**MM systems 5 : Ganzenbord**

**1. Inleiding**

Voor het vak MM Systems 5 moesten we dit semester een multiplayer bordspel maken voor de Windows Phone in combinatie met Windows Azure. Wij, Kevin Heyvaert en Joris Huybrechts hebben de opdracht samen aangenomen en we hebben besloten om het klassieke 'Ganzenbord' te maken.

**2. Spelregels**

**Uitleg van** <http://nl.wikipedia.org/wiki/Ganzenbord>

Het bestaat uit een speelbord, een dobbelsteen en enkele pionnen van verschillende kleuren. De bedoeling van het spel is in zo weinig mogelijk beurten een pion van het begin naar het eind van een reeks velden te voeren, waarbij elke speler in elke beurt zijn pion zoveel velden moet verplaatsen als men ogen gooit met de dobbelstenen. Het speelbord is voorzien van een aantal speciale velden zoals de put en de gevangenis, die de speler hinderen op de weg naar zijn einddoel.

Ganzenbord heeft een speelvlak van 63 velden. Vakjes met een speciale betekenis zijn:

|  |  |  |
| --- | --- | --- |
| **nummer** | **naam** | **betekenis** |
| 6 | brug | Ga verder naar 12 |
| 19 | herberg | Een beurt overslaan |
| 31 | put | Wie hier komt moet er blijven tot een andere speler er opkomt. Degene die er het eerst was speelt dan verder. |
| 42 | doolhof of doornstruik | Terug naar 39 |
| 52 | gevangenis | Wie hier komt moet er blijven tot een andere speler er opkomt |
| 58 | dood | Terug naar begin, opnieuw beginnen |
| 63 | einde | Wie hier als eerste komt heeft gewonnen |

Wie te veel ogen gooit en daardoor voorbij 63 zou spelen, moet vanaf 63 weer terugspelen. Dit vergroot het risico dat men op het gevreesde hokje 58 of 52 terechtkomt. Komt men bij het terugtellen op een hokje met een gans, dan telt men weer het gegooide aantal ogen terug. Wie bijvoorbeeld op 60 staat en 7 gooit, komt in de gevangenis.

**3. Use Cases**

|  |  |
| --- | --- |
| **Use Case ID**: | 1 |
| **Use Case Name:** | Inloggen op spel |
| **Priority** | 2 |

|  |  |
| --- | --- |
|  |  |
| **Description:** | Inloggen met username + password op spel. Username met bijhorend paswoord wordt gecontroleerd met de data die per speler in de data base staat. |
| **Trigger:** | The need for playing a bord game |
| **Preconditions:** | Internetconnectie + gebruiker zit in de database |
| **Postconditions:** | Ingelogd op de applicatie |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding / verbinding met database / user bestaat niet |

**Alternatieve flow**

**A. Wat als username/password combination incorect is**

1. System validation of the username/password combination fails due to incorrect entry.

2. Systems asks the user to re-enter the username/password combination.

3. Go back to basic flow met ID nummer 1.

**B. Gebruiker bestaat niet in de database**

1. System validation finds that the user record does not exist in the database.

2. Systems alerts the user that their record does not exist.

|  |  |
| --- | --- |
| **Use Case ID**: | 2 |
| **Use Case Name:** | Zien van speloverzicht |
| **Priority** | 1 |

3. Go back to basic flow met ID nummer 1.

|  |  |
| --- | --- |
| **Description:** | UI : zien van het bordspel op scherm : uitvergroot en in detail + alle andere spelers zien (positie) |
| **Trigger:** | The need for playing a bord game |
| **Preconditions:** | Inloggen op de applicatie + verbinding met de database |
| **Postconditions:** | Spelen van het bordspel met meerdere spelers + het spel volgen |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding of geen verbinding met database |

Voor verder in detail te gaan van het bordspel, hebben we gedacht om het "bord" in stukken (vierkanten) te verdelen en elk een chronologisch nummer te geven. Want in een bordspel wordt meestal een cyclus doorlopen in 1 richting. Dus zo kan men beter de volgende stap op het scherm tonen. Het is eigenlijk bijhouden van plaatsen in een stijgende lijn. Want anders is het onmogelijk om het (volledige) spel te volgen op een klein scherm van een smartphone.

|  |  |
| --- | --- |
| **Use Case ID**: | 3 |
| **Use Case Name:** | Eigen zet doen |
| **Priority** | 1 |

|  |  |
| --- | --- |
|  |  |
| **Description:** | Speler kan dobbelen op zijn beurt en kan een zet doen |
| **Trigger:** | Vorige speler heeft zijn beurt afgerond |
| **Preconditions:** | Verbinding met database + beurt van de speler |
| **Postconditions:** | Speler heeft zijn zet gedaan : bijhorende acties bij die zet worden gedaan ( pion kwijt, plaatsen terug gezet,...) + beurt aan volgende speler |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding of geen verbinding met database |

Voor het 'soepele' verloop van het spel op de Windows Phone, gaan we moeten bepalen hoe we de data naar de database gaan pushen en hoeveel keren we data gaan vragen aan de database. We moeten data krijgen om te zien wat de andere spelers hebben gedaan in het spel. En voor een snel (en soepel) verloop, gaan we dat vragen op bepaalde tijdstippen of bij bepaalde handelingen ( vb : bij maken van zet van een andere speler)

|  |  |
| --- | --- |
| **Use Case ID**: | 4 |
| **Use Case Name:** | Zetten van andere spelers zien en binnenkrijgen |
| **Priority** | 1 |

|  |  |
| --- | --- |
|  |  |
| **Description:** | Zien wie aan de beurt is + wat voor zet ze hebben gedaan |
| **Trigger:** | Eigen beurt is verlopen |
| **Preconditions:** | Eigen beurt is verlopen en zet is naar data base gepusht |
| **Postconditions:** | Zien wat het verdere spelverloop is + eigen beurt |
| **Normal Flow:** |  |
| **Exceptions:** | Geen connectie met de data base |

Hier gaan we bepalen wat we gaan doen als men het einde van het spel bereikt of klaar is met spelen.

|  |  |
| --- | --- |
| **Use Case ID**: | 5 |
| **Use Case Name:** | Einde spel aangeven + 'score's' van tegenspelers |
| **Priority** | 2 |

|  |  |
| --- | --- |
|  |  |
| **Description:** | Einde van spel tonen met winnaar en verliezers ( met bijhorende eindplaatsen en aantal zetten?) |
| **Trigger:** | Einde van spel of dichste bij einde |
| **Preconditions:** | Spel is bijna uitgespeeld |
| **Postconditions:** | Spel afsluiten + winnaar en verliezers tonen (met bijhorende plaatsen) + (Win/verlies bijhouden in database) |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding of geen verbinding met database |

We kunnen ook de score van de verschillende spelers laten zien in een aparte pagina van de applicatie. We pullen de data per speler en laten die zien in een lijst : Van hoge score naar lage score.

|  |  |
| --- | --- |
| **Use Case ID**: | 6 |
| **Use Case Name:** | Tonen van score lijst |
| **Priority** | 4 |

|  |  |
| --- | --- |
|  |  |
| **Description:** | Tonen van spelers en hun aantal win/verlies tonen in lijst |
| **Trigger:** | Einde van spel of aparte pagina/toets in applicatie |
| **Preconditions:** | Klikken op pagina 'score list' + internetconnectie met database |
| **Postconditions:** | Tonen van 'scores' per speler |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding of geen verbinding met database |

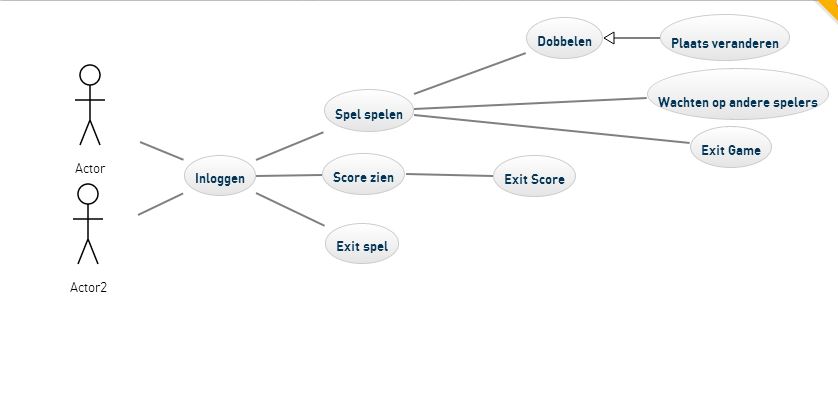
|  |  |
| --- | --- |
| **Use Case ID**: | 7 |
| **Use Case Name:** | Exit |
| **Priority** | 1 |

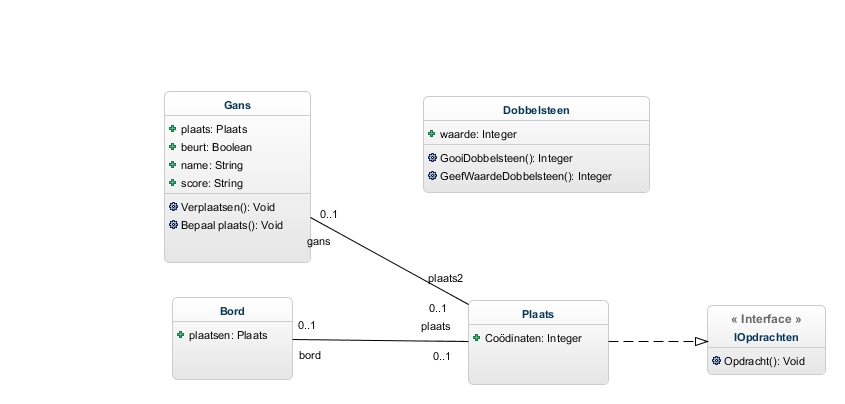
|  |  |
| --- | --- |
|  |  |
| **Description:** | Gebruiker kan het spel verlaten met exit button |
| **Trigger:** | Indrukken van 'exit' toets |
| **Preconditions:** | Speler is ingelogd |
| **Postconditions:** | Speler is niet meer ingelogd |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding of geen verbinding met database |

|  |  |
| --- | --- |
| **Use Case ID**: | 8 |
| **Use Case Name:** | Dobbelsteen gooien |
| **Priority** | 1 |

|  |  |
| --- | --- |
|  |  |
| **Description:** | Spel vraagt om dobbelsteen te gooien en speler kan gooien. |
| **Trigger:** | Indrukken van 'gooi' knop |
| **Preconditions:** | Speler is aan de beurt |
| **Postconditions:** | Speler krijgt een nummer tussen 1-6 of 2-12 |
| **Normal Flow:** |  |
| **Exceptions:** | Geen internetverbinding of geen verbinding met database  Speler krijgt geen nummer / server heeft geen nummer opgeslagen |

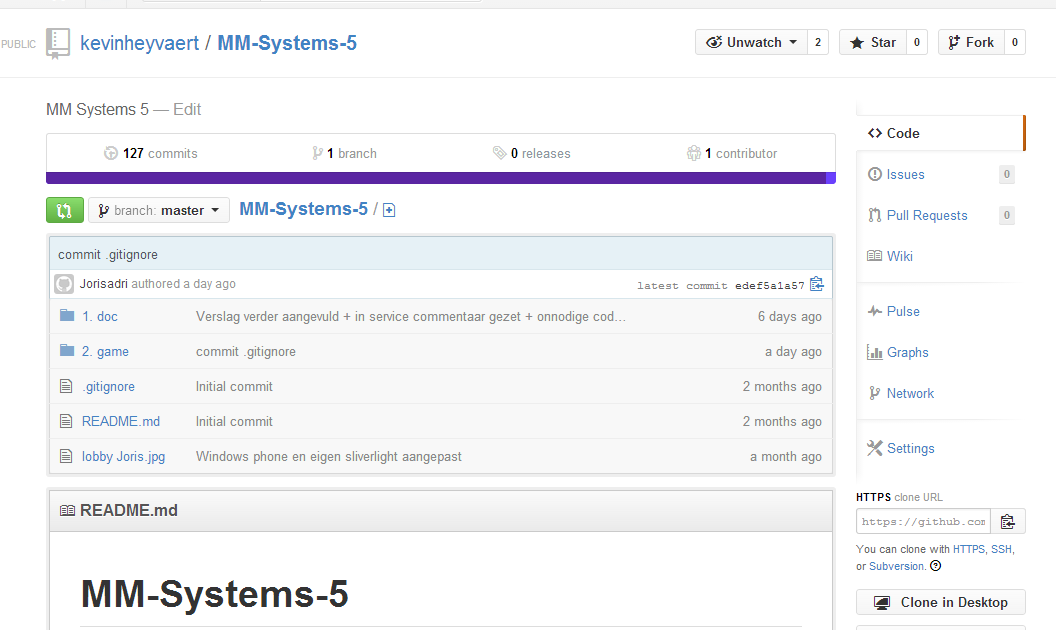
Hieronder staan de use cases nog visueel voorgesteld met een use case met actors + een klassediagram.

****

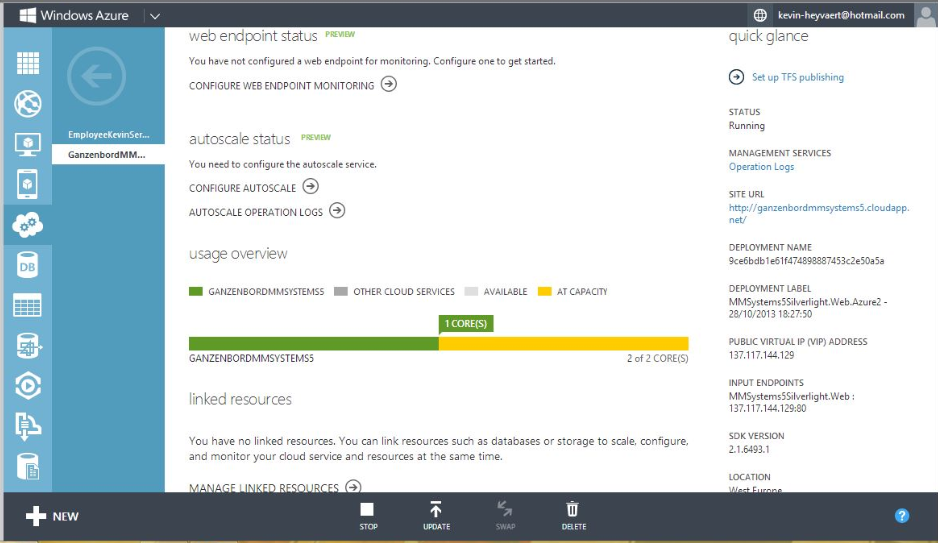
****

**5. Github**

Voor het maken van ons spel hebben we gebruik gemaakt van Visual Studio en C#. Om de vooruitgang op te slaan, maken we gebruik van Github. Onze repository kan je vinden op deze link : <https://github.com/kevinheyvaert/MM-Systems-5>



**5. Service**

****

Wij hebben een service op Windows Azure gemaakt voor ons ganzenbordspel. Deze hebben we GanzenbordMMSystems5 genoemd. In deze service zetten we methodes die we niet nodig hebben op de phone. De service koppelen we aan de aan de phone applicatie. Daardoor kan de phone applicatie alle methodes in de service aanroepen. Deze worden aan geroepen met een event.

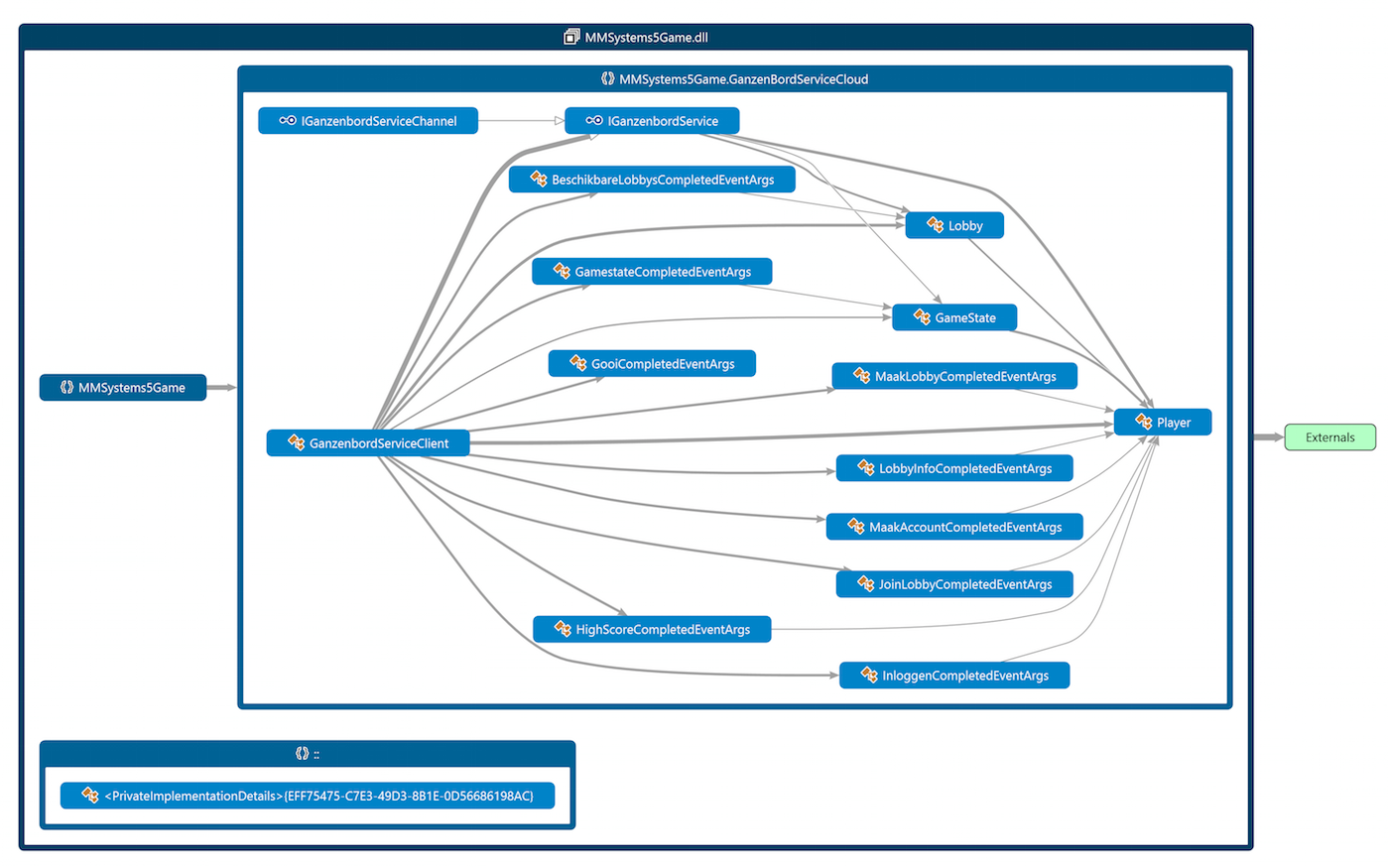
* List<int> Gooi(DTO.Player player);
* Gooien van dobbelsteen en bepalen van de locatie speler
* DTO.Player Inloggen(string naam, string wachtwoord);
* Inloggen van een speler
* DTO.Player MaakAccount(string PlayerNaam, string Wachtwoord);
* Maken van een account voor een speler
* List<DTO.Lobby> BeschikbareLobbys();

**🡪** Geeft een lijst van beschikbare lobby’s weer

* DTO.Player MaakLobby(DTO.Player player);
* Maken van een nieuwe lobby
* DTO.Player JoinLobby(DTO.Lobby lobby, DTO.Player player);
* Het joinen van een lobby
* void ExitLobby(DTO.Player player);
* Het verlaten van een lobby
* void Start(DTO.Lobby lobby);
* Het starten van het spel
* DTO.GameState Gamestate(DTO.Player player);
* Het opvragen van de gamestate : locatie van andere spelers, beurten,…

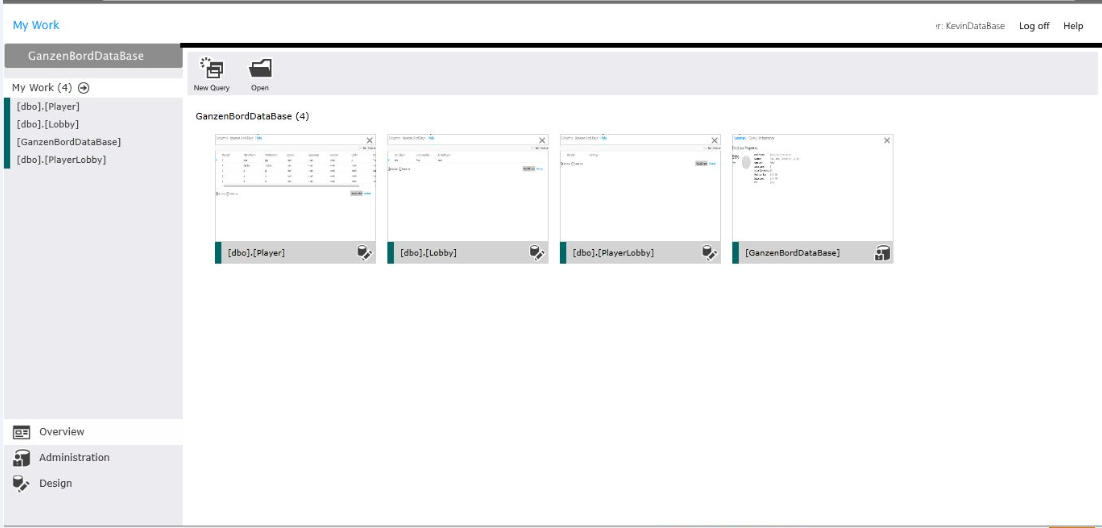
Er zijn ook nog private methodes in de service die meewerken met bovenstaande methodes maar deze zijn niet zichtbaar voor de phone applicatie :

* private void plaatsgame(DTO.Player player)
* private void brug(DTO.Player player)
* private void herberg(DTO.Player player)
* private void put(DTO.Player player)
* private void doornstruik(DTO.Player player)
* private void jail(DTO.Player player)
* private void dead(DTO.Player player)
* private void Einde(DTO.Player player)
* private void updatelobby(int lobby)
* private void updateaantal(int lobby, bool canjoin)
* private void next(DTO.Player player)



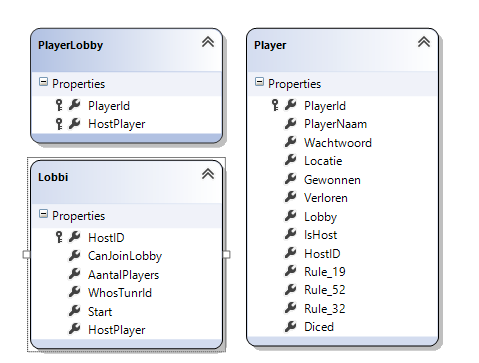
Hierboven zie je een schema van de beschikbare functies die onze service bevat en waar onze cliënt gebruik van maakt

**6. Database**

****

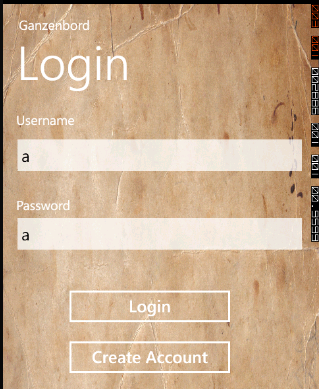
Voor het spel hebben we ook een database gemaakt om data (omtrend het spel) op te slaan. Hieronder vindt je de verschillende lijsten die we in de database gebruiken :

* **Player** : PlayerId, PlayerNaam, Wachtwoord, Locatie, Gewonnen, Verloren, Lobby, IsHost, HostID, Rule 19, Rule 52, PlaceGame
* **Lobbi** :HostPlayer, Can/JoinLobby, AantalPlayers, HostID, WhosTurnId, Start
* **PlayerLobby** : Playerid, HostPlayer
* **Gamestate** : ID, Update, YourTurn
* **Update** : HostName



**7. Ganzenbord**

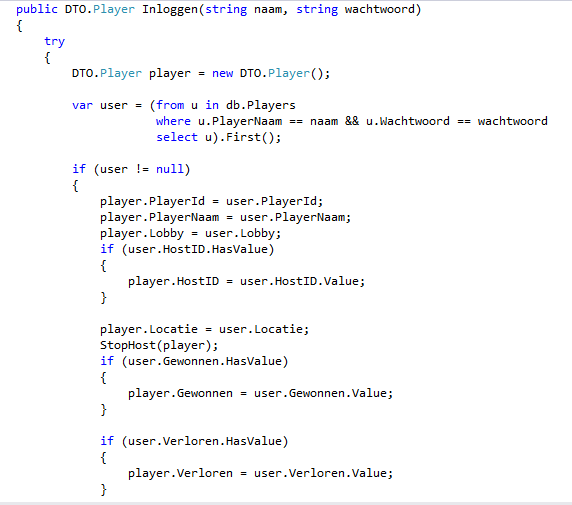
**7.1 Inloggen**



* Inloggen : met username en paswoord

Wordt gecontroleerd met query in de database voor te zien of deze erin staat en dan gaat de applicatie naar de main view.

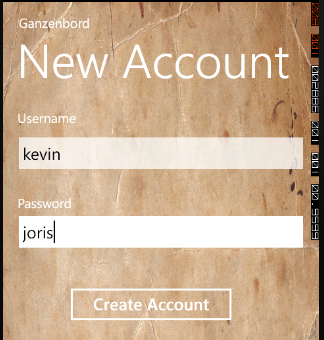
* Create account : maak nieuw account



Hiernaast zie je de code die in de service staat voor het inloggen. Deze gaat met een query kijken of de ingevoerde username en paswoord in de database staat en gaat dan deze ophalen en zijn onderdelen zoals de naam, lobby en score.

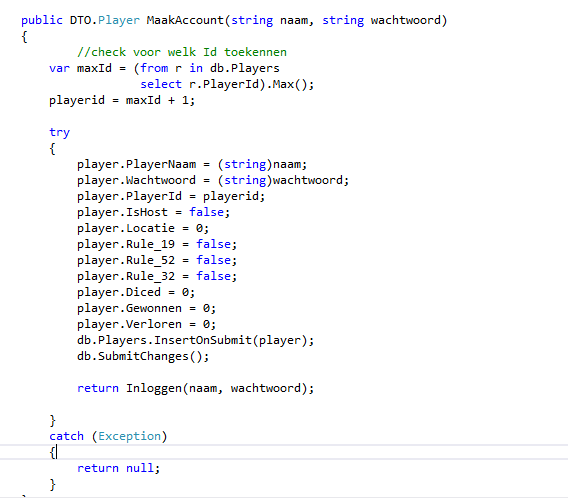
Ook extra aspecten worden veranderd : locatie van vorige spellen (wordt voor de zekerheid) terug op 0 gezet + de regels voor bepaalde plaatsen op het spelbord op “false” gezet. Ook eventuele lobby’s dat speler had aangemaakt, worden dan stopgezet. Dit gebeurd met de functie “StopHost”.

**7.2 Maken van een nieuw account**



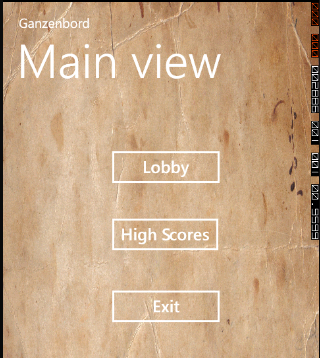
* Maak nieuw account

--> in de textboxen wordt er een nieuwe username en wachtwoord ingegeven en deze worden dan met een query in de data base gezet. En dan gaat de applicatie inloggen en naar main view gaan.



Hiernaast zie je de code die de telefoon aanroept met de service om een nieuw account aan te maken. Hierbij gaan ze zoeken eerst zoeken naar het volgende ID dat ze kunnen toewijzen in de database. Dan worden alle eigenschappen van de player toegevoegd aan een account.

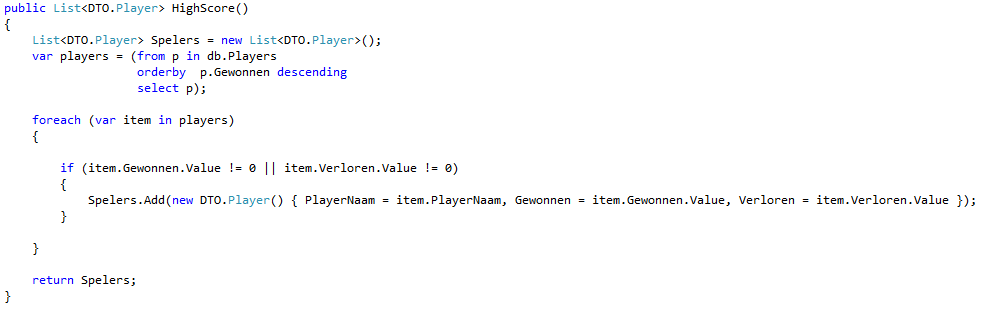
**7.3 Main view**



* Lobby : ga naar lobby pagina
* High scores : toon high scores van alle spelers in een lijst. (met behulp van query alles in een list krijgen)
* Exit : ga terug naar inlog pagina

**7.4 High Scores**

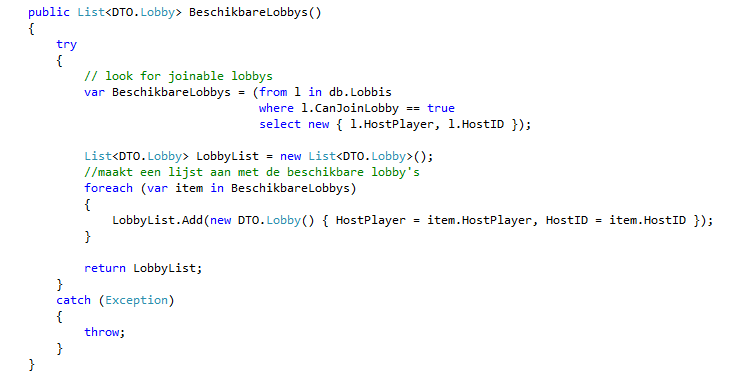
* Na laden van pagina komt er een list van alle spelers met hun gewonnen en verloren spelletjes (met query)
* Back : ga terug naar main view



**7.5 Lobby**

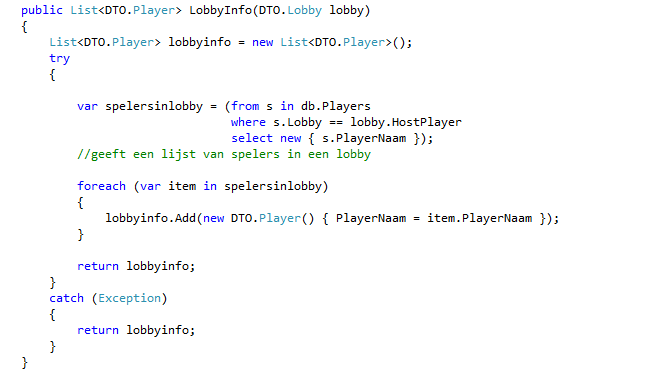


* Bij laden van de pagina komen de beschikbare lobby's tevoorschijn in de linkse listbox. Lobby namen zijn de namen van de hostplayers die de lobby hebben aangemaakt.



* Als men op 1 van de items duwt in die list (vb. Textbox) dan komt er in de rechtse list een lijst met de spelers die in die lobby zitten ( host + max 3 andere spelers)

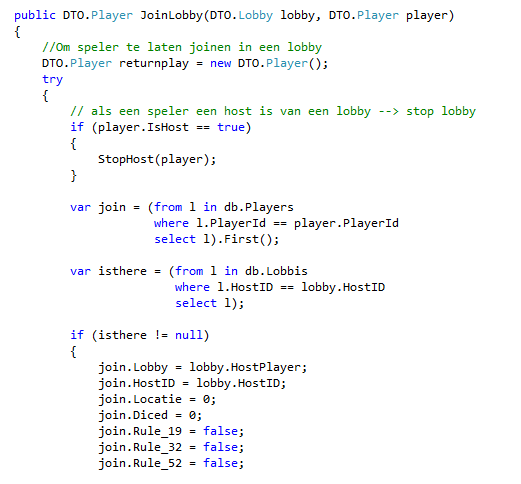




* Als de host kiest voor een andere lobby( join), dan zal zijn (al dan niet) aangemaakte lobby verdwijnen en dan zal deze in de lobby zitten van een andere hostplayer

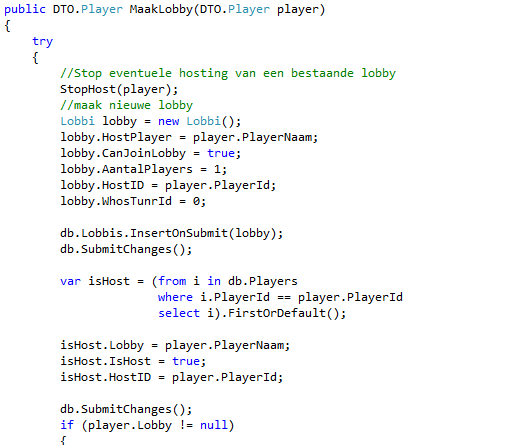
--> met query de lobby property van de speler veranderen + 'ishost' property veranderen)

Ook zullen weer de regels voor de bepaalde plaatsen op “false” worden gezet en de locatie op 0.

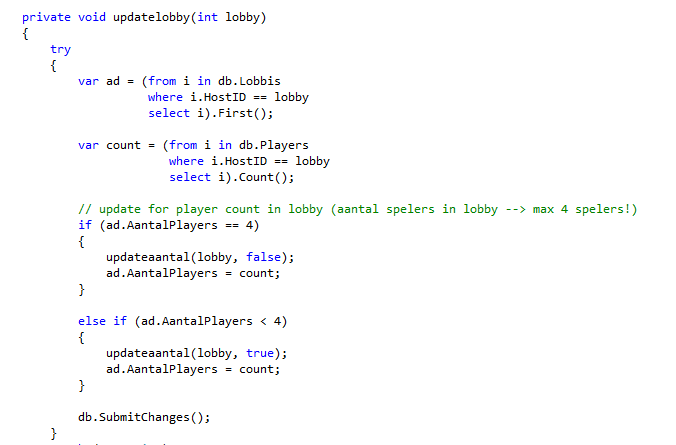


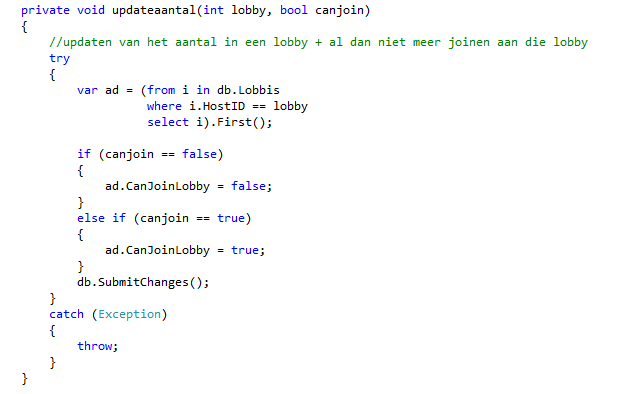
# Create lobby : speler zal een lobby aanmaken die de naam van de hostplayer krijgt.

--> met query de lobby property van de speler veranderen + ishost property veranderen)

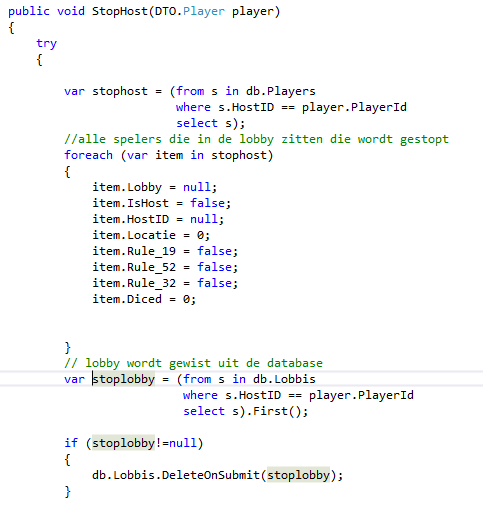


In deze functie wordt ook altijd de “updatelobby” aangeroepen om het aantal spelers in de lobby (max 4) te controleren. Als deze het maximum van 4 heeft bereikt, dan zal deze het joinen van een lobby niet meer mogelijk zijn. Deze zal de functie “updateaantal” daarvoor gebruiken.

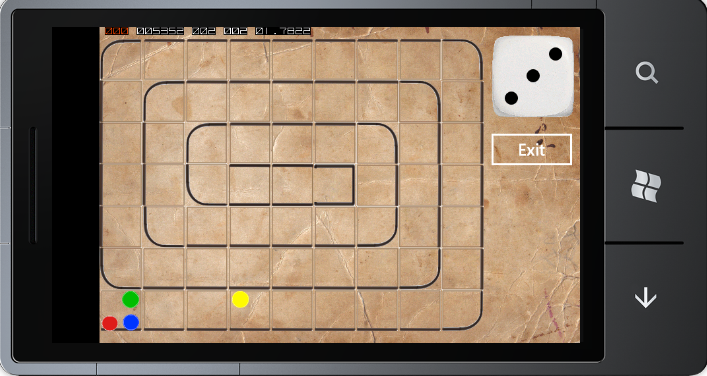




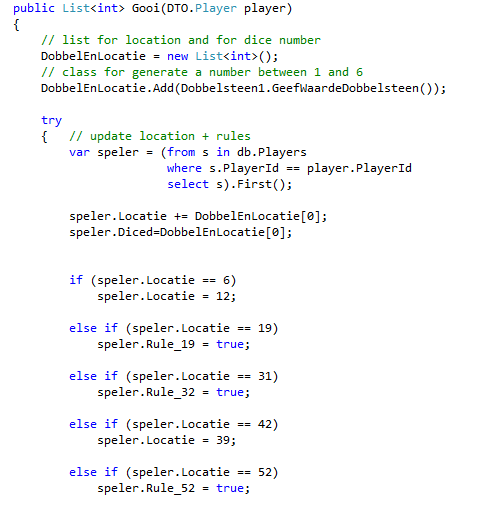
Hieronder vind je de code voor de host van een lobby te stoppen.

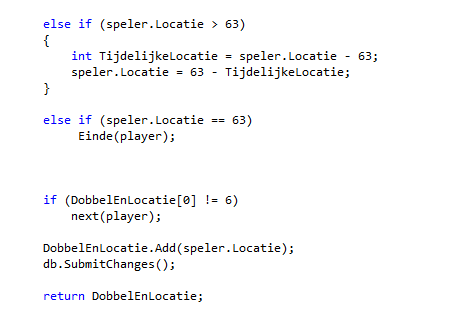


**7.6 Spelbord**

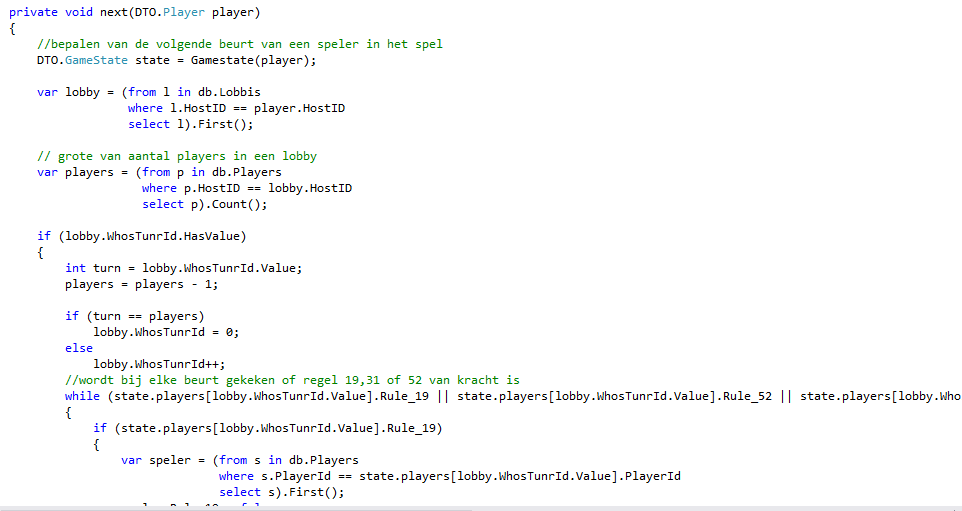


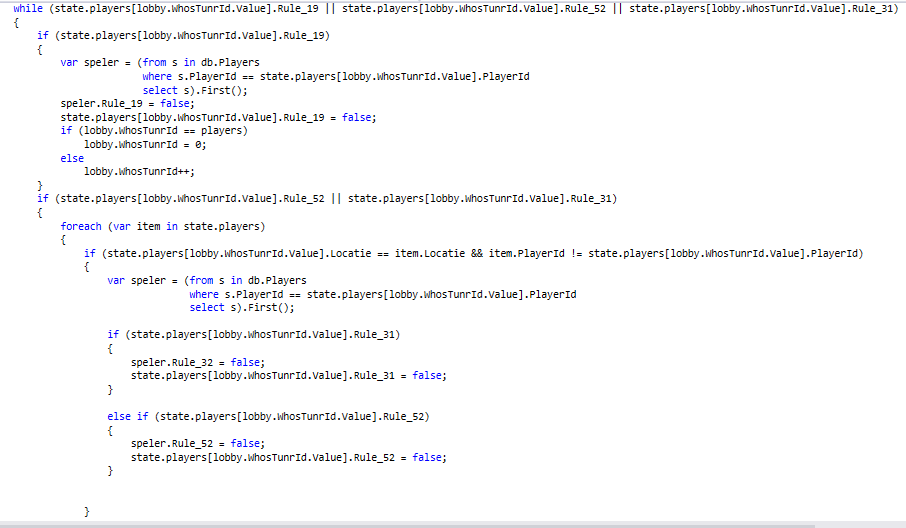
* Dobbel : dobbelsteen aanroepen via service en via service ook de plaats bepalen van de speler. Op deze manier altijd de game status property's aanpassen. En dan deze data (zoals locatie) pushen naar de database.



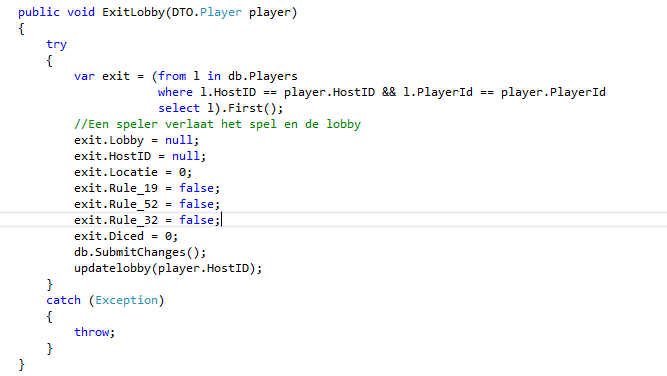


Ook wanneer de speler blijft 6 gooien, blijft deze aan de beurt. Als de speler geen 6 gooit, worden zijn stappen gedaan en gaan we naar de volgende speler m.b.v. de “next” functie.



Deze bovenstaande functie gaat de Gamestate (lijst van players) checken en gaat zo bepalen welke speler aan de beurt is. Daarna gaat deze functie bekijken of de regels 19,31 of 52 van kracht zijn op de speler die aan de beurt is.

* Met de exit knop kan je het spel verlaten



**8. Demo**

Hieronder kan je zien dat we het spel in een testfase aan het spelen zijn met 3 spelers.

