19-11-2018

Door: Marco Jacobs

Software Architecture Document

“Big Idea” Semester 3 Software & FUN

Documenthistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Versienr. | Wijzigingen | Auteur | Datum |
| 1.0 | Eerste opzet document | Marco Jacobs | 19-11-2018 |
| 1.1 | Verder invulling | Marco Jacobs | 25-11-2018 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Inhoud

[H1 Inleiding 3](#_Toc530952481)

[H2 Systeem Context (C1) 4](#_Toc530952482)

[H3 Containers en technologiekeuzes (C2) 5](#_Toc530952483)

[H4 Componenten (C3) 6](#_Toc530952484)

[H5 Klassendiagram 7](#_Toc530952485)

[H6 Persistentie per component 8](#_Toc530952486)

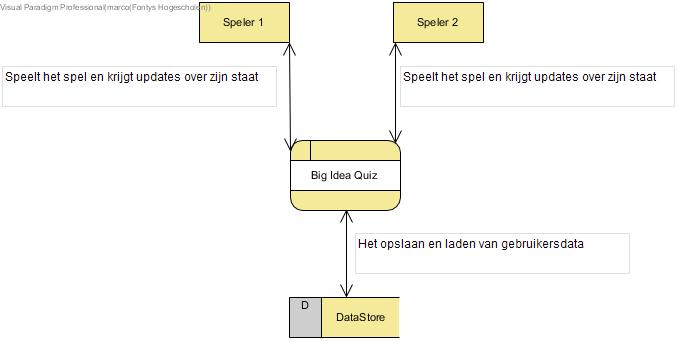
# H1 Inleiding

Voor semester 3 moeten we een big idea uitwerken. Hierbij zal de applicatie worden opgedeeld in verschillende componenten. De client, de authenticatie server en de game server. Om de data te kunnen bewaren over de games heen zal er gebruik worden gemaakt van een database waar deze informatie in wordt opgeslagen.

In dit document staat de architectuur van de software beschreven via diagrammen die zijn toegelicht met documentatie. Op deze manier wordt bepaald hoe de applicatie gebouwd gaat worden.

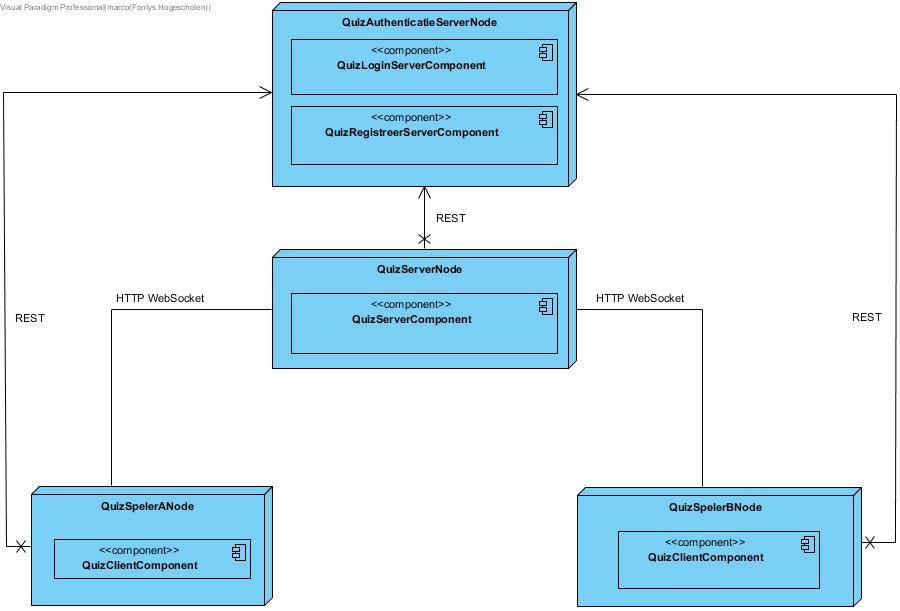
De (niet-)functionele requirements en het domeinmodel zijn aanwezig in het analysedocument die is ingeleverd via canvas.

# H2 Systeem Context (C1)



In bovenstaand diagram staat het systeem context van mijn quiz uitgewerkt. Hierin wordt de verbinding tussen de spelers met betrekking tot de server en database aangegeven.

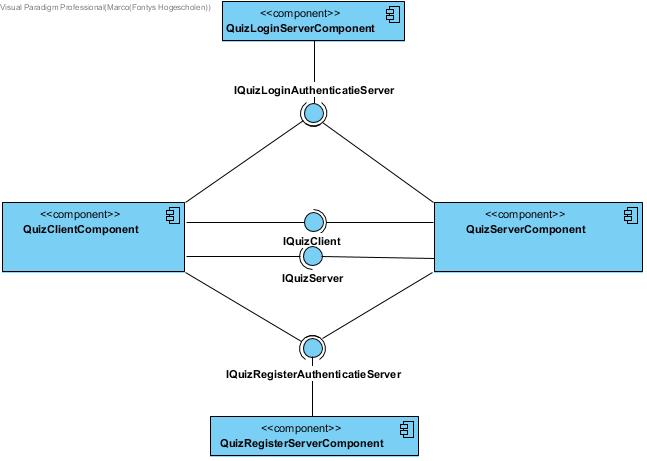
# H3 Containers en technologiekeuzes (C2)



In het bovenstaande deployment diagram staan de betrokken hardware systemen aangeven met bijbehorende componenten. Zoals in het diagram te zien is maak ik gebruik van 4 hardware systemen voor de applicatie. De authenticatie component wordt gebruikt voor de login en registratie om gebruikers te kunnen authenticeren. Vervolgens is er een game server die draait waaraan 2 clients gekoppeld kunnen worden. De communicatie tussen de servers zal via RESTful API gebeuren terwijl de communicatie tussen de server en de clients gebruik maakt van HTTP websockets.

De authenticatie van de gebruiker vind op 2 plaatsen plaats. Zowel tijdens het inloggen als het beginnen met een game wordt de speler gecontroleerd. Dit voer ik uit om een directe link tussen de gebruikers en de authenticatie voor login te creëren, maar tegelijk in de game zelf nog wel controleren dat de gebruikers toegang hebben tot het spel bijvoorbeeld bij een gecrackte versie.

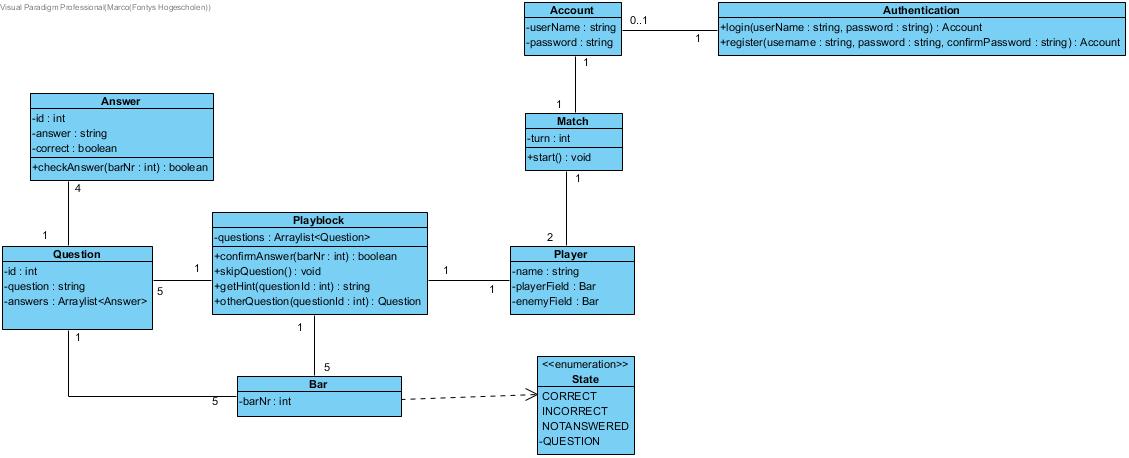
# H4 Componenten (C3)



Bovenstaand diagram is een UML Component Diagram. Hierin is te zien hoe de vier verschillende componenten met elkaar communiceren. De interfaces IQuizAuthenticatieServer is een RESTful API die wordt geleverd door de authenticatie server. De interface IQuizServer wordt geleverd door de spel server en is nodig voor de client. De interface IQuizClient wordt geleverd door de client en is nodig voor de game server. De interfaces IQuizServer en IQuizClient worden gerealiseerd door middel van WebSockets

Twee interfaces.

# H5 Klassendiagram



In bovenstaand diagram staat het klassendiagram beschreven zoals deze op de client gebruikt zal worden. Hierbij wordt rekening gehouden welke onderdelen verantwoordelijk zijn om de connectie met de verschillende servers op te zetten en te onderhouden.

# H6 Persistentie per component

