405273@student.fontys.nl

405273  
Semester 6 ICT & Software  
Versie: 0.4  
Datum 19/3/2021

Bithero online

Jaap van der Meer

# Versiebeheer

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Versie | Omschrijving | Auteur |
| 24/02/2021 | 0.1 | Opzet document | Jaap van der Meer |
| 03/03/2021 | 0.2 | Use case view, Logical view | Jaap van der Meer |
| 05/03/2021 | 0.3 | Dbo toegevoegd. Wireframes verwerkt. | Jaap van der Meer |
| 19/03/2021 | 0.4 | Container + component diagram uitgewerkt | Jaap van der Meer |
|  |  |  |  |

Inhoud

[Versiebeheer 1](#_Toc67473672)

[Introductie 3](#_Toc67473673)

[2. Architectuur representatie 4](#_Toc67473674)

[Use case view 4](#_Toc67473675)

[Logical view 4](#_Toc67473676)

[Data view 4](#_Toc67473677)

[Deployment view 4](#_Toc67473678)

[3. Use case view 5](#_Toc67473679)

[3.1 Actoren 5](#_Toc67473680)

[3.2 Use cases 5](#_Toc67473681)

[3.3 Use case Diagram 10](#_Toc67473682)

[3.4 Systeem context 11](#_Toc67473683)

[3.5 Wireframes 12](#_Toc67473684)

[5. Logical view 18](#_Toc67473685)

[5.1 Frameworks 18](#_Toc67473686)

[5.2 Component diagram 20](#_Toc67473687)

[5.3 Code 21](#_Toc67473688)

[5.4 Interfaces 22](#_Toc67473689)

[6. Data view 23](#_Toc67473690)

[6.1 Database model 23](#_Toc67473691)

[7. Deployement view 24](#_Toc67473692)

[7.1 Deployement diagram 24](#_Toc67473693)

# Introductie

Voor semester 6 van de opleiding ICT & Software van Fontys hogescholen heb ik de opdracht gekregen om een Enterprise applicatie te maken. De applicatie die ik ga maken is Return of Bithero Online (RoBHO). RoBHO is een browser game die je samen kan spelen met 1-4 mensen en is gebaseerd op de Bithero game die ik heb gemaakt in semester 2.

Dit document moet de architectuur van de applicatie in kaart brengen.

# 2. Architectuur representatie

Dit document geeft de architectuur weer d.m.v. het “4+1” model [Kruchten]. De views die in dit document worden gebruikt zijn:

## Use case view

**Doelgroep:** Alle stakeholders van het systeem.

**Context:** In deze view worden alle actoren beschreven. Vervolgens worden de belangrijkste scenario’s beschreven d.m.v. use cases en use case diagrammen.

## Logical view

**Doelgroep:** Designers

**Context:** In de logical view worden alle componenten van het systeem beschreven. Hierin wordt er verteld wat een component doet en hoe dit in connectie staat met andere componenten. Ook worden de interfaces gedefinieerd.

## Data view

**Doelgroep:** Data specialisten, Database administrators

**Context:** Hierin wordt het Database model toegelicht en wordt er duidelijk gemaakt hoe de data flow in elkaar zit.

## Deployment view

**Doelgroep:** Deployment managers.

**Context:** In deze view wordt beschreven hoe de CI/CD is opgezet.

# 3. Use case view

## 3.1 Actoren

**Gebruiker**

Gebruiker staat voor een niet ingelogde gebruiker. De gebruiker kan registreren en inloggen.

**Speler**

Een speler is een inlogde gebruiker. De speler kan zijn karakters beheren, een server joinen en het spel spelen.

**Admin speler**

Een admin speler is een speler die een server heeft aangemaakt. De admin speler heeft iets meer rechten dan een normale speler.

## 3.2 Use cases

|  |  |
| --- | --- |
| Use Case 01 | Registreren |
| Actor | Gebruiker |
| Aannames | De actor bevindt zich op de registratie pagina |
| Basic Flow | 1. De actor vult een username, email, password en repeat password in. 2. Het systeem controleert de gegevens en laat een melding zien dat een account is aangemaakt. De actor wordt doorgestuurd naar de login pagina. [1][2][3] |
| Uitzonderingen | 1. De username bestaat niet uit 4-16 characters. Geef een melding. 2. Er is geen geldige email ingevuld. Toon een melding. 3. De wachtwoorden bestaan niet uit 5-16 characters of komen niet overeen. Geef een melding. |
| Resultaat | De actor heeft zich geregistreerd. |

|  |  |
| --- | --- |
| Use Case 02 | Inloggen |
| Actor | Gebruiker |
| Aannames | De actor bevindt zich op de inlog pagina |
| Basic Flow | 1. De actor voert zijn username en wachtwoord in en geeft in te willen loggen. 2. Het systeem controleert de gegevens en logt de gebruiker in. [1] |
| Uitzonderingen | 1. De gegevens kloppen niet. Laat een melding zien. |
| Resultaat | De actor is ingelogd. |

|  |  |
| --- | --- |
| Use Case 03 | Character maken |
| Actor | Speler |
| Aannames | De actor bevindt zich op de character creation pagina |
| Basic Flow | 1. De actor voert een naam, class en avatar in en geeft aan zijn character aan te willen maken. 2. Het systeem controleert de gegevens en maakt de character aan. De actor wordt doorgestuurd naar het character overzicht.[1][2][3] |
| Uitzonderingen | 1. De naam bestaat niet uit 4-15 characters. Geef een melding. 2. Er is geen class gekozen. Geef een melding. 3. Er is geen avatar gekozen. Geef een melding. |
| Resultaat | De actor heeft een character gemaakt. |

|  |  |
| --- | --- |
| Use Case 04 | Character kiezen |
| Actor | Speler |
| Aannames | De actor bevindt zich op de character overview pagina |
| Basic Flow | 1. Het systeem laat alle characters van de actor zien. 2. De actor kiest een character en geeft aan om verder te gaan. 3. Het systeem controleert of er een character gekozen is en stuurt de actor door naar de server pagina. [1] |
| Uitzonderingen | 1. Er is geen character gekozen. Laat een melding zien. |
| Resultaat | De actor heeft een character gekozen. |

|  |  |
| --- | --- |
| Use Case 05 | Server joinen |
| Actor | Speler |
| Aannames | De actor bevindt zich op de server pagina |
| Basic Flow | 1. De actor kiest ervoor om een server te joinen. 2. Het systeem laat een mogelijkheid zien om een servercode in te voeren. 3. De actor voert een code in en geeft aan deze server te willen joinen. 4. Het systeem controleert of de server bestaat of vol zit en stuurt de speler door naar de server lobby.[1][2] |
| Uitzonderingen | 1. Er is geen code ingevoerd, geef een melding. 2. De code klopt niet. Geef een melding. |
| Resultaat | De actor is een server gejoined. |

|  |  |
| --- | --- |
| Use Case 06 | Server aanmaken |
| Actor | Speler |
| Aannames | De actor bevindt zich op de server pagina |
| Basic Flow | 1. De actor kiest ervoor om een server te maken. 2. Het systeem maakt een server aan en plaatst de actor in de server lobby. |
| Uitzonderingen | - |
| Resultaat | De actor heeft een server gemaakt. |

|  |  |
| --- | --- |
| Use Case 07 | ReadyUp |
| Actor | Speler |
| Aannames | De actor bevindt zich in een server lobby |
| Basic Flow | 1. De actor geeft aan dat hij klaar is om het spel te starten. [1] 2. Het systeem verandert de status van de actor en toont dit aan de rest van de lobby. |
| Uitzonderingen | 1. Als de actor al klaar is verandert hij zijn status naar niet ready. |
| Resultaat | De actor is klaar om het spel te starten. |

|  |  |
| --- | --- |
| Use Case 08 | Lobby leaven |
| Actor | Speler |
| Aannames | De actor bevindt zich in een server lobby |
| Basic Flow | 1. De actor geeft aan dat hij de lobby wil verlaten. 2. Het systeem verwerkt de aanvraag en toont dit aan de rest van de lobby. |
| Uitzonderingen | - |
| Resultaat | De actor heeft de lobby verlaten. |

|  |  |
| --- | --- |
| Use Case 09 | Lobby disbanden |
| Actor | Admin speler |
| Aannames | De actor bevindt zich in een server lobby |
| Basic Flow | 1. De actor geeft aan dat hij de lobby wil disbanden. 2. Het systeem disband de lobby en geeft elke gebruiker een melding. |
| Uitzonderingen | - |
| Resultaat | De actor heeft de lobby disband. |

|  |  |
| --- | --- |
| Use Case 10 | Game starten |
| Actor | Admin speler |
| Aannames | De actor bevindt zich in een server lobby |
| Basic Flow | 1. De actor geeft aan het spel te willen starten 2. Het systeem controleert of iedereen ready is en start het spel.[1] |
| Uitzonderingen | 1. Niet iedereen is ready. Geef een melding. |
| Resultaat | De actor heeft het spel gestart. |

## 3.3 Use case Diagram

Hieronder is de use case diagram te zien.



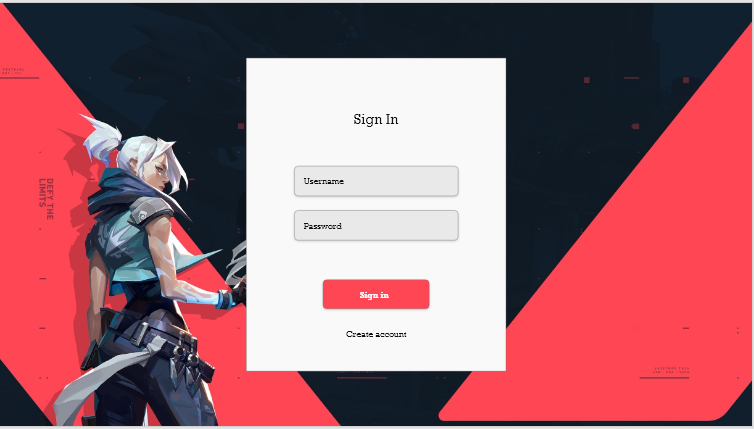
## 3.4 Systeem context



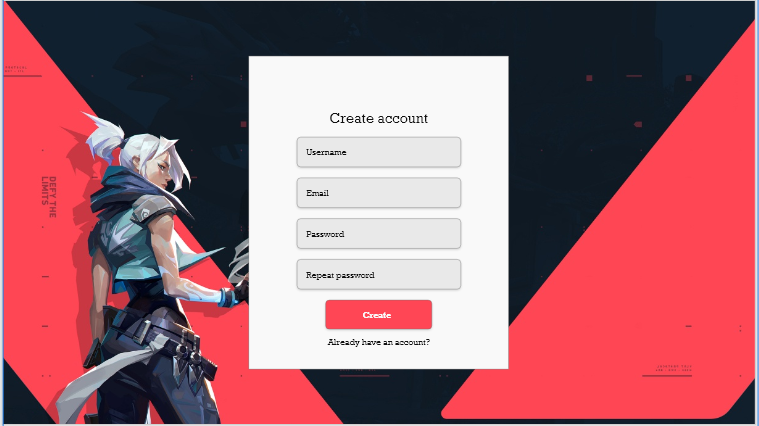
De applicatie bestaat uit 1 systeem en 1 database. Het systeem maakt contact met de gebruikers en de gebruikers met het systeem. Het systeem genereert ook mails die verstuurd kunnen worden naar de gebruikers.

## Wireframes

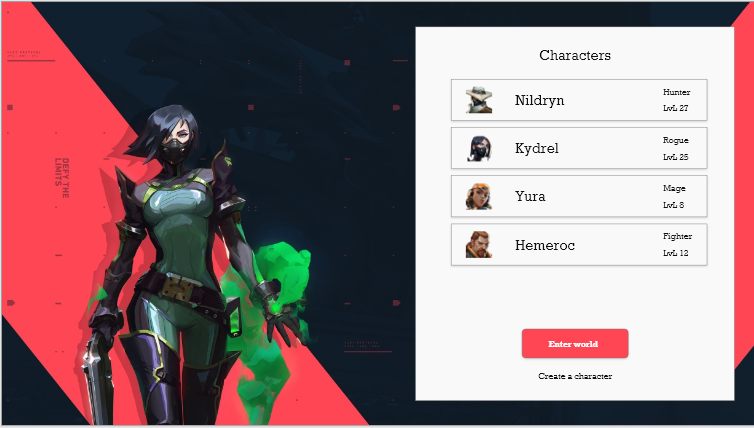
**Login**



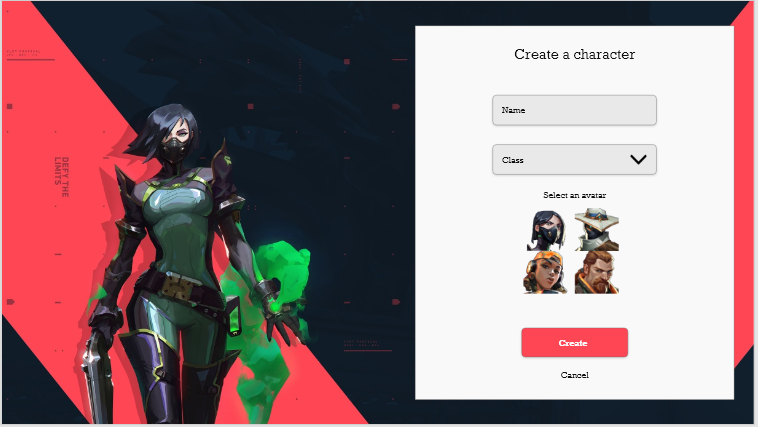
**Register**

****

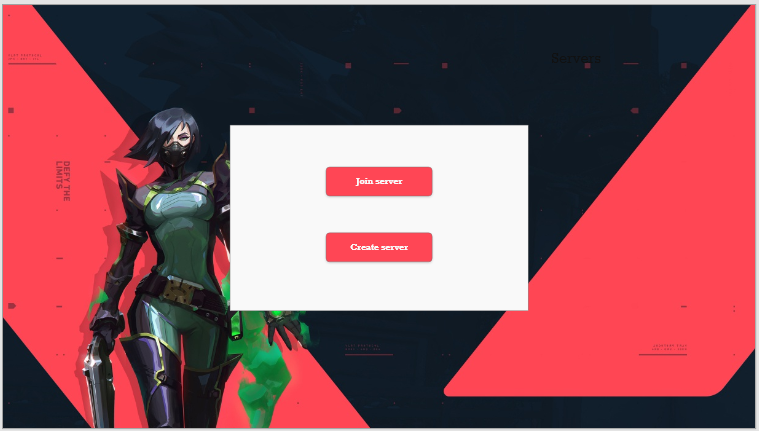
**Character overzicht**

****

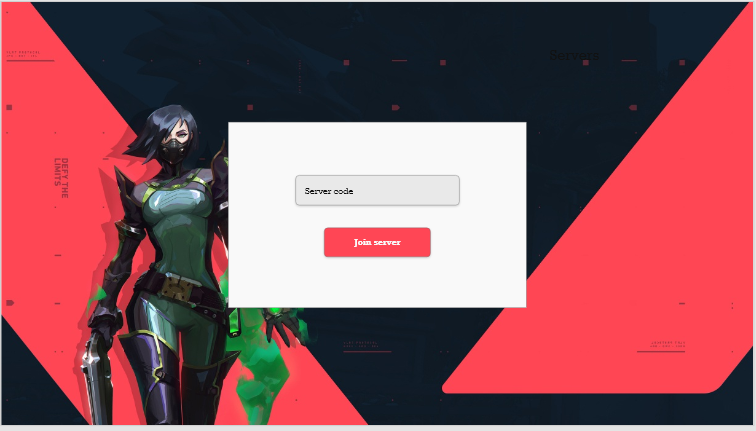
**Character creation**

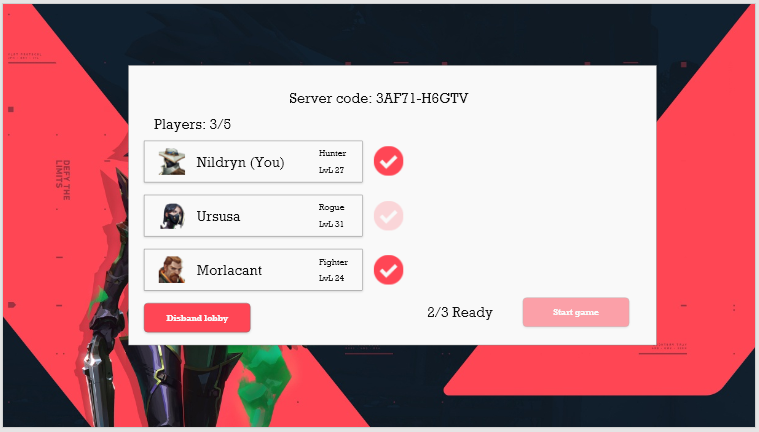
****

**Server options**

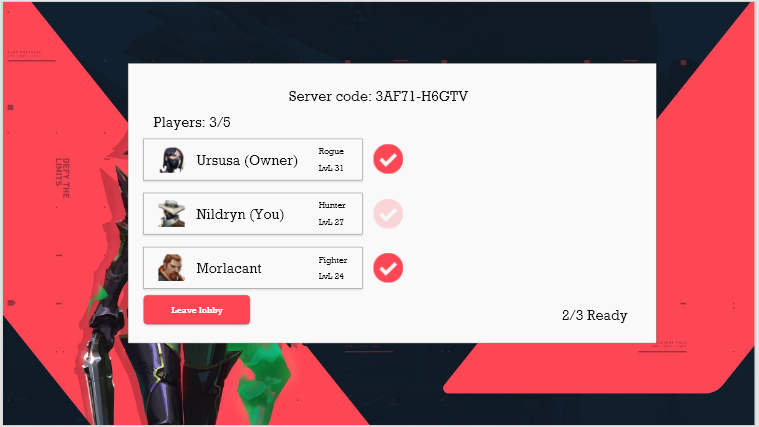
****

**Server joinen**

****

**Admin speler lobby**

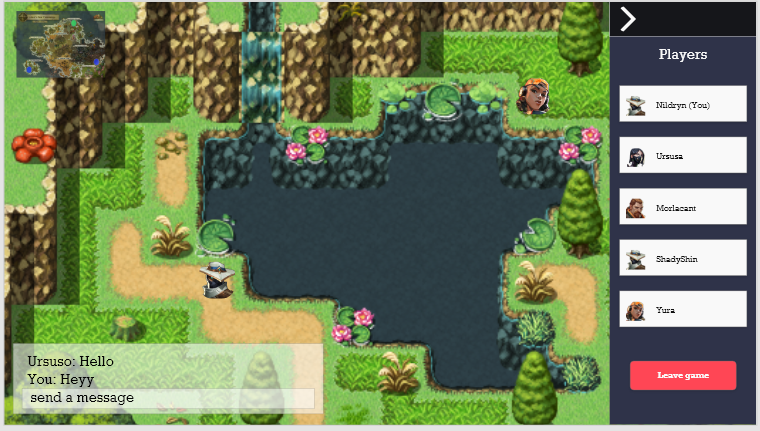
**Speler lobby**

****

**Gameworld**

**Game stats**

**Game party**

****

# 5. Logical view

## 5.1 Frameworks

In dit hoofdstuk wordt mijn onderzoek naar frameworks uitgelegd. Hierbij kies ik per onderdeel een framework op basis van dit onderzoek.

Ik heb deze frameworks op de volgende criteria beoordeeld:

Voorkennis: Hoe goed ken ik het framework? Heb ik er al mee gewerkt?

Documentatie: Hoe goed is het framework gedocumenteerd?

Ease of use: Hoe makkelijk is het framework te begrijpen/op te pakken?

Community: Hoe groot is de community?

Product Maturity: Is het platform nog in de begin fase?

**Front-end**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Criteria | Weight | React | Angular | Vue | Blazor | Asp.net |
| Documentatie | **10** | 6 | 7 | 8 | 8 | 10 |
| Product maturity | **5** | 10 | 9 | 6 | 6 | 9 |
| Community | **10** | 10 | 9 | 7 | 3 | 10 |
| Voorkennis | **6** | 1 | 5 | 7 | 9 | 9 |
| Ease of use | **10** | 6 | 4 | 9 | 9 | 0 |
| **Totaal** | **41** | 27.6 | 27.5 | 31.2 | 28.4 | 29.9 |

Uit de matrix hierboven komt Vue eruit als winnaar. Ik heb in semester vier kennis opgebouwd van vue maar die wil ik verder uitwerken. Om mij hierbij meer uit te dagen heb ik ook gekozen om er typescript bij te gebruiken.

**Backend**

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Weight | Java | .net core |
| Documentatie | **10** | 7 | 10 |
| Product maturity | **5** | 10 | 10 |
| Community | **10** | 10 | 10 |
| Voorkennis | **6** | 9 | 8 |
| Ease of use | **10** | 8 | 10 |
| **Totaal** | **41** | 35.4 | 38.6 |

Ik heb ervoor gekozen om met .net te gaan werken. Naast dat het hoger uit de matrix komt is voor mijn gevoel ook handiger om meer kennis hierover te hebben.

**Orm**

ik heb gekozen voor een ORM zodat ik code first een database kan opzetten. Dit zorgt ervoor dat het aanpassen en opzetten van de database geheel vanuit backend code gedaan worden in plaats van database scripts. Dit scheelt een hoop werk en is overzichtelijk.

Om een ORM te kiezen heb ik gekeken naar ORMs die aansluiten op mijn backend framework keuze .net core. Hierbij is Entity framework core de beste keuze.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Criteria | Weight | Entity framework core | NHibernate | Dapper | LINQtoDB |
| Documentatie | **10** | 10 | 10 | 10 | 6 |
| Voorkennis | **6** | 9 | 4 | 0 | 0 |
| Community | **10** | 10 | 5 | 3 | 1 |
| Ease of use | **10** | 8 | 7 | 4 | 7 |
| Product maturity | **5** | 10 | 10 | 7 | 6 |
| **Totaal** | **41** | 38.4 | 29.4 | 20.5 | 17 |

**Andere keuzes**

Als project management tool maak ik gebruik van azure devops. Hier houd ik ook mijn repository en pipelines bij. Ik heb hier een aantal semesters mee gewerkt en vind dit tot nu toe een heel fijn en flexibel programma.

**security**

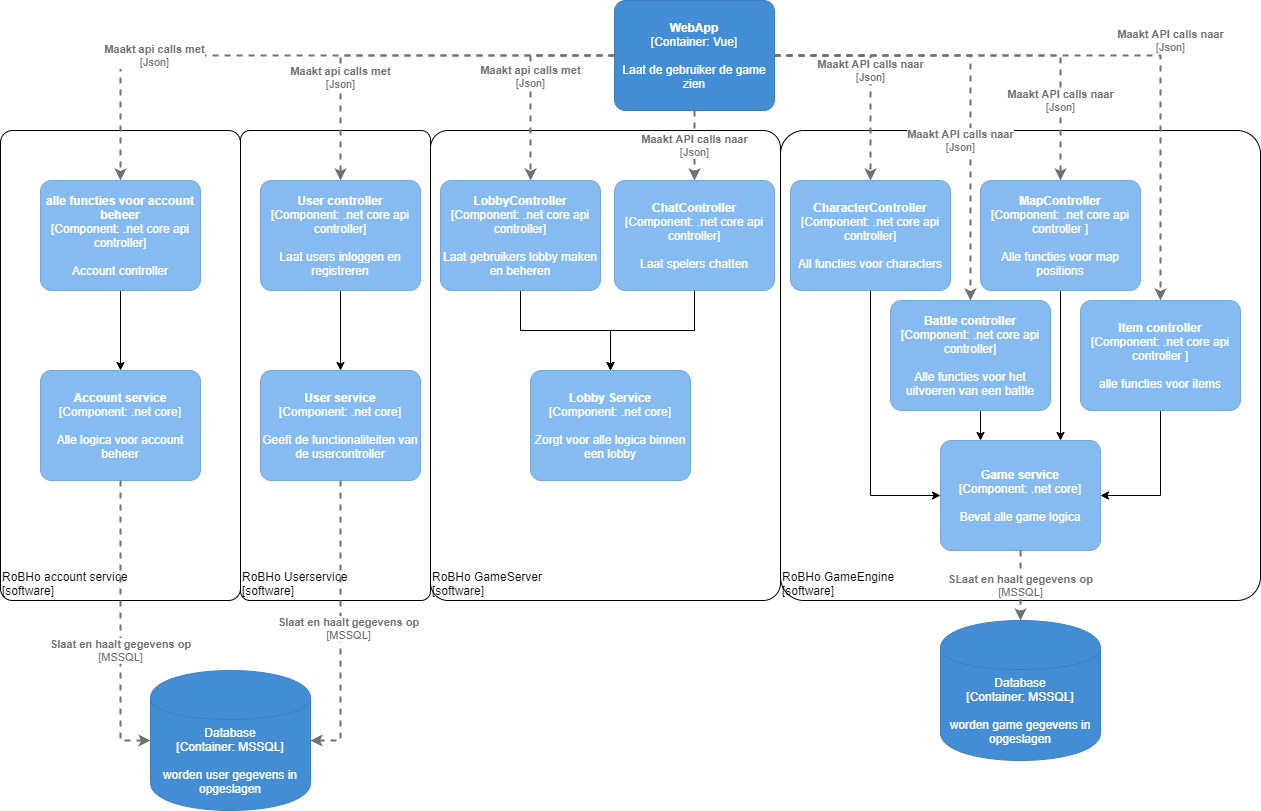
https://docs.microsoft.com/en-us/aspnet/core/security/data-protection/consumer-apis/password-hashing?view=aspnetcore-5.0

https://medium.com/dealeron-dev/storing-passwords-in-net-core-3de29a3da4d2

https://www.c-sharpcorner.com/article/hashing-passwords-in-net-core-with-tips/

https://medium.com/@nick\_92077/user-authentication-basics-hashing-and-jwt-3f9adf12272

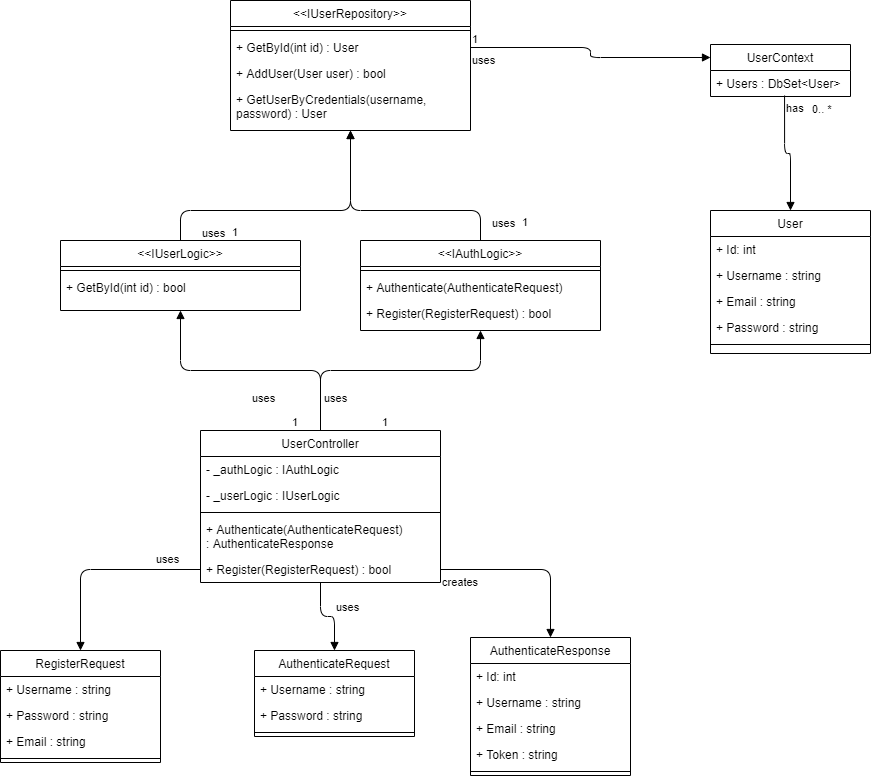
## 5.2 Component diagram



## 5.3 Code

Hieronder worden de klassendiagrammen voor elke service uitgewerkt. Hierin worden vooral de hoofdclasses uitgewerkt.

**User service**



## 5.4 Interfaces

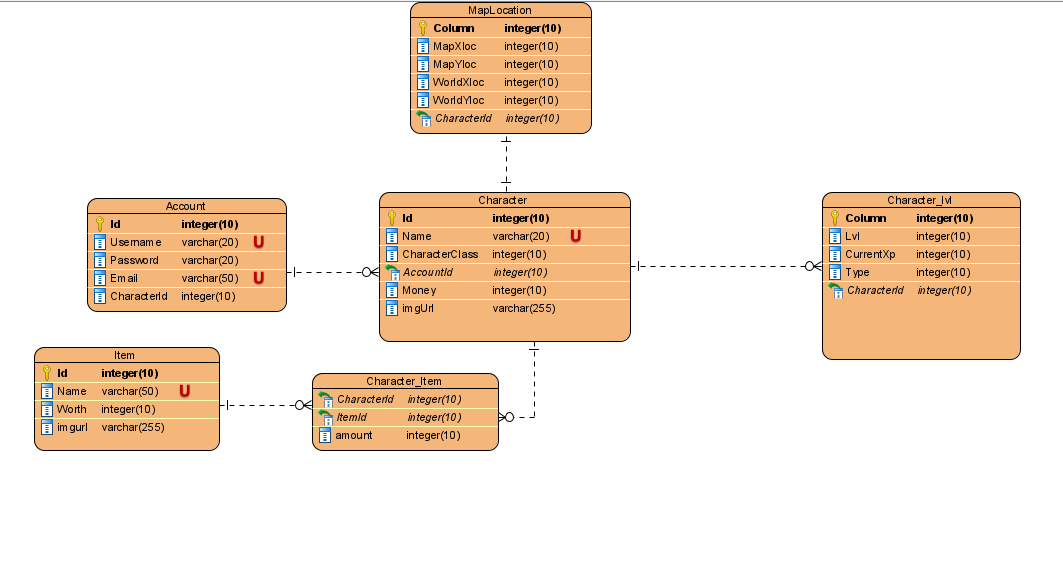
Hieronder worden alle interfaces uitgewerkt per service.

**User service**

|  |  |  |
| --- | --- | --- |
| Naam | Methode | Endpoint |
| Authenticate | POST | /authenticate |
| Register | POST | /register |
|  |  |  |

# 6. Data view

## 6.1 Database model



**Account**

Hierin worden de gegevens opgeslagen van je account. Met deze username en password kan je inloggen.

**Character**

Een account heeft meerdere characters. Hierin staan de standaard gegevens van een character.

**Character\_lvl**

Een character heeft meerdere character lvls. Het type lvl wordt aangegeven met de Type. Denk hierbij aan combat, crafting en gathering.

**Item**

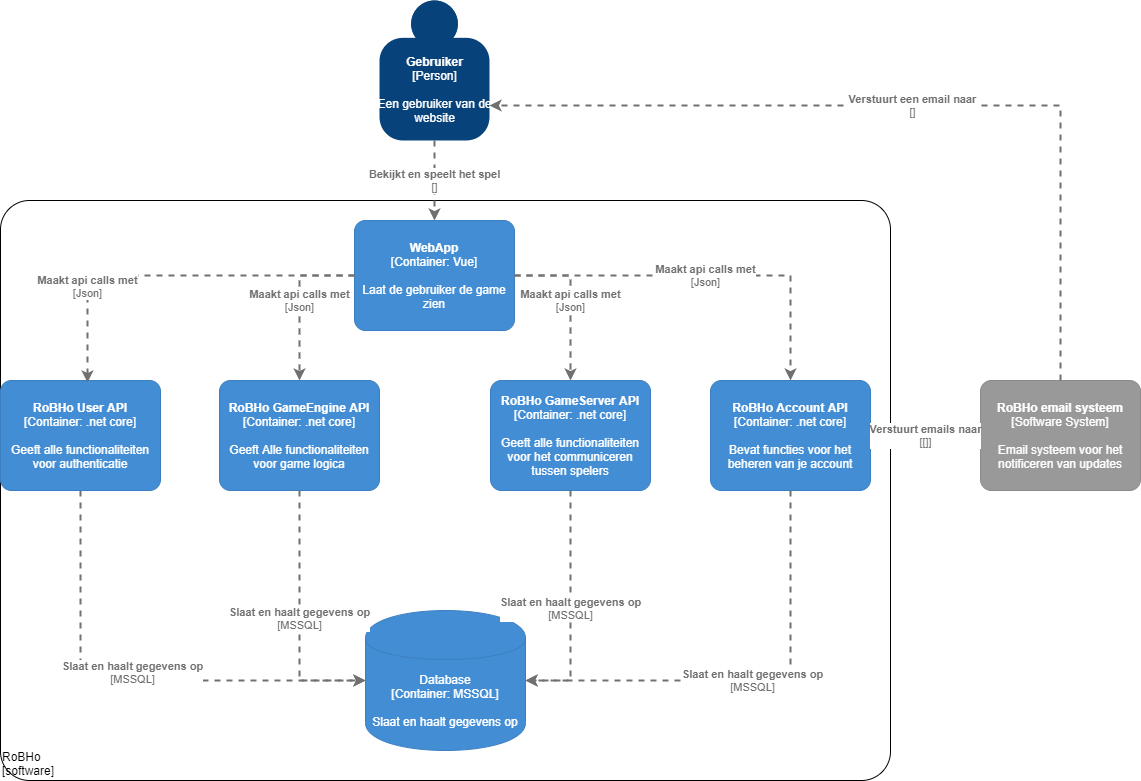
Een item heeft simpele gegevens zoals een naam, worth(gold) en img url. Een character kan meerdere items hebben en een type item kan bij meerdere characters voorkomen.

**MapLocation**

De maplocation slaat op waar een character zich bevindt op de wereld.

# 7. Deployment view

## 7.1 Deployment diagram



Het system is verdeeld over een website, database en 5 micro services.

**User service:** Deze service regelt alles m.b.t. de authenticatie en autorisatie.

**GameEngine service:** In deze service zit alle game logica.

**GameServer service:** In deze service wordt de communicatie tussen de verschillende spelers van een server geregeld. Dit stuurt ook updates van andere spelers naar een speler.

**Account service:** In deze service zit alle logica voor account beheer. Denk hierbij aan characters beheren.

**Mail service:** Deze service stuurt mails naar gebruikers voor updates.