2022

Hassan Mahmoud – 2065740

Joes Boymans – 2104080

Lotte Soudant – 1624318

Gamification

Casusgroep 4

Inhoud

[Samenvatting 2](#_Toc107781653)

[Inleiding 3](#_Toc107781654)

[Requirements 4](#_Toc107781655)

[Functioneel 4](#_Toc107781656)

[Niet Functioneel 6](#_Toc107781657)

[Use-Case Diagram 7](#_Toc107781658)

[Use-Case Beschrijvingen 8](#_Toc107781659)

[UCB1: Spel spelen 8](#_Toc107781660)

[UCB2: Categorieën maken 9](#_Toc107781661)

[UCB3: Vragen maken 10](#_Toc107781662)

[UCB4: Spelers Beheren 11](#_Toc107781663)

[UCB5: Vragen Beheren 12](#_Toc107781664)

[Klassendiagram 13](#_Toc107781665)

[Sequencediagram 14](#_Toc107781666)

[ERD (Entiteit-Relationship Diagram) 0](#_Toc107781667)

[LGS (Logische GegevensStructuur) 0](#_Toc107781668)

[Component Diagram 2](#_Toc107781669)

[State-Machine Diagram 3](#_Toc107781670)

[Deployement Diagram 4](#_Toc107781671)

[Wireframes 5](#_Toc107781672)

# Samenvatting

# Inleiding

Voor de casus van blok 4 moet er een desktop applicatie ontwerpt worden en (deels) gerealiseerd. Het doel van deze app is om ervoor te zorgen dat studenten zelfstandig of in groepsverband kennis eigen kunnen maken. Er zal een serious game uitgewerkt moeten worden. Als basis van deze game nemen we een bestaand kaartspel/bordspel.

In dit bestand zal de benodigde documentatie komen te staan van het project. Doormiddel van de wekelijkse discussiecolleges en werkcolleges wordt er kennis opgedragen om de opdrachten kunnen uit te voeren en een goed product op te stellen.

# Requirements

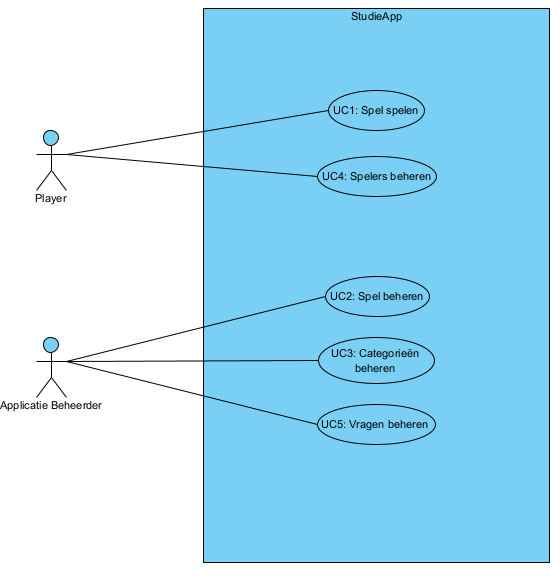
## Functioneel

|  |  |  |  |
| --- | --- | --- | --- |
| # | Requirement | Prioritering | Geaccepteerd |
| F1.M | Elke speler moet eigen naam invullen | Must |  |
| F2.M | Een speler moet op een vak | Must |  |
| F3.M | Een speler kan een vak kopen door het correct beantwoorden van een vraag en het betalen van het benodigde bedrag | Must |  |
| F4.M | Een speler moet huur betalen wanneer hij op het vak van een andere speler terecht komt | Must |  |
| F5.M | Een speler hoeft geen huur te betalen indien hij vragen correct beantwoord | Must |  |
| F6.M | Elke speler begint het spel met dezelfde hoeveelheid geld | Must |  |
| F7.C | De speler die het hoogste getal gooit met de dobbelsteen mag beginnen | Could |  |
| F8.C | Applicatie zorgt ervoor dat ronden met de klok mee gespeeld | Could |  |
| F9.M | Een speler moet een dobbelsteen met 6 zijdes kunnen rollen | Must |  |
| F10.C | Een speler moet geld krijgen wanneer hij opnieuw het startvak passeert | Could |  |
| F11.M | Een speler moet een winnaar worden wanneer hij zijn beurt start en alle andere spelers geen geld meer hebben | Must |  |
| F12.M | Een speler moet een verliezer worden wanneer hij zijn beurt start en hij geen geld meer heeft | Must |  |
| F13.M | Een verliezer mag geen beurten meer nemen | Must |  |
| F14.M | Een speler kan andere spelers uitnodigen om te spelen | Must |  |
| F15.M | Applicatiebeheerder kan de vraagcategorieën per spel beheren | Must |  |
| F16.M | Een speler moet een dobbelsteen gooien om een beurt te kunnen spelen | Must |  |
| F17.M | Een speler moet een kaart rapen om een bepaald effect in werking te zetten (kan voordelig of nadelig zijn) | Must |  |

## Niet Functioneel

|  |  |  |
| --- | --- | --- |
| # | Requirement | Geaccepteerd |
| NF1 | Het spel bevat een vierkant spelbord |  |
| NF2 | Het spel bevat minimaal 2 spelers en maximaal 4 spelers |  |
| NF3 | Het spel bevat 16 kanskaarten |  |
| NF4 | Het spel bevat 16 algemeen fondskaarten |  |
| NF5 | Het spelbord bevat vakken rondom de rand van het spelbord |  |
| NF6 | Het spelbord bevat 3 verschillende soorten spelvakken, die aangeeft hoeveel correcte vragen beantwoord moet worden |  |
| NF7 | Het spelbord bevat speciale vakken met ander gedrag |  |
| NF8 | Een speler moet speelgeld kunnen bezitten |  |

# Use-Case Diagram



# Use-Case Beschrijvingen

|  |  |  |
| --- | --- | --- |
| Code | UCB1: Spel spelen |  |
| **Naam** | Er kan een potje gespeeld worden |  |
| **Actor** | Gebruiker |  |
| **Triggers** | Gebruiker klikt op ‘Spel spelen’ |  |
| **Pre-Condities** | Gebruiker heeft de applicatie opgestart |  |
| **Post-Condities** | Gebruiker heeft het potje afgerond |  |
| **Stappen** | **Actor** | **Systeem** |
|  | 1. Gebruiker start applicatie op |  |
|  |  | 1. Systeem start applicatie op |
|  | 1. Gebruiker klikt op ‘Spel spelen’ |  |
|  |  | 1. Systeem laat Setup scherm zien |
|  | 1. Gebruiker nodigt vrienden uit om te spelen |  |
|  |  | 1. Systeem nodigt vrienden uit |
|  | 1. Gebruiker start het spel |  |
|  |  | 1. Systeem start een spel op |
|  |  |  |
|  | 1. Gebruiker kan geen vrienden uitnodigen |  |
|  |  | 1. Systeem start applicatie niet op |
|  |  |  |
| **Main success Scenario** | 1 – 2 – 3 – 4 – 5 – 6 – 8 |  |
| **Alternatieve Scenario** | 1 – 10 (Start niet op)  1 – 2 – 3 – 4 – 5 – 9 (Kan geen vrienden uitnodigen) |  |

**Toelichting: Setup scherm is het scherm waarop eventuele instellingen kunnen worden aangepast en spelers kunnen worden uitgenodigd.**

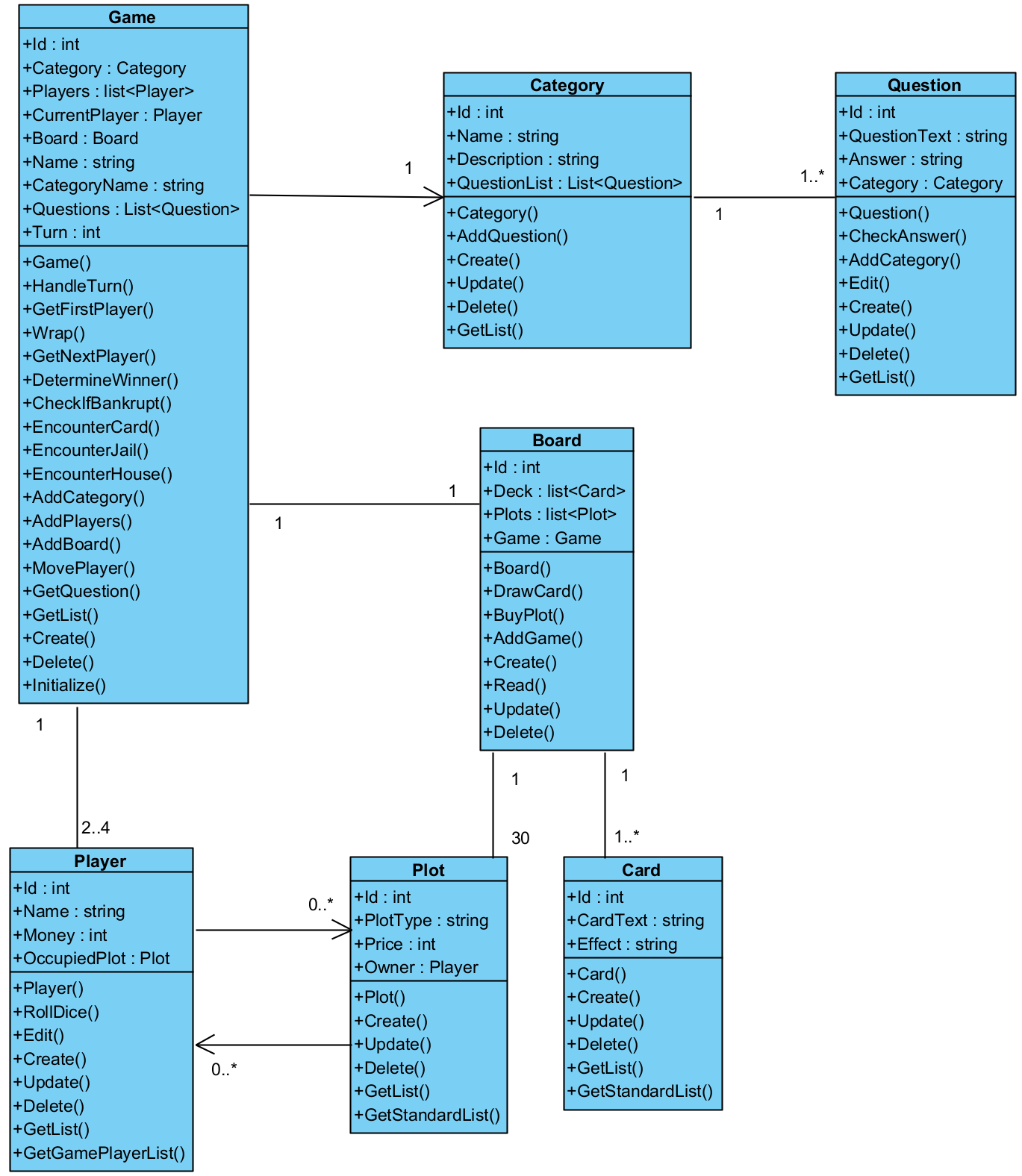
|  |  |  |
| --- | --- | --- |
| Code | UCB2: Categorieën maken |  |
| **Naam** | Gebruiker kan categorieën maken |  |
| **Actor** | Gebruiker |  |
| **Triggers** | Gebruiker klikt op categorie aanmaken |  |
| **Pre-Condities** | Applicatiebeheerder start de applicatie op |  |
| **Post-Condities** | Applicatiebeheerder heeft een categorie aangemaakt |  |
| **Stappen** | **Actor** | **Systeem** |
|  | 1. Gebruiker start app op |  |
|  |  | 1. Systeem laat app zien |
|  | 1. Gebruiker klikt op “Categorie maken” |  |
|  |  | 1. Systeem laat pagina zien om vragen te maken |
|  | 1. Gebruiker maakt een categorie aan |  |
|  | 1. Gebruiker slaat op |  |
|  |  | 1. Systeem slaat op |
|  |  |  |
|  | 1. Gebruiker maakt geen categorie aan |  |
|  |  | 1. Systeem slaat niet op |
| **Main success Scenario** | 1 – 2 – 3 – 4 – 5 – 6 – 7 |  |
| **Alternatieve Scenario** | 1 – 2 – 3 – 4 – 8 (geen categorie aangemaakt)  1 – 2 – 3 – 4 – 5 – 6 – 7 – 9 (categorie wordt niet opgeslagen) |  |

|  |  |  |
| --- | --- | --- |
| Code | UCB3: Vragen maken |  |
| **Naam** | Gebruiker kan vragen maken |  |
| **Actor** | Gebruiker |  |
| **Triggers** | Gebruiker klikt op vragen aanmaken |  |
| **Pre-Condities** | Gebruiker heeft de app opgestart |  |
| **Post-Condities** | Gebruiker heeft vragen kunnen aanmaken |  |
| **Stappen** | **Actor** | **Systeem** |
|  | 1. Gebruiker start app op |  |
|  |  | 1. Systeem laat app zien |
|  | 1. Gebruiker klikt op “Vraag maken” |  |
|  |  | 1. Systeem laat pagina zien om vragen te maken |
|  | 1. Gebruiker maakt een vraag aan |  |
|  | 1. Gebruiker slaat de vraag op |  |
|  |  | 1. Systeem slaat op |
|  |  |  |
|  | 1. Gebruiker maakt geen vraag aan |  |
|  |  | 1. Systeem slaat niet op |
| **Main success Scenario** | 1 – 2 – 3 – 4 – 5 – 6 – 7 |  |
| **Alternatieve Scenario** | 1 – 2 – 3 – 4 – 8 (geen vraag aangemaakt)  1 – 2 – 3 – 4 – 5 – 6 – 7 – 9 (vraag wordt niet opgeslagen) |  |

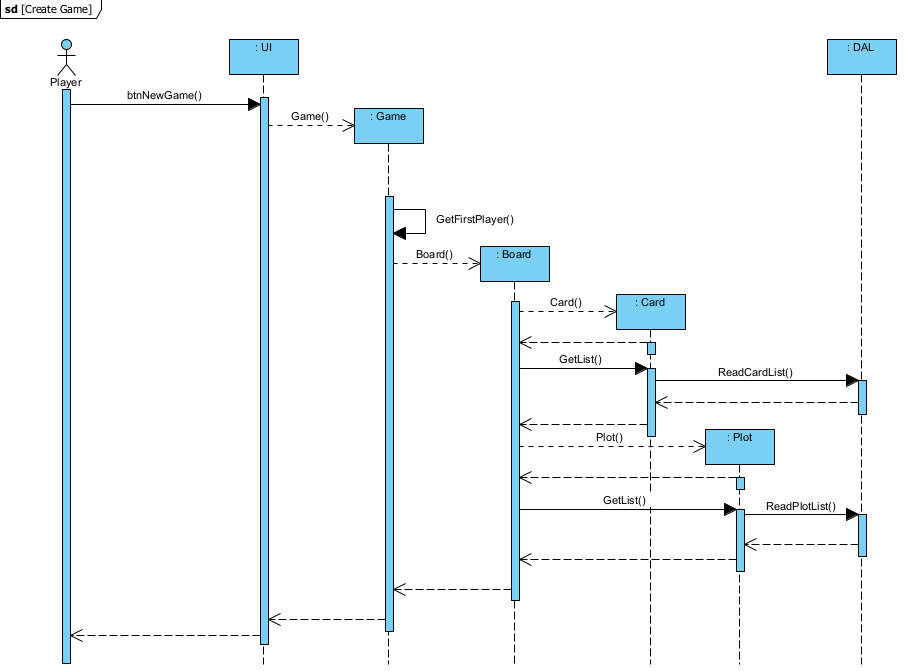
|  |  |  |
| --- | --- | --- |
| **Code** | UCB4: Spelers Beheren |  |
| **Naam** | Gebruiker spelers beheren. |  |
| **Actor** | Gebruiker |  |
| **Triggers** | Gebruiker klikt op “Players”. |  |
| **Pre-Condities** | Gebruiker heeft de app opgestart |  |
| **Post-Condities** | Gebruiker een game gemaakt. |  |
| **Stappen** | **Actor** | **Systeem** |
|  | 1. Gebruiker start de applicatie. |  |
|  |  | 1. Systeem start de applicatie. |
|  | 1. Gebruiker klikt op “Games Beheren”. |  |
|  |  | 1. Systeem laat het “Game Overview” scherm zien. |
|  | 1. Gebruiker drukt op “Players”. |  |
|  |  | 1. Systeem laat het “Players” scherm zien. |
|  | 1. Gebruiker voegt een speler toe. |  |
|  |  | 1. Systeem slaat de speler op in het spel en de database. |
|  | 1. Gebruiker verwijdert een speler. |  |
|  |  | 1. Systeem verwijderd de speler uit het spel en de database. |
|  | 1. Gebruiker past een speler naam aan. |  |
|  |  | 1. Systeem past de speler naam aan in de game en de database. |
| **Main success Scenario** | 1-2-3-4-5-6-7-8 |  |
| **Alternatieve Scenario** | 1-2-3-4-5-6-9-10 Gebruiker verwijderd een speler.  1-2-3-4-5-6-11-12 Gebruiker past een speler naam aan. |  |

|  |  |  |
| --- | --- | --- |
| **Code** | UCB5: Vragen Beheren |  |
| **Naam** | Gebruiker vragen beheren. |  |
| **Actor** | Gebruiker |  |
| **Triggers** | Gebruiker klikt op “Add”. |  |
| **Pre-Condities** | Gebruiker heeft de app opgestart |  |
| **Post-Condities** | Gebruiker een Category gemaakt. |  |
| **Stappen** | **Actor** | **Systeem** |
|  | 1. Gebruiker start de applicatie. |  |
|  |  | 1. Systeem start de applicatie. |
|  | 1. Gebruiker klikt op “Vragen Beheren”. |  |
|  |  | 1. Systeem laat het “Vragen Overview” scherm zien. |
|  | 1. Gebruiker drukt op “Add”. |  |
|  |  | 1. Systeem laat het “Questions” scherm zien. |
|  | 1. Gebruiker voegt een vraag toe. |  |
|  |  | 1. Systeem slaat de vraag op in de category en de database. |
|  | 1. Gebruiker verwijdert een vraag. |  |
|  |  | 1. Systeem verwijderd de vraag uit de category en de database. |
|  | 1. Gebruiker past een vraag aan. |  |
|  |  | 1. Systeem past de vraag aan in de category en de database. |
| **Main success Scenario** | 1-2-3-4-5-6-7-8 |  |
| **Alternatieve Scenario** | 1-2-3-4-5-6-9-10 Gebruiker verwijderd een vraag.  1-2-3-4-5-6-11-12 Gebruiker past een vraag aan. |  |

# Klassendiagram



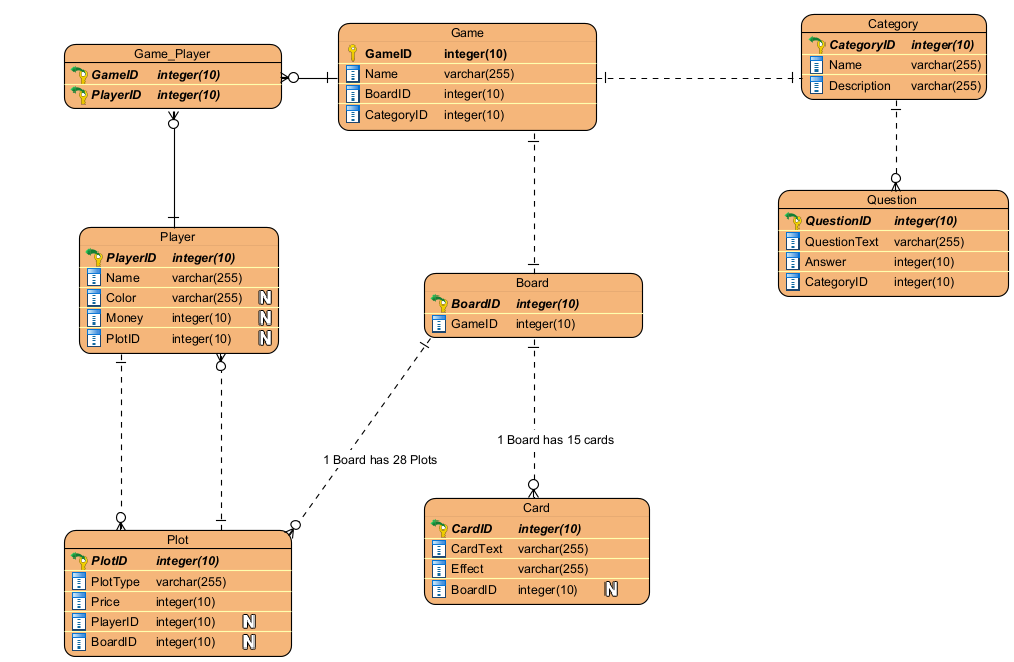
# Sequencediagram



In dit sequence diagram wordt er een Game object aangemaakt. In de constructor van Game, word er een Board aangemaakt. In de constructor van Board wordt er een Card aangemaakt, die de lijst van standaard kaarten van de SQL Server haalt en word er een Plot aangemaakt, die de lijst van standaard Plots van de SQL Server haalt.

Aan het einde van deze sequence is er een Game object gemaakt, de eerste speler gekozen, is er een Board gemaakt en zijn alle standaard Cards en Plots in board gestopt.

# ERD (Entiteit-Relationship Diagram)



# LGS (Logische GegevensStructuur)

**GAME** (gameid, categoryid, playerid, boardid)

* FK: categoryid > CATEGORY (categoryid)
* FK: playerid > PLAYER (playerid)
* FK: boardid > CATEGORY (categoryid)

**GAME\_PLAYER** (gameid, playerid)

* FK: gameid > GAME (gameid)
* FK: playerid > PLAYER (playerid)

**QUESTION** (questionid, questi0ntext, answer, categoryid)

* FK: categoryid > CATEGORY (categoryid)

**PLAYER** (playerid, name, color, money, plotid)

* FK: plotid > PLOT (plotid)

**BOARD** (boardid, gameid)

* FK: gameid > GAME (gameid)

**CATEGORY** (categoryid, name, description)

* FK: -

**PLOT** (plotid, plottype, price, playerid, boardid)

* FK: playerid > PLAYER (playerid)
* FK: boardid > BOARD (boardid)

**CARD** (cardid, cardtext, effect, boardid)

* FK: boardid > BOARD (boardid)

**PLOT** (plotid, plottype, playerid)

* FK: playerid > PLAYER (playerid)

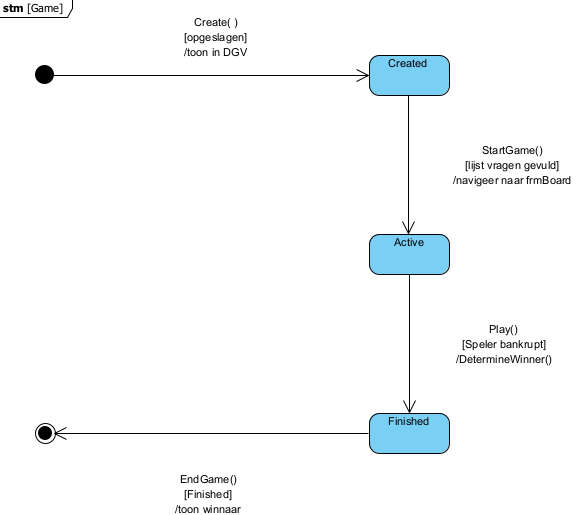
**CARD** (cardid, cardtext, effect)

* FK: -

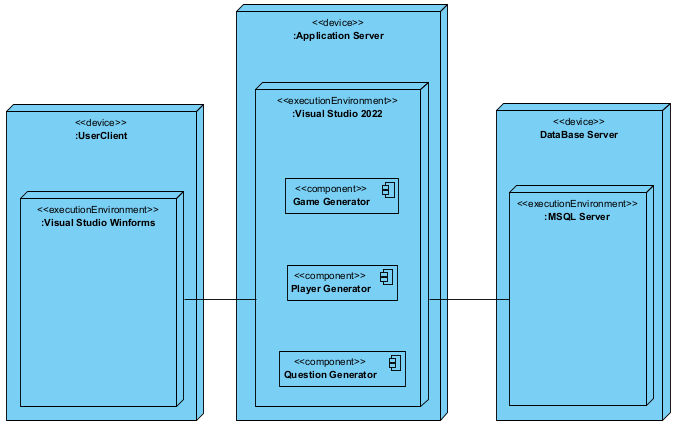
# Component Diagram



# State-Machine Diagram



# Deployement Diagram



# Wireframes

