Functioneel Ontwerp

Planning Poker

Versie 1.0

Timme Kingma s11  
2025

# Distributie

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Datum | Wijzigingen | Ontvangers |

# Inhoudsopgaven

[Distributie 2](#_Toc190619351)

[Inhoudsopgaven 3](#_Toc190619352)

[Inleiding 4](#_Toc190619353)

[1 Domeinanalyse 5](#_Toc190619354)

[1.1 Huidige situatie 5](#_Toc190619355)

[1.2 Gewenste situatie 5](#_Toc190619356)

[1.3 Entiteiten en relaties Domein model 6](#_Toc190619357)

[1.4 Use case diagram 6](#_Toc190619358)

[1.5 Entiteiten en relaties Use case diagram 7](#_Toc190619359)

[2 Users stories 8](#_Toc190619360)

[2.1 Als Scrum Master wil ik 8](#_Toc190619361)

[2.2 Als Developer wil ik 8](#_Toc190619362)

[3 Use cases 9](#_Toc190619363)

[3.1 Toelichting Betrokken requirements Asset/Type 9](#_Toc190619364)

[3.2 Toelichting Betrokken requirements MoSCoW 9](#_Toc190619365)

[3.3 Use Case: Bekijken Home pagina 10](#_Toc190619366)

[3.4 Use Case: Accountcreatie 12](#_Toc190619367)

[3.5 Use Case: Game starten 14](#_Toc190619368)

[3.6 Use Case: Game joinen 16](#_Toc190619369)

[3.7 Use Case: Game spelen 18](#_Toc190619370)

[3.8 Use Case: naam 20](#_Toc190619371)

[Bijlage 1 Aanpak Functioneel Ontwerp 21](#_Toc190619372)

[3.9 Domein 21](#_Toc190619373)

[3.10 Use case diagram 22](#_Toc190619374)

[3.11 Use cases 22](#_Toc190619375)

# Inleiding

Dit document beschrijft het functionele ontwerp van de webapplicatie Planning Poker. Het ontwerp is gebaseerd op een uitgebreide requirementsanalyse en richt zich op het ontwikkelen van een veilige, intuïtieve en gebruiksvriendelijke applicatie. Het doel van dit project is om teams effectief te ondersteunen bij het schatten van backlog items door middel van een interactieve en gestructureerde Planning Poker-game.

# Domeinanalyse

In dit hoofdstuk wordt het domein beschreven op basis van eigen ervaring. Het domein sluit aan bij de opgestelde requirements. Om het domein inzichtelijk te maken, is gebruikgemaakt van een domeinmodel en een use-case diagram. Dit hoofdstuk begint met de huidige situatie, gevolgd door een beschrijving van de gewenste situatie. Daarna wordt een domein-/use-case diagram gepresenteerd met toelichting.

## Huidige situatie

In de huidige webapplicaties kunnen developers al planning poker spelen, maar er ontbreken enkele functionaliteiten. De betere functies zitten achter een paywall, en daarnaast ontbreken enkele gewenste features voor student Timme Kingma. Zo is er geen manier om inzicht te krijgen in eerder gepokerde items, en ontbreekt een verbeterde methode voor het toekennen van punten. Zie IV1 in de requirementsanalyse voor meer informatie.

## Gewenste situatie

In de gewenste situatie kunnen developers, zoals een werkend agile-team, een planning poker-spel spelen waarbij ze verschillende backlogitems storypoints kunnen toekennen. Het proces moet soepel en flexibel zijn, zodat developers hun eigen werkwijze kunnen hanteren. Ze moeten de reeks storypoints kunnen aanpassen aan hun voorkeur, een timer in- of uitschakelen en het spel starten of eraan deelnemen via een veilig authenticatieproces. Dit kan bijvoorbeeld met e-mailverificatie of door in te loggen met een bestaand account, zodat alleen geautoriseerde gebruikers toegang hebben. Hierdoor ontstaat een efficiëntere en gebruiksvriendelijkere applicatie die aansluit bij de behoeften van het team. Hieronder staat het domeinmodel van de applicatie in de gewenste situatie.

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Figure Domein model Planing poker

## Entiteiten en relaties Domein model

Hieronder een tabel met alle entiteiten binnen de applicatie kort toegelicht. De naam van de bijbehorende klasse, die terugkomt in het domeinmodel, wordt ook vermeld.

|  |  |  |
| --- | --- | --- |
| Entiteit | Klassenaam | Toelichting |
| **Developer/**  **Scrummaster** | **User** | Een developer of scrummaster van de applicatie. |
| **2 Authentication** | **2 Authentication** | Een beveiliging code die gebruikt word om als gebruiker zich te identificeren. |
| **Match** | **Match** | De connectie tussen mensen en dieren. |
| **Planning poker spel** | ***Game*** | Het spel wat gespeeld gaat worden en bijbehorende settings. |

Tabel Tabel toelichting domein model

Tabel Entiteiten en relatie domein model

## Use case diagram

In dit hoofdstuk wordt het use-case diagram gepresenteerd. Het diagram geeft een overzicht van alle use cases en biedt inzicht in de verschillende actoren die betrokken zijn bij het systeem. Hieronder staat een figuur van het use-case diagram, gevolgd door tabellen met verdere toelichting.

Afbeelding met tekst, diagram, schermopname, lijn

Automatisch gegenereerde beschrijving

Figure 2 Use case diagram Planing poker

## Entiteiten en relaties Use case diagram

Hieronder twee tabellen met alle entiteiten binnen de applicatie kort toegelicht. De naam van de bijbehorende use case, die terugkomt in het use case diagram, wordt ook vermeld. De tabellen zijn opgesplitst in twee delen actoren en use cases.

|  |  |
| --- | --- |
| Actor | Toelichting |
| **Scrum master** | Een leider van een groep hij zorgt er voor dat een spel goed verloopt en ook ingesteld is volgens zijn wensen. |
| **Developer** | de developer speelt een spel onder leiding van de scrum master. |
| **Extern email systeem** | Een API systeem die de authenticatie code kan versturen naar developers. |

Tabel Toelichting actoren Use case diagram

|  |  |
| --- | --- |
| Use case | Toelichting |
| **Bekijken home pagina** | Een use case waarin de hoofdpagina bekeken gaat worden. |
| **Account maken** | Een use case waarin de developer een acount aan maakt. |
| **Game joinen** | Een use case waarin developers een game joined. |
| **Game starten** | Een use case waarin de game aangemaakt en gestart word. |
| **Game spelen** | Een use case waarin het spel gespeeld word. |

Tabel 4 Toelcihting use cases Use case diagram

# Users stories

In dit hoofdstuk worden de user stories beschreven. Hierdoor wordt een duidelijk overzicht gegeven van de functionaliteiten die de applicatie moet bieden.

## Als Scrum Master wil ik

* Een game kunnen starten om poker te spelen.
* Mensen kunnen uitnodigen om poker te spelen.
* Eén of meerdere backlogitems kunnen verzamelen en in de juiste volgorde tonen.
* De game kunnen leiden.

## Als Developer wil ik

* Een game kunnen joinen.
* Punten kunnen geven aan een backlogitem.
* Kunnen inloggen.
* Me kunnen registreren.
* Informatie over de applicatie kunnen lezen.
* Een andere naam dan mijn eigen naam kunnen invoeren.
* Automatisch ingelogd kunnen blijven zonder steeds mijn naam in te voeren.

# Use cases

Dit hoofdstuk beschrijft de use cases vanuit functioneel perspectief.

## Toelichting Betrokken requirements Asset/Type

|  |  |
| --- | --- |
| Asset/Type | Toelichting |
| **beperking** | **Een technische of wettelijke limiet die moet worden gerespecteerd bij de uitvoering van het project. Bijvoorbeeld regelgeving of systeemeisen.** |
| **Kwaliteit** | **Minimale standaarden waaraan het product moet voldoen om acceptabel te zijn voor gebruik. Denk aan performance, betrouwbaarheid en stabiliteit.** |
| **Proces** | **Activiteiten en methodologieën die nodig zijn om het project efficiënt uit te voeren, zoals documentatie en werkstromen.** |
| **Veiligheid** | **Bescherming tegen ongeoorloofde toegang, datalekken of andere beveiligingsrisico's.** |
| **Prestatie** | **Het vermogen van het systeem om binnen vastgestelde tijd en met de juiste snelheid te functioneren.** |
| **gebruikersgemak** | **De mate waarin het systeem eenvoudig en intuïtief te gebruiken is voor eindgebruikers.** |

## Toelichting Betrokken requirements MoSCoW

|  |  |
| --- | --- |
| definitie | Toelichting |
| **M (Must)** | **Verplichte onderdelen. Zonder deze vereisten kan het project niet als succesvol worden beschouwd.** |
| **S (Should)** | **Belangrijke onderdelen die niet essentieel zijn, maar een aanzienlijke toegevoegde waarde bieden als ze worden geïmplementeerd.** |
| **C (Could)** | **Wenselijke onderdelen die leuk zijn om te hebben, maar kunnen worden uitgesteld zonder grote impact.** |
| **W (Won't)** | **Zaken die voorlopig niet worden geïmplementeerd, mogelijk wel in toekomstige iteraties.** |

## Use Case: Bekijken Home pagina

|  |  |
| --- | --- |
| Use case naam | Bekijken home pagina |
| Actoren | Scrum master, Developer |
| Doel | Een developer kan de hoofdpagina bekijken en zich oriënteren over de het planning poker. |
| Precondities | Leeg |
| Hoofdscenario | 1. De developer bezoekt de Home |
| Alternatieve scenario’s | 1. De website is niet geschikbaar |
| Postcondities | De website is inzichtbaar. |

### Betrokken requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Beschrijving | Asset/Type | MoSCoW | Testen |
| NFR1 | De gegevens op de pagina zijn niet via de webpagina te beheren | Beperking | Must |  |
| NFR2 | De pagina wordt binnen 1 seconde geladen | Kwaliteit | Must |  |

### Schermontwerp

Afbeelding met tekst, schermopname, Lettertype, ontwerp

Automatisch gegenereerde beschrijving

Figure Shermontwerp hoofdpagina

### Evil user stories

|  |  |  |
| --- | --- | --- |
| # | Evil user story | Mitigatie |
| EUS1 | Als aanvaller wil ik toegang krijgen tot de cookievoorkeuren van gebruikers, zodat ik de sessie kan overnemen. | SM1 |

### Activity diagram

Hieronder een activity diagram om de functionele flow duidelijk te maken.

## Use Case: Accountcreatie

|  |  |
| --- | --- |
| Use case naam | Accountcreatie |
| Actoren | Developer |
| Doel | Een nieuw account te maken en in te loggen |
| Precondities | leeg |
| Hoofdscenario | 1. De developer maakt een account aan  2. De developer verifieerd zijn account via email authenticatie.  3. De developer logt in met zijn nieuwe account  4 De sessie word opgeslagen |
| Alternatieve scenario’s | 2.1 Het acount word niet geauthentificeerd.  3.1 De developer heeft zijn acount vergeten of niet geverifieerd  De developer kan bij het inloggen alsnog authenticeren. |
| Postcondities | Er is een sessie/cookie opgeslagen waarin de developer niet weer hoeft inte loggen pas na 15 dagen.  De gebruiker heeft een acount  De gebruiker kan inloggen en word developer. |

### Betrokken requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Beschrijving | Asset/Type | MoSCoW | Testen |
| NFR3 | Developer kan een acount aanmaken | Functionaliteit | Must have |  |
| NFR4 | Verificatie is vereist | Beveiliging | Must have |  |
| NFR5 | Sessie/cookie word opgeslagen voor 15 dagen | Beveiliging | Should have |  |
| NFR6 | Developer kan inloggen met nieuw account. | Functionaliteit | Must have |  |

### Schermontwerp

Afbeelding met tekst, schermopname, software, Multimediasoftware

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermopname, software, Multimediasoftware

Automatisch gegenereerde beschrijving

Figure Registratie Schermontwerp Figure Login Schermontwerp

### Evil user stories

|  |  |  |
| --- | --- | --- |
| # | Evil user story | Mitigatie |
| EUS2 | Een aanvaller probeert een account aan te maken met een bestaand e-mailadres | Unieke e-mailvalidatie en verificatie |
| EUS3 | Een aanvaller probeert brute-force logins | Rate limiting en CAPTCHA bij mislukte loginpogingen |
| EUS4 | Een aanvaller probeert de sessie te kapen | Secure cookies, HTTPS, en token-based authenticatie |
| EUS5 | Een aanvaller probeert met bots de servers te overflowen door meerdere malen te registreren | Rate limiting en CAPTCHA |

### Activity diagram

Hieronder een activity diagram om de functionele flow duidelijk te maken.

## Use Case: Game starten

|  |  |
| --- | --- |
| Use case naam | Game starten |
| Actoren | Scrum Master |
| Doel | Een game starten om planning poker te spelen |
| Precondities | Scrum master is ingelogd |
| Hoofdscenario | 1. Scrum Master navigeert naar de game-startpagina. 2. Scrum Master selecteert de optie om een nieuwe game te starten. 3. Scrum Master vult de game-instellingen in (zoals naam van de game, timerinstellingen, en type stemmen). 4. Scrum Master bevestigt de instellingen en start de game. 5. De game wordt gecreëerd en zichtbaar voor uitgenodigde deelnemers. 6. Na het joinen van deelnemers start de Scrum master de game. |
| Alternatieve scenario’s | 2.1 Scrum Master besluit halverwege het instellen van de game om te annuleren.  2.2 Het systeem sluit het instellingsscherm zonder een game te starten. |
| Postcondities | * De game is succesvol gestart en klaar voor deelname. * Spelers kunnen deelnemen op basis van uitnodigingen of een gamecode. * Game is gestart en kan gespeeld worden. |

### Betrokken requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Beschrijving | Asset/Type | MoSCoW | Testen |
| NFR7 | De Scrum Master moet een game kunnen starten | Functionaliteit | Must have | - |
| NFR8 | De game-instellingen moeten configureerbaar zijn | Functionaliteit | Should have | - |
| NFR9 | De game moet een unieke ID genereren | Systeem | Must have | - |

### Schermontwerp

Afbeelding met tekst, schermopname, software, Website

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermopname, Rechthoek, ontwerp

Automatisch gegenereerde beschrijving

Figure Game starten schermontwerp Figure Game join schermontwerp

### Evil user stories

|  |  |  |
| --- | --- | --- |
| # | Evil user story | Mitigatie |
| EUS6 | Een niet-ingelogde gebruiker probeert een game te starten | Alleen geauthenticeerde gebruikers mogen een game starten |
| EUS7 | Een kwaadwillende gebruiker start meerdere games om het systeem te overbelasten | Beperk het aantal gelijktijdige games per gebruiker |
| EUS8 | Een gebruiker vult ongeldige of schadelijke gegevens in bij het aanmaken van de game | Implementeer inputvalidatie en sanitatie |

### Activity diagram

Hieronder een activity diagram om de functionele flow duidelijk te maken.

## Use Case: Game joinen

|  |  |
| --- | --- |
| Use case naam | Game joinen |
| Actoren | Developer |
| Doel | Een planning poker game te kunnen joinen |
| Precondities | Er is een game aangemaakt en de developer heeft een code gekregen om zichzelf te kunnen authentiseren en ook een link om te kunnen joinen. |
| Hoofdscenario | 1. De developer gebruikt de link om op de juiste pagina te komen  2. De developer voert de een eigen naam in of logt in  3. De developer verifieert zichzelf met de bijgegeven code  4. de developer joined de game |
| Alternatieve scenario’s | 2. Er komt een error met het inloggen van de developer  De developer moet opnieuw proberen  4 de game bestaat niet meer  Er moet een nieuwe game gemaakt worden |
| Postcondities | De developer joind een game |

### Betrokken requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Beschrijving | Asset/Type | MoSCoW | Testen |
| NFR10 | De developer kan via een link op de juiste game pagina komen | Functionaliteit | Must have |  |
| NFR11 | De developer kan een eigen naam invoeren of inloggen | Functionaliteit | Must have |  |
| NFR12 | De developer moet een geldige code invoeren om zichzelf te verifiëren | Beveiliging | Must have |  |
| NFR13 | De developer kan succesvol een game joinen | Functionaliteit | Must have |  |
| NFR14 | Er word een foutmelding weergegeven als de game niet meer bestaat | UX | Should have |  |

### Schermontwerp

Afbeelding met tekst, schermopname, Rechthoek, ontwerp

Automatisch gegenereerde beschrijving

Figure Game join schermontwerp

### Evil user stories

|  |  |  |
| --- | --- | --- |
| # | Evil user story | Mitigatie |
| UES9 | Een aanvaller probeert zonder geldige code een game te joinen | Unieke en eenmalig te gebruiken verificatiecode |
| UES10 | Een kwaadwillende gebruiker probeert meerdere keren een foute code in te voeren | Rate limiting en account blokkering na X pogingen |
| UES11 | Een kwaadwillende gebruiker probeert het systeem te overbelasten door herhaaldelijk te joinen met scripts | Rate limiting en CAPTCHA bij overmatig gebruik |

### Activity diagram

Hieronder een activity diagram om de functionele flow duidelijk te maken.

## Use Case: Game spelen

|  |  |
| --- | --- |
| Use case naam | Game spelen |
| Actoren | Developer,Scrum master |
| Doel | Als developers moet er een game gespeeld kunnen worden en backlog item/s storypoints gegeven kunnen worden. |
| Precondities | Er bestaat een game met spelers er in. |
| Hoofdscenario | 1. De scrum master begint een game en voert of pakt uit een bestaande lijst een backlog item.  2. Het geselecteerde backlog item word weergegeven.  3. spelers en de scrum master kunnen storypoints opgeven via de vooraf ingestelde hoeveelheden.  4. De applicatie OF de scrummaster beëindigt de ronde.  5. gaat terug naar stap 1 |
| Alternatieve scenario’s | 2.1. Er kunnen syncing errors ontstaan.  Start een nieuwe game |
| Postcondities | De spelers kunnen input geven.  De scrum master kan input voor een taak/en opgeven. |

### Betrokken requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Beschrijving | Asset/Type | MoSCoW | Testen |
| NFR15 | De scrum master kan een game starten en een backlog item selecteren | Functionaliteit | Must have |  |
| NFR16 | Het geselecteerde backlog item wordt correct weergegeven met realtime comunicatie | Functionaliteit | Must have |  |
| NFR17 | Spelers en de scrum master kunnen storypoints toekennen | Functionaliteit | Must have |  |
| NFR18 | De applicatie of de scrum master kan een ronde beëindigen | Functionaliteit | Must have |  |
| NFR19 | Er moet foutafhandeling zijn bij synchronisatiefouten | Beveiliging | Should have |  |
| NFR20 | De scrum master moet de mogelijkheid hebben om de game opnieuw te starten bij fouten | Functionaliteit | Could have |  |

### Schermontwerp

Afbeelding met tekst, schermopname, ontwerp

Automatisch gegenereerde beschrijving

Figure Game spelen schermontwerp

### Evil user stories

|  |  |  |
| --- | --- | --- |
| # | Evil user story | Mitigatie |
| UES12 | Een aanvaller probeert ongeautoriseerd een game te starten | Alleen geauthenticeerde scrum masters mogen een game starten |
| UES13 | Een kwaadwillende gebruiker probeert ongeldige storypoints in te voeren | Alleen vooraf ingestelde storypoint-opties toestaan |
| UES14 | Een speler probeert het stemmen van anderen te manipuleren | Storypoints pas zichtbaar maken na inzending van alle stemmen |
| UES15 | Een aanvaller probeert een game vast te laten lopen door overmatige requests te sturen | Rate limiting en server-side validatie |

### Activity diagram

Hieronder een activity diagram om de functionele flow duidelijk te maken.

## Use Case: naam

|  |  |
| --- | --- |
| Use case naam |  |
| Actoren |  |
| Doel |  |
| Precondities |  |
| Hoofdscenario |  |
| Alternatieve scenario’s |  |
| Postcondities |  |

### Betrokken requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Beschrijving | Asset/Type | MoSCoW | Testen |

### Schermontwerp

### Evil user stories

|  |  |  |
| --- | --- | --- |
| # | Evil user story | Mitigatie |

### Activity diagram

Hieronder een activity diagram om de functionele flow duidelijk te maken.

# Bijlage 1 Aanpak Functioneel Ontwerp

In deze bijlage een overzicht van de stappen die genomen zijn om te komen tot een functioneel ontwerp.

Eerst een schematische weergave van de stappen:

Afbeelding met tekst, schermopname, lijn, Lettertype

Automatisch gegenereerde beschrijving

Figuur Ontwikkelstappen Functioneel Ontwerp

In het SSDLC ziet dit er als volgt uit:

Afbeelding met tekst, schermopname, cirkel, diagram

Automatisch gegenereerde beschrijving

Figuur Threat Modeling in het Functioneel Ontwerp

## Domein

Het domein bepalen zorgt voor een gemeenschappelijk referentiekader tussen de stakeholders en het development team.  
Dit referentiekader is een vertrekpunt voor de vervolgstappen. Tijdens het ontwerpen kan het noodzakelijk zijn om het domein aan te passen of uit te breiden. Overleg met stakeholders moet daarvoor plaatsvinden.  
In use cases mogen geen nieuwe domein entiteiten of eigenschappen worden gebruikt, alleen wat in het domein model is opgenomen mag gebruikt worden.

## Use case diagram

Om een functioneel overzicht te bieden van de applicatie is een use case diagram opgenomen. Het zoeken naar de relatie tussen Domein en Use case diagram is een belangrijke stap voor stakeholders en developers.

## Use cases

De use cases moeten in lijn zijn met:

* Requirements analyse
* Domein
* Use case diagram
* Use Case beschrijving (optioneel te gebruiken, wel erg bruikbaar!)

### Principes

Een requirement kan gelden voor meer dan één use case. Bijvoorbeeld: wanneer een request wordt verstuurd, dan toont de pagina een spinner. Dit is een algemeen acceptatiecriterium dat als een algemeen principe kan worden opgenomen in de requirementsanalyse. Hieronder een voorbeeld:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Bron | Beschrijving | Asset/Type | MoSCoW | Testen |
| P1 | IV# | Wanneer een request wordt verstuurd, dan toont de pagina een spinner. | Principe |  |  |
| UC1 | IV1 | Als gebruiker wil ik de CV van een developer kunnen zien om te oriënteren voordat ik contact leg |  | Must | Functioneel |
| … |  | … | .. | .. |  |

Als identificatie is P1 gebruikt. Maar een onderverdeling had ook gekund: PF1 als identificatie voor **p**rincipes voor **f**ormulieren en gegevens versturen.

Een alternatief is om principes op te nemen in een hoofdstuk/bijlage van het Functioneel Ontwerp.

### Acceptatiecriteria

Bij de use case zijn acceptatiecriteria opgenomen, deze zijn overgenomen van de requirementanalyse. Zoals te zien is aan de markeringen zijn nieuwe requirements nodig en aanscherping van bestaande requirements.  
Vervolgens zijn Evil user stories opgesteld en gekeken of de Security Measurements (SM#) vastgelegd in de requirementsanalyse afdoende waren. De conclusie was: SM1 moet worden aangescherpt!

### Schermontwerpen

Om een goed beeld te krijgen van de functionele werking van een webpagina kun je niet zonder een schermontwerp. Aangeraden wordt om Figma te gebruiken. Met deze tool kun je wireframes en componenten maken (bijvoorbeeld een status aan een knop meegeven: hover, disabled).

### Controle requirements

Na het uitwerken van het Functioneel Ontwerp is een controle uitgevoerd of het ontwerp overeenkomstig de eisen de in de requirementsanalyse is opgesteld.