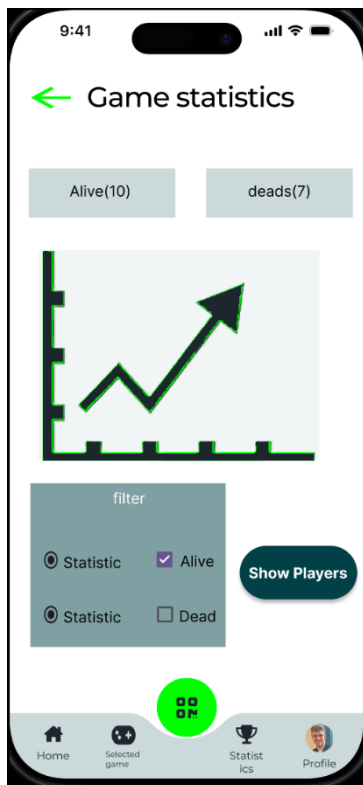
QR-Code page



De QR-code in het midden van de pagina kan snel en gemakkelijk gescand worden door je moordenaar. Als je zelf je target wilt scannen, kun je op de knop Scan QR drukken. Deze knop opent de camera van je apparaat, zodat je een QR-code kunt scannen.



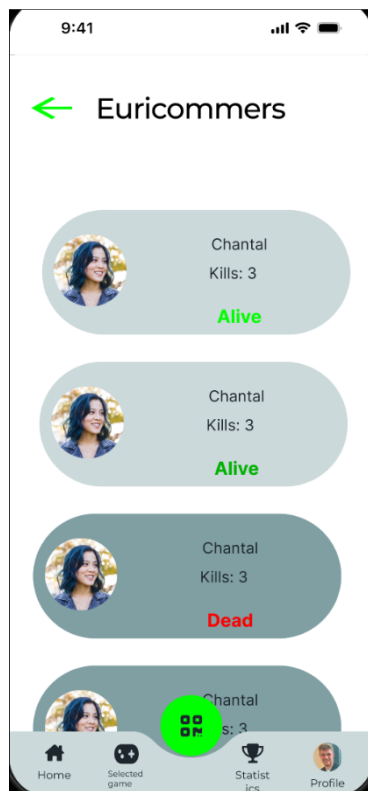
Game Statistieken



Op deze pagina vind je alle statistieken van een bepaalde game, zoals het aantal mensen dat nog leeft en het aantal dat al dood is. Je kunt filteren op specifieke gegevens via de filtersectie, waardoor de grafiek zich aanpast op basis van de gekozen criteria.

Door op de knop Show players te drukken, ga je naar een lijst van spelers die nog levend zijn of al dood zijn.

Speler gegevens per game



Op deze pagina worden alle spelers getoond die nog levend zijn of dood, samen met het aantal targets dat elke speler heeft geraakt. De spelers worden gesorteerd op hun status, en je kunt snel zien wie actief is en wie niet, evenals hun prestaties in de game.

Persoonlijke Statics



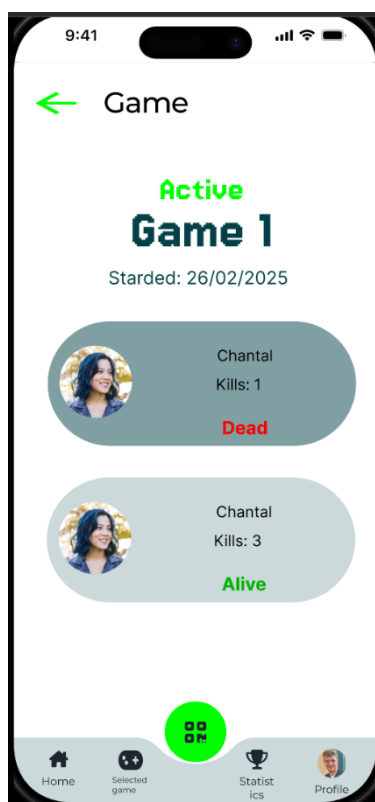
Je persoonlijke gegevens als speler worden weergegeven, zoals het aantal targets dat je hebt geraakt, hoeveel keer je bent dood gegaan, en nog veel meer. Je kunt de statistieken filteren op specifieke gegevens via de filtersectie om bepaalde periodes of prestaties te bekijken. Onderaan de pagina kun je je behaalde achievements zien, die je voortgang en successen in de game weergeven.

Admin Home Page



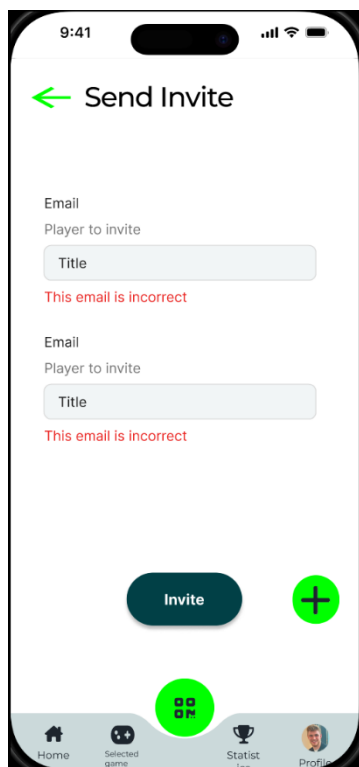
Hier zie je twee knoppen. De eerste knop stelt de admin in staat om alle spelers in zijn game te bekijken, zodat hij een overzicht heeft van wie er deelneemt. De tweede knop geeft de admin de mogelijkheid om nieuwe spelers uit te nodigen voor de game, mocht hij iemand vergeten zijn.

Admin Toon spelers in zijn game



Op deze pagina zie je de gamegegevens, zoals wanneer de game is gestart en of deze nog actief is. Je kunt ook zien welke spelers deelnemen aan de game en wie er nog levend is en wie niet. Dit geeft een overzicht van de voortgang en de status van de spelers in de game.

Admin invite speler



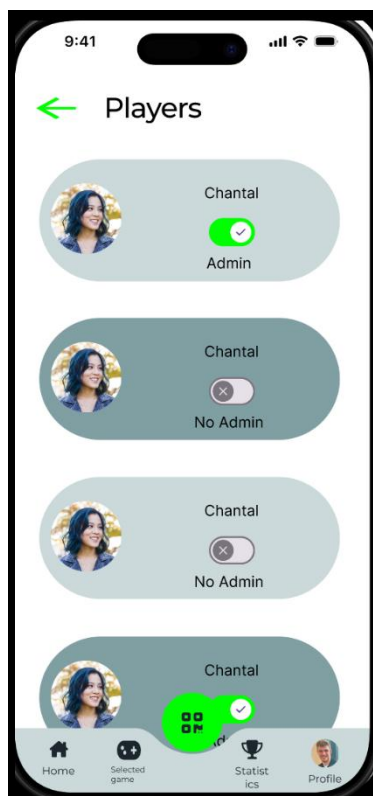
Hier kan men de namen ingeven van de mensen dat men wilt invite. Als men op het groene plusje drukt komt er elke keer een extra input field waar men een user kan ingeven als men alle users heeft die hij wilt invite kan hij op de onderste knop drukken en de mensen invite.

Super Admin



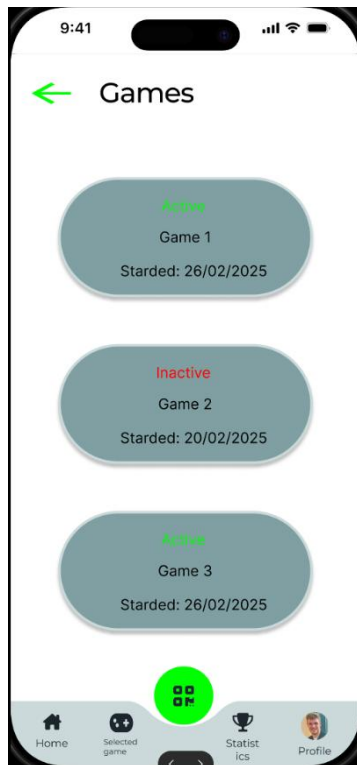
Hier ziet men het menu van de super admin, waar verschillende tabs beschikbaar zijn. De tabs omvatten Show all players, waarmee alle spelers in het systeem kunnen worden bekeken, en Show all games, die een overzicht biedt van alle actieve en afgeronde games. De super admin kan ook nieuwe achievements toevoegen via de Add achievements tab, en nieuwe regels instellen of bestaande regels aanpassen door respectievelijk de tabs Add rules en Change rules te gebruiken. Dit menu biedt de super admin volledige controle over de game-instellingen en spelergegevens.

Show all players

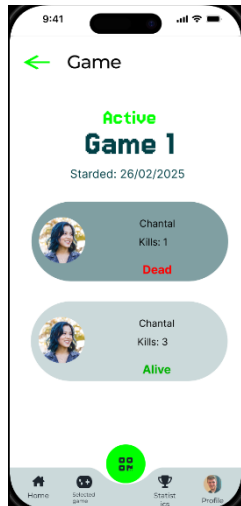


Op deze pagina worden alle spelers die in het systeem zitten weergegeven. Onder elke speler is er een shift-knop waarmee je deze speler admin kunt maken. In de toekomst zou er wellicht een delete-knop of bewerk-knop toegevoegd kunnen worden, waarmee je de gegevens van een speler kunt verwijderen of bewerken. Dit biedt de beheerder meer controle over de spelerinformatie.

All Games

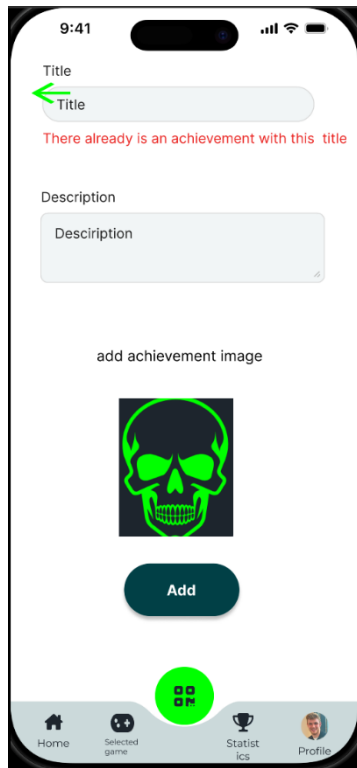


Op deze pagina staan **alle games** die ooit zijn aangemaakt, met een aanduiding of ze **actief** of **inactief** zijn, zodat je kunt zien of ze nog bezig zijn of al beëindigd. Door op een game te klikken,



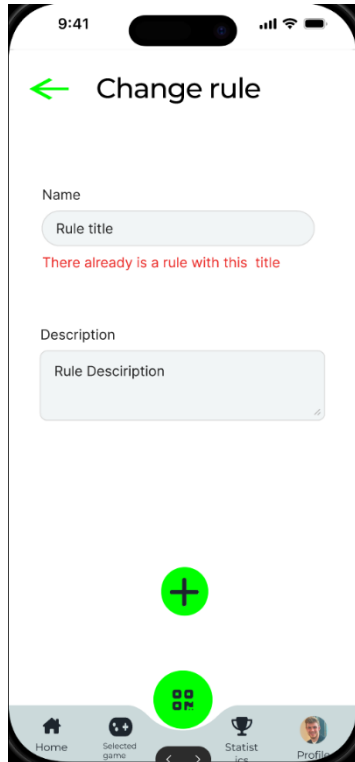
ga je naar de **game detail page**, waar je specifieke details van de game kunt bekijken, zoals welke spelers er nog leven en welke niet meer.

Add achievements



Op deze pagina kan de super admin nieuwe achievements toevoegen die spelers kunnen verdienen. De admin kan de naam, beschrijving en voorwaarden van de achievement instellen, zodat spelers deze kunnen behalen tijdens hun deelname aan de games.

Add rules



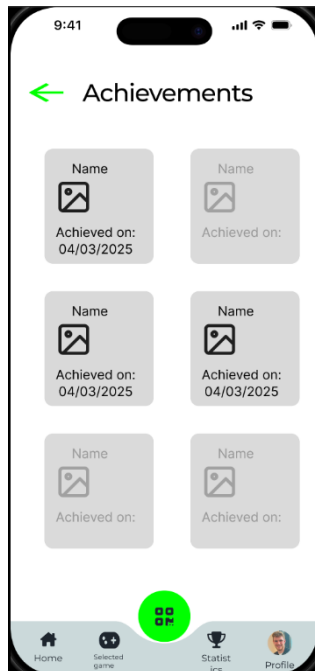
Op deze pagina kan een super admin custom rules maken en nieuwe regels toevoegen voor verschillende game modes. Hier kan de admin regels aanpassen, zoals het aantal spelers, tijdslimieten, en andere game-specifieke instellingen. De Change rules pagina zal er hetzelfde uitzien als deze, maar in plaats van Add zal het Edit zijn, zodat bestaande regels kunnen worden aangepast of bewerkt.

Invite page



Op deze pagina krijgen de spelers die uitgenodigd zijn voor een game hun uitnodiging te zien. Ze kunnen de uitnodiging accepteren door op het vinkje te klikken, of weigeren door het kruisje te selecteren. Hiermee kunnen ze aangeven of ze willen deelnemen aan de game of niet.

Achievements



Op deze pagina kan een speler zijn behaalde achievements bekijken, evenals de achievements die hij nog niet heeft behaald. Het geeft een overzicht van zijn voortgang en prestaties in de game.

Authentication / login

Login met Microsoft (Entra ID, SSO) via MSAL.js

Voor de authenticatie in de applicatie maken we gebruik van Single Sign-On (SSO) via Entra ID (voorheen Azure Active Directory). Dit zorgt ervoor dat gebruikers eenvoudig kunnen inloggen met hun Microsoft-account.

Om deze integratie te realiseren, gebruiken we MSAL.js (Microsoft Authentication Library for JavaScript), een officiële library van Microsoft. Deze library biedt ondersteuning voor veilige authenticatie en kan worden gebruikt in zowel React, Vue als .NET.

Voordelen van deze aanpak:

- **Gebruiksgemak:** Gebruikers hoeven slechts één keer in te loggen via hun bestaande Microsoft-account.
- **Beveiliging:** Ondersteunt moderne beveiligingsstandaarden zoals OAuth 2.0, OpenID Connect, en Multi-Factor Authenticatie (MFA).
- **Compatibiliteit:** Werkt naadloos samen met React, Vue en .NET, wat consistentie biedt in de volledige applicatie.
- **Toegang tot Microsoft-diensten:** Hiermee kunnen we indien nodig ook extra Microsoft-services benaderen, zoals de Microsoft Graph API voor profielgegevens of andere bedrijfsinformatie.

Door gebruik te maken van MSAL.js zorgen we voor een veilige, efficiënte en moderne authenticatieoplossing die eenvoudig te integreren is in de applicatie en voldoet aan de vereisten voor SSO via Entra ID.

Unieke QR-code

Voor het elimineren van spelers in het spel maken we gebruik van QR-codes. Elke speler krijgt een unieke QR-code die gescand kan worden bij eliminatie. Dit maakt het proces eenvoudiger, sneller en voorkomt fouten bij het doorgeven van targets.

Om deze QR-codes te genereren, gebruiken we de `qrcode.react`-library. Dit is een veelgebruikte en betrouwbare oplossing binnen React om QR-codes te maken en te tonen.

Hoe maken we de QR-code uniek?

Elke speler ontvangt een QR-code die gekoppeld is aan een **unieke identificatie**. Deze identificatie kan bestaan uit een gebruikers-ID, een willekeurig gegenereerde code (**UUID**).

Bij het scannen van de QR-code wordt deze unieke code doorgestuurd naar de backend. Daar wordt gecontroleerd of de eliminatie geldig is en wordt een nieuw target toegewezen. Om te voorkomen dat spelers zichzelf als target krijgen of dat één speler door meerdere mensen wordt achtervolgd, wordt hier extra logica toegepast.

Waarom kiezen voor `qrcode.react`?

Het gebruik van `qrcode.react` biedt meerdere voordelen:

- **Gebruiksgemak:** QR-codes kunnen eenvoudig gegenereerd en weergegeven worden in de applicatie.
- **Unieke identificatie:** Elke speler heeft een persoonlijke code, waardoor fouten in het spel worden voorkomen.
- **Snelheid en efficiëntie:** Spelers kunnen direct een eliminatie bevestigen door een QR-code te scannen.
- **Veiligheid:** De QR-code bevat enkel een unieke code die gecontroleerd wordt op de backend, wat manipulatie voorkomt.

Deze aanpak zorgt voor een gestroomlijnd proces waarin eliminaties nauwkeurig worden geregistreerd en het spelverloop vlot blijft.

QR-code scannen?

Om het scannen van QR-codes in de applicatie te implementeren, kunnen we gebruik maken van populaire React-bibliotheken zoals `react-qr-reader` of `react-qr-scanner`. Beide bieden eenvoudige manieren om QR-codes te scannen, maar met enkele belangrijke verschillen:

1. **`react-qr-reader`:** Deze bibliotheek biedt meer configuratiemogelijkheden, zoals het aanpassen van camera-instellingen en vertraging. Het is ideaal wanneer je meer controle wilt over het scanproces, zoals het kiezen van een specifieke camera (bijvoorbeeld de voor- of achtercamera). Het ondersteunt goed moderne browsers en mobiele apparaten, maar kan iets complexer zijn om op te zetten.

2. react-qr-scanner: Deze optie is eenvoudiger en sneller in te stellen. Het richt zich op een gestroomlijnde ervaring voor het scannen van QR-codes zonder te veel configuraties. Het is ideaal voor basisgebruik, waarbij je snel een QR-code wilt scannen zonder uitgebreide instellingen.

Voor ons spel kiezen we ervoor om **react-qr-scanner** te gebruiken. Deze bibliotheek biedt een eenvoudigere en snellere manier om QR-codes te scannen, wat ideaal is voor ons gebruiksgemak. Omdat het snel in te stellen is en de focus ligt op een gestroomlijnde ervaring, kunnen we direct de unieke speler-ID scannen en deze naar de backend sturen voor verwerking. Dit maakt het proces van eliminatie efficiënt en foutloos, zonder dat we veel configuratie nodig hebben. Het stelt ons in staat om het spel soepel te laten verlopen, terwijl we de nodige veiligheid en snelheid behouden.

Clean code

Clean Architecture en 3-Layer Programming zijn beide architecturale patronen die structuur en duidelijkheid brengen in softwareprojecten. Ze spelen een cruciale rol bij het waarborgen van schaalbaarheid, onderhoudbaarheid en testbaarheid. Hoewel beide benaderingen hun voordelen hebben, biedt Clean Architecture vaak meer flexibiliteit en robuustheid voor complexe applicaties.

Clean Architecture

Clean Architecture is een architectuurprincipe dat de applicatie opdeelt in verschillende lagen met duidelijke verantwoordelijkheden. Het belangrijkste doel van dit patroon is om de core businesslogica, ook wel de "domain layer" genoemd, te isoleren van externe systemen zoals databases, API's of gebruikersinterfaces. Dit zorgt ervoor dat de kernfunctionaliteit van de applicatie onafhankelijk blijft van technische implementaties, waardoor het eenvoudiger wordt om onderdelen te wijzigen of te vervangen zonder de gehele applicatie te beïnvloeden.

Een fundamenteel principe van Clean Architecture is dat afhankelijkheden altijd van buiten naar binnen lopen. Externe componenten, zoals de UI of de database, mogen afhankelijk zijn van de kernlogica, maar de kernlogica zelf mag geen kennis hebben van deze externe systemen. Dit minimaliseert de impact van veranderingen in de infrastructuur en bevordert de modulariteit.

Daarnaast biedt Clean Architecture een strikte scheiding van verantwoordelijkheden. Elke laag heeft een specifieke taak, wat de codebase overzichtelijker maakt en het eenvoudiger maakt om fouten te lokaliseren en te corrigeren. Deze isolatie maakt de businesslogica bovendien eenvoudig te testen, omdat je het kunt valideren zonder dat er afhankelijkheden zijn van externe systemen of frameworks.

De structuur van Clean Architecture bestaat doorgaans uit vier lagen:

1. Core (Domain Layer)
De core, of domeinlaag, bevat de kernfunctionaliteit van de applicatie. Hier bevinden zich de bedrijfsregels, entiteiten en interfaces die de essentiële logica van de applicatie definiëren. Deze laag is volledig onafhankelijk van technische implementaties, wat betekent dat het geen directe connecties heeft met frameworks of databases.
2. Application Layer
De applicatielaag bevat de use cases en de service-logica. Dit zijn de specifieke acties

die de applicatie ondersteunt, zoals het verwerken van een eliminatie in een spel. Deze laag regelt de interactie tussen de core en de buitenwereld en is verantwoordelijk voor het uitvoeren van bedrijfsprocessen.

3. Infrastructure Layer

De infrastructuurlaag bevat de technische implementaties, zoals database-toegang, externe API's en authenticatiediensten zoals Entra ID. Deze laag ondersteunt de applicatielaag maar is er niet direct mee verweven. Hierdoor kunnen wijzigingen in infrastructuurcomponenten plaatsvinden zonder de core of applicatielaag te beïnvloeden.

4. Presentation Layer (UI)

De presentatielaag bevat de gebruikersinterface, zoals de React-gebaseerde Progressive Web App (PWA) in jouw project. Deze laag communiceert met de applicatielaag via API's en biedt de gebruiker toegang tot de functionaliteiten van de applicatie.

Clean Architecture biedt verschillende voordelen. Ten eerste zorgt het voor schaalbaarheid, omdat je gemakkelijk nieuwe functionaliteiten kunt toevoegen zonder bestaande logica te breken. Daarnaast verhoogt het de onderhoudbaarheid, omdat wijzigingen in de gebruikersinterface of database geen directe invloed hebben op de kernlogica. Ook verbetert het de testbaarheid, doordat de businesslogica losstaat van externe systemen, waardoor unit testing eenvoudiger wordt. Tot slot biedt het een duidelijke structuur waarbij elke laag zijn eigen verantwoordelijkheid heeft, wat de samenwerking in een ontwikkelteam vergemakkelijkt.

3-Layer Programming

3-Layer Programming is een eenvoudiger architectuurpatroon dat de applicatie verdeelt in drie hoofdcomponenten. Hoewel het minder complex is dan Clean Architecture, biedt het nog steeds een duidelijke scheiding van verantwoordelijkheden en is het bijzonder geschikt voor kleinere tot middelgrote applicaties.

De structuur van 3-Layer Programming bestaat uit de volgende lagen:

1. Presentation Layer (UI)

De presentatielaag bevat alles wat de gebruiker ziet en waarmee hij interacteert. In jouw project is dit de React PWA die QR-codes scant en resultaten toont. Deze laag is verantwoordelijk voor het verzamelen van input van de gebruiker en het presenteren van output.

2. Business Logic Layer (BLL)

De businesslogicalaag verwerkt de kernfunctionaliteit van de applicatie. In jouw project zou dit de .NET API zijn die bepaalt wie de volgende target is en het eliminatieproces afhandelt. Deze laag implementeert de regels en processen die de applicatie aandrijven.

3. Data Access Layer (DAL)

De data access layer is verantwoordelijk voor het beheren van de gegevensopslag. Dit omvat interacties met databases, zoals het opslaan van gebruikers, targets en scores in jouw applicatie. Deze laag biedt een abstractie van de onderliggende database-technologie.

Hoewel 3-Layer Programming eenvoudiger is te implementeren en geschikt is voor kleinere projecten, heeft het enkele beperkingen. Omdat er directe koppelingen bestaan tussen de

businesslogica en de data laag, is de testbaarheid beperkter en kunnen wijzigingen in de ene laag direct impact hebben op de andere. Dit maakt de architectuur minder flexibel en moeilijker schaalbaar naarmate de applicatie groeit.

Verschillen tussen Clean Architecture en 3-Layer Programming

Het belangrijkste verschil tussen Clean Architecture en 3-Layer Programming ligt in de complexiteit en de manier waarop afhankelijkheden worden beheerd. Clean Architecture is complexer, maar biedt meer flexibiliteit en testbaarheid doordat afhankelijkheden van buiten naar binnen lopen en de core volledig losstaat van externe systemen. In 3-Layer Programming zijn de afhankelijkheden direct gekoppeld, wat de eenvoud bevordert, maar de uitbreidbaarheid en testbaarheid beperkt.

Bij complexe applicaties met uitgebreide bedrijfslogica biedt Clean Architecture aanzienlijke voordelen. Het maakt het eenvoudiger om nieuwe functionaliteiten te integreren, fouten op te sporen en wijzigingen door te voeren zonder andere delen van de applicatie te beïnvloeden. 3-Layer Programming is daarentegen geschikter voor eenvoudige tot middelgrote applicaties, waar snelheid van implementatie en eenvoud van onderhoud de belangrijkste overwegingen zijn.

Waarom kiezen voor Clean Architecture in jouw project?

In dit project is Clean Architecture de meest geschikte keuze vanwege de modulariteit, schaalbaarheid en onderhoudsgemak die het biedt. Door de duidelijke scheiding tussen businesslogica, infrastructuur en presentatie kun je eenvoudig nieuwe game modes toevoegen zonder de bestaande code te breken. Bovendien vergemakkelijkt de isolatie van lagen het opsporen en corrigeren van fouten, omdat problemen zich vaak binnen één specifieke laag voordoen.

Ook biedt Clean Architecture een hoge mate van schaalbaarheid, wat cruciaal is als het spel groeit of wanneer er meerdere edities van de DEV-Cruise worden georganiseerd. Door deze architectuur te volgen, zorg je ervoor dat de applicatie flexibel blijft, eenvoudig te onderhouden is en klaar is voor toekomstige uitbreidingen.

PWA

Een Progressive Web App (PWA) is een webapplicatie die werkt als een native app op mobiele apparaten en desktops. Het combineert de flexibiliteit van een website met de gebruikerservaring van een mobiele applicatie. Dit betekent dat de applicatie toegankelijk is via een webbrowser, maar ook offline kan werken en op het startscherm van een apparaat kan worden geïnstalleerd.

Waarom kiezen we voor een PWA?

Een Progressive Web App (PWA) biedt verschillende voordelen. Gebruikers hoeven geen app te downloaden uit de App Store of Google Play Store, omdat de applicatie direct beschikbaar is via een URL. Daarnaast kunnen bepaalde onderdelen van de app ook zonder internetverbinding worden gebruikt dankzij offline functionaliteit. Nieuwe functionaliteiten of bugfixes zijn direct beschikbaar zonder dat gebruikers handmatig een update hoeven te installeren, wat zorgt voor snelle updates. Bovendien besparen we tijd en kosten, omdat we geen aparte native apps voor

iOS en Android hoeven te ontwikkelen. Tot slot kunnen gebruikers de app eenvoudig toevoegen aan hun startscherm, waardoor deze net als een native app te openen is en een gebruiksvriendelijke ervaring biedt.

PWA implementeren met React

Het bouwen van een PWA met React is relatief eenvoudig dankzij bestaande tools en libraries. Create React App (CRA), een veelgebruikte tool voor het opzetten van React-projecten, heeft standaard ondersteuning voor het maken van een PWA.

React maakt gebruik van een service worker om de applicatie offline beschikbaar te maken en zorgt ervoor dat resources (zoals afbeeldingen, scripts en CSS) worden gecachet. Hierdoor blijft de app werken, zelfs als de gebruiker tijdelijk geen internetverbinding heeft.

Waarom is een PWA ideaal voor dit project?

Gezien de aard van het spel, waarbij spelers snel toegang moeten hebben tot hun QR-code en notificaties ontvangen bij eliminaties, is een PWA de meest efficiënte keuze. Gebruikers kunnen de app eenvoudig installeren op hun telefoon, hebben altijd toegang, en het systeem kan offline blijven functioneren als er tijdelijk geen internet beschikbaar is.

Deze aanpak zorgt ervoor dat we een schaalbare, snelle en gebruiksvriendelijke applicatie leveren die eenvoudig te beheren en bij te werken is.

Grafieken

Op basis van de verschillende vereisten voor het project, zoals gebruiksgemak, prestaties, flexibiliteit en functionaliteit, zijn hier drie geschikte grafiekbibliotheken voor React:

Recharts

Recharts is een uitstekende keuze voor eenvoudige en middelgrote datavisualisaties, zoals leaderboards en statistieken in de applicatie. Het is gebruiksvriendelijk, biedt goede documentatie, en heeft ingebouwde functionaliteiten zoals tooltips en animaties. Dit maakt het bijzonder geschikt voor het tonen van eliminaties, ranglijsten en andere eenvoudige gegevens.

Voordelen:

- Makkelijk te integreren en te configureren.
- Ondersteunt verschillende grafiektypen (staaf-, lijn-, en cirkeldiagrammen).
- Uitgebreide en gebruiksvriendelijke documentatie.
- Geschikt voor eenvoudige en interactieve visualisaties.

Nadelen:

- Beperkt in maatwerk en flexibiliteit voor zeer complexe visualisaties.
- Niet de beste keuze voor toepassingen met zeer grote datasets.

Chart.js (via react-chartjs-2)

Chart.js is ideaal voor grotere datasets en complexere grafieken. Het is zeer geschikt voor toepassingen waarbij prestaties en interactie belangrijk zijn, zoals dynamische grafieken voor het bijhouden van prestaties of trends in eliminaties over de tijd. Via de **react-chartjs-2** wrapper kan het eenvoudig in React worden geïntegreerd.

Voordelen:

- Ondersteunt veel verschillende grafiektypen, zoals radar- en bubbeldiagrammen.
- Goede prestaties bij grote hoeveelheden data door gebruik van Canvas.
- Ingebouwde animaties en interactie (zoals tooltips en zoom).
- Actieve community en documentatie.

Nadelen:

- Complexer dan Recharts, met een hogere leercurve.
- Minder geschikt voor eenvoudige visualisaties.

ApexCharts (via react-apexcharts)

ApexCharts is zeer geschikt voor real-time dashboards en grafieken met dynamische updates. Het biedt krachtige interactieve mogelijkheden zoals zoom, filters en drill-down, wat handig kan zijn voor meer geavanceerde visualisaties van het spelverloop. Het is ideaal voor het visualiseren van de voortgang van spelers in real-time.

Voordelen:

- Ondersteunt complexe interacties zoals dynamische updates, zoom, en filters.
- Zeer geschikt voor realtime data en gedetailleerde visualisaties.
- Veel configuratiemogelijkheden en een breed scala aan grafiektypen.
- Goed voor grotere en dynamische datasets.

Nadelen:

- Grotere pakketgrootte dan andere opties.
- Meer configuratie vereist voor eenvoudige toepassingen.

Conclusie

Gezien de vereisten van het project – het tonen van een leaderboard, het volgen van eliminaties en het bieden van eenvoudige tot middelgrote visualisaties – is **Recharts** de gekozen bibliotheek. Recharts biedt de juiste balans tussen gebruiksgemak, flexibiliteit en prestaties voor de beoogde toepassingen. Het is eenvoudig te integreren, heeft een gebruiksvriendelijke configuratie en ondersteunt de benodigde grafiektypen, zoals staafdiagrammen en lijngrafieken, die ideaal zijn voor het visualiseren van eliminaties en ranglijsten. Bovendien biedt het de nodige interactiviteit voor een dynamische en overzichtelijke gebruikerservaring.

Styling conclusie aanpassen met prime react en tailwind

Na het vergelijken van de drie populairste CSS-frameworks voor React, Tailwind CSS, Material UI (MUI) en Bootstrap, blijkt dat elk zijn eigen voordelen en nadelen heeft, afhankelijk van de specifieke behoeften van het project.

Tailwind CSS

Tailwind is een utility-first CSS-framework, wat betekent dat je de styling van je componenten volledig kunt aanpassen door eenvoudige utility-klassen te combineren. Het biedt volledige controle over je ontwerp zonder de noodzaak om stijlen te overschrijven. Tailwind is perfect voor projecten waar maatwerk en ontwerpflexibiliteit centraal staan.

Voordelen:

- Volledige controle over het ontwerp zonder vooraf gedefinieerde stijlen.
- Geen overbodige stijlen die geladen worden, wat de bestandsgrootte verkleint.
- Makkelijker te onderhouden bij grotere projecten door de duidelijke structuur.

Nadelen:

- Vereist een hogere leercurve voor ontwikkelaars die gewend zijn aan traditionele frameworks.
- JSX-code kan wat rommelig worden bij veel klassen.

Tailwind biedt de grootste flexibiliteit en controle over het ontwerp, ideaal voor ontwikkelaars die custom designs willen maken zonder gebonden te zijn aan standaard stijlen.

Material UI (MUI)

Material UI biedt een uitgebreide bibliotheek van kant-en-klare React-componenten, gebaseerd op Google's Material Design-richtlijnen. Het is ideaal voor het snel bouwen van visueel aantrekkelijke en functionele interfaces.

Voordelen:

- Veel kant-en-klare componenten die snel geïntegreerd kunnen worden.
- Goede ondersteuning voor theming en aanpassingen.

Nadelen:

- Minder flexibel dan Tailwind voor volledig op maat gemaakte ontwerpen.
- Grotere bestandsgrootte, vooral bij gebruik van veel componenten.

MUI is een uitstekende keuze voor snelheid, maar als volledige ontwerpcontrole en maatwerk belangrijker zijn, is Tailwind een betere optie.

Bootstrap (via React-Bootstrap)

Bootstrap is een veelgebruikte CSS-bibliotheek met kant-en-klare componenten en een responsief grid-systeem. React-Bootstrap maakt het mogelijk om de Bootstrap-componenten naadloos in React-projecten te gebruiken.

Voordelen:

- Snelle ontwikkeling met veel kant-en-klare componenten.
- Sterke documentatie en community-ondersteuning.

Nadelen:

- Beperkte flexibiliteit in ontwerp en styling.
- Het ontwerp kan snel generiek aanvoelen, omdat Bootstrap vaak herkend wordt.

Hoewel Bootstrap snel resultaat biedt, biedt het niet dezelfde maatwerkmogelijkheden als Tailwind.

PrimeReact

PrimeReact is een veelzijdige UI-componentenbibliotheek die een breed scala aan thematische en functionele componenten biedt. Het onderscheidt zich door een uitgebreide set geavanceerde UI-elementen zoals tabellen, grafieken en drag-and-drop-functionaliteit.

Voordelen:

- Uitgebreide componentenset met complexe en geavanceerde UI-elementen.
- Ondersteuning voor theming en eenvoudige aanpassing aan de huisstijl.
- Goede prestaties en compatibiliteit met React, inclusief integratie met state management-oplossingen.
- Responsive design en ondersteuning voor moderne webstandaarden.

Nadelen:

- Grotere bestandsgrootte door de uitgebreide componentenbibliotheek.
- Minder populair dan MUI of Bootstrap, waardoor er minder community-resources beschikbaar zijn.

primeReact biedt de meest uitgebreide functionaliteit en is ideaal voor applicaties die complexe UI-componenten vereisen. Het biedt voldoende flexibiliteit om het design aan te passen aan specifieke huisstijlen en heeft sterke ondersteuning voor geavanceerde gebruikersinterfaces.

Conclusie

Na een grondige vergelijking kiezen we voor PrimeReact als component library voor dit project. PrimeReact biedt een uitgebreide set componenten die geschikt zijn voor zowel eenvoudige als complexe interfaces. Het ondersteunt flexibele theming, waardoor de applicatie eenvoudig aangepast kan worden aan de huisstijl van Euricom. Daarnaast biedt PrimeReact geavanceerde UI-functionaliteit, zoals tabellen, grafieken en interactieve elementen, die direct bruikbaar zijn.

Een belangrijk voordeel is dat PrimeReact goed aansluit bij Azure-integraties en de schaalbaarheid biedt die nodig is voor toekomstige uitbreidingen. Dankzij de brede

componentenset kunnen we sneller ontwikkelen en tegelijkertijd maatwerk leveren waar nodig, wat het een ideale keuze maakt voor dit project.

Pipeline

Een CI/CD pipeline (Continuous Integration en Continuous Deployment/Delivery) is een geautomatiseerd proces waarmee code sneller en betrouwbaarder kan worden gebouwd, getest en geïmplementeerd. Dit proces zorgt ervoor dat elke wijziging in de codebase automatisch wordt gecontroleerd en, indien succesvol, direct kan worden uitgerold naar een test- of productieomgeving.

- Continuous Integration (CI):
Hierbij worden nieuwe codewijzigingen regelmatig (meestal meerdere keren per dag) samengevoegd in de hoofdcodebase. Elke wijziging wordt automatisch getest om fouten of bugs vroegtijdig te detecteren.
- Continuous Deployment/Delivery (CD):
Na succesvolle tests wordt de nieuwe code automatisch uitgerold (Deployment) of klaargezet voor handmatige goedkeuring (Delivery) om naar de live omgeving te gaan.

Hoe kan ik een CI/CD Pipeline gebruiken voor dit project?

In dit project kan een CI/CD pipeline het ontwikkelproces efficiënter en betrouwbaarder maken. Het proces begint wanneer je wijzigingen aanbrengt in de applicatie, zowel in de React-frontend als de .NET-backend, en deze doorstuurt naar een versiebeheersysteem zoals GitHub of GitLab. Zodra er code wordt gepusht, start de pipeline automatisch met het uitvoeren van tests om te controleren of alles correct werkt. Dit omvat verschillende soorten tests, zoals unit tests, integratietests en end-to-end tests, die helpen om fouten vroegtijdig te detecteren en de kwaliteit van de applicatie te waarborgen.

Als de tests succesvol zijn, volgt het buildproces, waarbij de applicatie automatisch wordt opgebouwd en geoptimaliseerd. Voor de React Progressive Web App (PWA) betekent dit dat de app wordt voorbereid voor productie met minimale bundelgrootte en optimale prestaties. Voor de .NET-backend wordt de API gecompileerd en klaargemaakt voor gebruik.

Wanneer alle stappen succesvol zijn afgerond, kan de pipeline de nieuwste versie automatisch uitrollen naar een staging-omgeving voor verdere tests of direct deployen naar productie om de applicatie live te zetten. Deze geautomatiseerde workflow versnelt het ontwikkelproces, vermindert handmatige fouten en zorgt ervoor dat nieuwe functionaliteiten snel en betrouwbaar beschikbaar komen voor gebruikers.

Waarom is een CI/CD Pipeline nuttig voor dit project?

Een CI/CD pipeline biedt meerdere voordelen die het ontwikkelproces optimaliseren en de kwaliteit van de applicatie waarborgen. Een belangrijk voordeel is snellere ontwikkeling, omdat elke wijziging automatisch wordt getest en uitgerold. Dit maakt het mogelijk om sneller nieuwe functionaliteiten toe te voegen en eventuele fouten vroegtijdig te identificeren en op te lossen.

Daarnaast verhoogt de pipeline de betrouwbaarheid van de applicatie. Dankzij geautomatiseerde tests en gecontroleerde deployments worden fouten in de live omgeving

voorkomen, wat vooral cruciaal is voor een interactieve applicatie zoals deze. Efficiëntie speelt ook een grote rol, omdat er minder handmatige stappen nodig zijn. Dit bespaart tijd en vermindert het risico op menselijke fouten tijdens het testen en deployen.

Een ander voordeel is consistentie; iedere update doorloopt exact hetzelfde proces, wat zorgt voor een stabiele en voorspelbare applicatie. Bovendien biedt de pipeline snelle feedback. Wanneer er een fout optreedt, ontvang je direct een melding, zodat je snel kunt reageren en het probleem kunt oplossen.

Al deze aspecten samen zorgen voor een efficiënter, betrouwbaarder en flexibeler ontwikkelproces, wat essentieel is voor het succes en de stabiliteit van de applicatie.

Tools die je kunt gebruiken voor de CI/CD Pipeline:

Voor dit project kun je kiezen uit verschillende CI/CD-oplossingen die compatibel zijn met React en .NET:

- GitHub Actions: Ideaal als je je code host op GitHub. Hiermee kun je eenvoudig pipelines opzetten voor zowel de frontend als de backend.
- GitLab CI/CD: Als je GitLab gebruikt, biedt dit een krachtige integratie voor automatisering.
- Azure DevOps: Zeer geschikt voor .NET-projecten en biedt uitgebreide ondersteuning voor complexe pipelines.
- Jenkins: Open-source en flexibel, vooral handig voor grotere en meer aangepaste pipelines.

Door een CI/CD pipeline te gebruiken, zorg je ervoor dat je project altijd stabiel is, snel kan worden bijgewerkt, en voldoet aan de kwaliteitsnormen. Dit is vooral belangrijk voor een applicatie die live gebruikt wordt tijdens evenementen zoals de DEV-Cruise.

Conclusie

Voor dit project maken we gebruik van Azure DevOps. Dit platform is bij uitstek geschikt voor .NET-projecten en biedt uitgebreide ondersteuning voor zowel eenvoudige als complexe pipelines. Azure DevOps sluit bovendien goed aan bij onze bestaande infrastructuur in Microsoft Azure, waardoor integratie met andere Azure-diensten, zoals Azure App Services en Entra ID, soepel verloopt.

Met Azure Pipelines kunnen we zowel de React-frontend als de .NET-backend automatiseren, van build en testen tot deployments naar verschillende omgevingen (zoals staging en productie). Deze keuze zorgt ervoor dat ons project altijd stabiel is, snel kan worden bijgewerkt en voldoet aan de kwaliteitsnormen. Dit is vooral belangrijk voor een applicatie die live gebruikt wordt tijdens evenementen zoals de DEV-Cruise.

<https://www.youtube.com/watch?v=4BibQ69MD8c&t=1238s> min 7-23

Error handling

Application Insights is een monitoring- en analysetool van Microsoft Azure die helpt om prestaties, gebruik en fouten in je applicatie te volgen. Het biedt diepgaande inzichten in hoe je applicatie functioneert, inclusief foutdetectie, prestatie metrieken en gebruikersinteracties.

Het is vooral nuttig voor applicaties die gebouwd zijn met .NET (backend) en React (frontend) omdat het real-time gegevens geeft over hoe de applicatie presteert in productie.

Waarom Application Insights gebruiken?

Monitoring en analyse zijn cruciaal voor het waarborgen van de prestaties en betrouwbaarheid van de applicatie. Foutdetectie en logging maken het mogelijk om automatisch fouten en uitzonderingen (errors en exceptions) vast te leggen en te analyseren, waardoor problemen snel geïdentificeerd en opgelost kunnen worden. Daarnaast speelt prestatieanalyse een belangrijke rol door de responstijden van de .NET API en de laadtijden van de React PWA te meten. Dit helpt bij het optimaliseren van de applicatie en het opsporen van knelpunten.

Een ander essentieel aspect is het volgen van gebruikersgedrag, wat inzicht biedt in hoe gebruikers door de applicatie navigeren. Dit kan bijvoorbeeld aantonen hoeveel mensen een QR-code scannen of welke functies het meest gebruikt worden. Daarnaast zorgt real-time monitoring voor directe waarschuwingen bij kritieke problemen zoals downtime of prestatieverlies, waardoor je snel kunt ingrijpen en de continuïteit van de applicatie kunt garanderen.

Ten slotte biedt schaalbaarheid de mogelijkheid om te controleren hoe de applicatie presteert bij toenemende gebruikersaantallen. Dit is vooral belangrijk bij grootschalige evenementen zoals de DEV-Cruise, waar de applicatie onder hoge belasting betrouwbaar moet blijven functioneren. Samen zorgen deze monitoringfuncties voor een stabiele, efficiënte en gebruiksvriendelijke applicatie die proactief wordt beheerd en geoptimaliseerd.

Welke gegevens kan je analyseren met Application Insights?

Monitoring in de applicatie biedt waardevolle inzichten in prestaties, fouten en gebruikersgedrag. Fouten (Exceptions) helpen bij het vaststellen van welke fouten het meest voorkomen en in welk deel van de applicatie ze optreden, waardoor je gericht problemen kunt opsporen en oplossen. Verzoekprestaties (Request Performance) geven inzicht in de snelheid waarmee API-endpoints van de .NET backend reageren, wat cruciaal is voor het optimaliseren van de responstijden en het verbeteren van de gebruikerservaring.

Met gebruikersstromen (User Flows) kun je analyseren welke routes gebruikers binnen de React-app volgen, wat waardevolle informatie oplevert over hoe mensen door de applicatie navigeren en waar eventuele knelpunten zich bevinden. Daarnaast biedt het gebruik van Custom Events de mogelijkheid om specifieke acties te volgen, zoals het scannen van een QR-code of het behalen van een overwinning, waardoor je diepgaand inzicht krijgt in het gedrag van de gebruikers.

Tot slot maakt de Live Metrics Stream het mogelijk om real-time prestaties en foutmeldingen te monitoren zonder te wachten op batchverwerking. Dit zorgt ervoor dat je direct op problemen kunt reageren en de stabiliteit van de applicatie kunt waarborgen. Door deze monitoringsmogelijkheden te benutten, kun je proactief problemen opsporen, prestaties verbeteren en de algehele gebruikservaring optimaliseren.

Hoe analyseer je gegevens in Application Insights?

Azure Application Insights biedt diverse tools om de prestaties en het gedrag van je applicatie te monitoren en te analyseren. Het Overzichtsdashboard in het Azure Portal geeft een helder beeld van algemene prestatiecijfers, zoals het aantal verzoeken, fouten en laadtijden. Dit maakt het eenvoudig om in één oogopslag de gezondheid van de applicatie te beoordelen en snel afwijkingen te detecteren.

Voor diepgaandere analyses biedt Log Analytics de mogelijkheid om queries uit te voeren met behulp van de Kusto Query Language (KQL). Hiermee kun je gedetailleerde zoekopdrachten doen op loggegevens, bijvoorbeeld om specifieke foutpatronen te herkennen of prestatieproblemen te analyseren.

Met User Flows krijg je inzicht in hoe gebruikers door de applicatie navigeren. Dit kan variëren van het moment van inloggen tot acties zoals het scannen van een QR-code. Deze informatie is waardevol om te begrijpen hoe gebruikers de applicatie gebruiken en waar optimalisaties mogelijk zijn.

Daarnaast is er het Failures Blade, een speciaal tabblad in het Azure Portal waar je gedetailleerde foutanalyses kunt uitvoeren. Hier kun je stack traces, request logs en andere diagnostische gegevens bekijken, waardoor je snel de oorzaak van fouten kunt achterhalen en oplossen.

Door deze monitoringtools te gebruiken, kun je niet alleen de prestaties van je applicatie bewaken, maar ook proactief problemen opsporen en de gebruikerservaring verbeteren.

Waarom is dit belangrijk?

Voor een interactieve applicatie zoals de DEV-Cruise-game is het cruciaal om snel te reageren op fouten en prestatieproblemen. Application Insights biedt hiervoor de volgende voordelen:

Problemen snel te vinden en op te lossen: Bijvoorbeeld als een QR-code niet juist wordt verwerkt of als de API traag reageert.

Gebruikersgedrag te begrijpen: Het biedt inzicht in hoe spelers de game gebruiken en waar ze eventueel vastlopen.

Betere schaalbaarheid: Hiermee kun je monitoren hoe de app presteert als veel gebruikers tegelijk de game spelen.

Indien nodig kan er ook custom telemetry worden toegevoegd om specifieke gebeurtenissen te volgen, zoals eliminaties of het starten van de finale.