

|  |
| --- |
|  |
| **Sharkfrenzy.io** |
| **Project front End/Back End** |
|  |
|  |

# Sharkfrenzy.io

## Beschrijving

Multiplayer game rond het thema ‘haaien’. In dit spel is het de bedoeling zo groot mogelijk te worden als haai en andere spelers te elimineren. Het groeien tot een zo groot mogelijke haai kan worden voltooid door vissen, die random spawnen in het speelveld, op te eten en/of andere spelers te elimineren. Grote haaien kunnen kleinere haaien opeten, echter kunnen ook de kleine haaien deze keer de grote haai opeten. Echter wel door meermaals schade aan te richten aan de grotere haai. Hoeveel keer wordt berekend hoeveel kleiner de haai is t.o.v. de grotere haai.   
De bedoeling is ook de mogelijkheid tussen meerdere spelmodussen voorzien. Deze hierboven al uitgelegd wordt de FFA of ‘free for all’. Een andere die we wensen te creëren is TDM of ‘Team Deathmatch’. Hierbij kunnen slechts 8 spelers een kamer joinen, deze 8 spelers worden dan op hun beurt in 2 teams verdeeld. De bedoeling hierin is dan dat het team die het langst de grootste haai heeft wint.

## Doelgroep

Gebruikers die interesse hebben om een multiplayer game te spelen met vrienden of met andere mensen.

## Verwachtingen

Als we kijken naar het succes van Agar.io en Slither.io zou deze game wel kunnen aanslaan. We onderscheiden ons door de mogelijkheid tot teamplay toe te voegen aan onze game wat in de andere games niet het geval is.

## Teamleden en verantwoordelijkheden

* Gerben Rosseel: Frontend & Backend, Designer
* Lorenz Vercoutere: Groepsleider, verantwoordelijke voor deadlines, frontend & backend
* Mathias Claeys: Frontend & backend

## Technische analyse

In de technische analyse is het de bedoeling dat we nader verklaren over hoe we het volledige concept dat we voor ogen hebben praktisch tot uitvoering zullen brengen. Hiervoor bespreken we de verschillende stappen en research die we hebben overlopen en zullen gebruiken, hoe het spel eruit zal zien op grafisch vlak enz. Onderstaande documentatie is dus vooral een technische uitleg over hoe we tot het eindproduct zijn gekomen, welke code en programmatuur wij hiervoor gebruikt hebben en welke technieken we gebruiken om een multiplayer systeem op te bouwen.

## Kritische milestones(deadlines)

* 18 november: Github opzetten en projecten aanmaken
* 23 november: Designs klaar
* 30 november: Basis al reeds geprogrammeerd voor zowel frontend als backend
* 9 december: Frontend afgewerkt
* 20 december: Backend afgewerkt
* 23 december: Project in grote lijnen afgewerkt hebben

## Referentie project

<https://github.com/lorenzvercoutere/projectfrontenbackend>

# Implementatie

## In het begin

Na goedkeuring van het project begon natuurlijk het moeilijkste gedeelte. Het effectief starten aan het project, iets wat uiteindelijk langer heeft geduurd dan we zelf wilden, maar daar kom ik later nog op terug.

In het begin kozen we ervoor meteen onze structuur op te stellen en dus de nodige pagina’s aan te maken. Na deze opstelling konden we al een schatting doen van wat er allemaal zou moeten gebeuren en wie wat precies voor zijn rekening zou nemen.

## De basis

De structuur van onze applicatie was nu in orde en na overleg in de Facebookgroep en Slack werden de rollen verdeeld. Het grafische aspect alsook front-end development ging Gerben op zich nemen. Het grafische was een groot pak aangezien we meerdere haaien wilden implementeren het speelveld zelf creëren en natuurlijk de animatie van alles. Zonder deze basis kon er ook weinig gebeuren aan het maken van de rest van het spel dus werd er al een groot deel van het werk meteen vroeg gedaan zodat er geen vertraging zou oplopen bij de rest.

Terwijl Gerben bezig was met alle grafische delen van de applicatie begonnen Lorenz en Mathias aan de basis van de app, dit was ook het grootste deel van hun taak namelijk de backend van de applicatie. Meteen was dus de rollenverdeling duidelijk. Mathias ging op zoek naar een manier van het creëren van een multiplayer spel en Lorenz zette al de basis van de applicatie neer, onder meer de nodige installaties zoals gulp.

## Inspiratie

Het idee van Sharkfrenzy kwam natuurlijk van de alom bekende .io games. Een aantal bekende hieronder zijn natuurlijk curvefever en agar, maar de koning onder al deze is en blijft slither. Opdat dit een redelijk populair spel is vonden we al snel meerdere bronnen waar anderen hun eigen slither hadden gemaakt en ook documenteerde wat ze precies allemaal gebruikten om tot een werkende versie te komen. Echter waren deze allemaal op een andere manier of taal geschreven en konden we alleen de manieren bestuderen om deze dan ook bijgevolg zelf toe te passen.

## Complexiteit

We kwamen er al snel achter dat wat een simpel spel leek om te maken helemaal niet zo simpel ging zijn als we origineel dachten. Er kwam veel meer werk bij te kijken en dus moesten we al snel ons idee aanpassen. Waar we dachten meerdere gamemodes bij te houden en te maken moesten we al snel schrappen omdat dit veel meer complexiteit ging meebrengen met zich. Wat ons meteen brengt tot het volgend punt.

## De inhoud

Om alles bij te houden (spelers, gekozen skin, score, munten, etc.) moet er gebruik worden gemaakt van een database. Hiervoor kozen we zoals gezien in de les van back-end voor MongoDB. Om dit alles te kunnen opslaan zal de gebruiker zich dus moeten kunnen inloggen, indien niet zal alles meteen verloren gaan. Met dit in het achterhoofd besloten we dan ook een chatsysteem te implementeren met de naam dus van de ingelogde gebruiker. Dit is pas iets waar we later op terug zijn gekomen.

# Realisatie

## Werkuren

De werkuren van elk groepslid is anders, de ene zijn kennis over een onderwerp is nu eenmaal groter of kleiner dan dat van zijn medeteamlid, wat ten volste normaal is. Iedereen heeft zijn of haar sterke en minder sterke punten. Hier hebben we dan ook op ingespeeld bij het verdelen van de rollen.

In het begin heeft Gerben meer uren en werk gestoken in het project dan Mathias of Lorenz, het grafische aspect moest namelijk ook eerder af zijn. Eenmaal het grafische aspect af was gingen zowel de uren die Mathias als Lorenz in het project staken de lucht in. Terwijl Mathias en/of Lorenz bezig waren met het back-end programmeren werd er gewerkt aan de documentatie, opzoekwerk of front-end door de rest van het team of omgekeerd, er was nooit iemand die niks deed.

Back-end zou sneller moeten gerealiseerd kunnen worden dan front-end werd ons altijd wijsgemaakt, echter leek dit niet echt van toepassing op ons project. Het backend heeft veel uren op zijn naam gekregen en veel tijd in beslag genomen, veel meer als je het vergelijkt met het front-end gedeelte of zelfs het grafische. Toch kunnen we besluiten dat elk teamlid zijn deel tot zover zijn capaciteiten reiken heeft uitgevoerd.

## Struikelstenen

Het grootste struikelsteen was natuurlijk het spel zelf, de mechaniek dan vooral. Het was op elk vlak nieuw voor ieder van ons en dat zag je ook. Er werd veel tijd gestoken in dingen die veel sneller gingen gaan moesten we het bijvoorbeeld in een les al eens hebben aangehaald. Veel moest er opgezocht worden en dit had natuurlijk een nefaste werking op de vooruitgang van alles.

Het gebruik van Github bracht ook wat problemen met zich mee. Regelmatig waren files van de ene groepslid corrupt en kreeg deze niet meer de laatste updates binnen waardoor er regelmatig conflicten waren. Weer een aspect die tijd innam, tijd die we minder en minder hadden.

Een ander groot struikelsteen is dat we te veel hooi op onze vork hebben genomen, door te veel te willen toevoegen aan het spel waren we al snel aan het verdrinken als het ware, dan hebben we ook meteen de beslissing genomen een aantal dingen te schrappen of aan de kant te schuiven totdat we het belangrijkste aspect werkende kregen, namelijk het spel zelf. Hierdoor werd dus een tweede gamemode geschrapt.

Het grootste probleem blijft wel de communicatie tussen het team. Er werd niet altijd genoeg gecommuniceerd waardoor deadlines niet werden nageleefd. Uitstelgedrag speelde hier ook een rol in.

Het is door een combinatie van al deze moeilijkheden dat we dan ook niet volledig klaar zijn met alles. Je denkt dat iets nog snel en makkelijk zal zijn om te maken en al snel blijkt dit niet zo, steekt er een probleem op en eenmaal je deze hebt opgelost verschijnt een volgend probleem. We kennen dit allemaal wel en het is dit sneeuwbaleffect dat ons de das wat heeft omgedaan.

## Positief

Zoals ik al zei hebben we veel zelf moeten opzoeken. Hoe werkt iets, hoe kunnen we dit oplossen, allemaal vragen die we regelmatig hebben gehad en het feit dat we telkens zelf tot een oplossing zijn gekozen is toch wel iets waar we trots mogen op zijn. We hebben het ons niet makkelijk gemaakt en toch hebben we veel werk kunnen verzetten die we niet hebben gezien in de lessen.

Als we de applicatie starten en we zien wat we hebben gemaakt, puur uit zelfstudie borrelt er toch wat trots omhoog bij ieder van ons.

## Good to know

Met alles wat we nu weten denk ik dat we allemaal van mening zijn dat we nooit voor een multiplayer spel gingen kiezen. Het werk dat het met zich meebracht is veel meer dan ieder van ons had verwacht. Een site die wat info uit een database haalt, bijvoorbeeld die van je favoriet voetbalteam, en daar iets mee doen leek ons een dan meer veilige oplossing. Ook één waar we veel minder gingen moeten opzoekwerk voor doen en zelfstudie.

De communicatie hadden we waarschijnlijk ook wel anders aangepakt, en ook meer de mensen gepushed om de gekozen deadlines te respecteren.

## Samenvatting

Ondanks alles denk ik dat we toch blij mogen zijn met wat we hebben gepresteerd. Voor anderen zal dit misschien makkelijker zijn, maar ons team bestond nu eenmaal niet uit de master programmeurs die we willen zijn. Elk heeft zijn best gegeven om tot ons eindproduct te komen, ook al is deze niet volledig zoals we hadden gehoopt.