Analysedocument

Live performance

Roy v.Oldenbeek

2016

Inhoud

[Inleiding 2](#_Toc454356104)

[Scenario 3](#_Toc454356105)

[User requirements 4](#_Toc454356106)

[Must haves: 4](#_Toc454356107)

[Should haves: 4](#_Toc454356108)

[Could haves: 5](#_Toc454356109)

[Wont haves: 5](#_Toc454356110)

[ER-Diagram 6](#_Toc454356111)

[UseCase diagram 7](#_Toc454356112)

[UseCases 8](#_Toc454356113)

[Planning 9](#_Toc454356114)

# Inleiding

In dit document zal aan de hand van de uitwerking van een scenario een analyse worden uitgevoerd wat er nodig zal zijn om te bootverhuurapplicatie van botenverhuurbedrijf ’t Sloepke te realiseren. Aan de hand van het scenario zullen de User requirements opgesteld worden door middel van het MoSCoW principe. Verder zal er een ERD in zitten alsmede een use case diagram en uitgewerkte use cases. Als laatste zit er een planning bij. ’t Sloepke wilt een applicatie hebben om eenvoudig zijn boten en bijbehorende materialen te kunnen verhuren. Deze applicatie moet met een opgegeven budget en type huurboot automatisch uitrekenen hoeveel meren er bevaren mogen worden. Tevens moeten gegevens over de huurder en het huurcontract opgeslagen worden in een database, en moet het huurcontract geëxporteerd kunnen worden naar een leesbaar tekstbestand.

# Scenario

Jan en Anja willen voor een weekendje weg gezellig gaan varen met een zeilboot op de Friese meren. Ze zoeken online naar een geschikt bootverhuur bedrijf en komen uit bij bootverhuur ’t Sloepke. Ze noteren het telefoonnummer van dit bedrijf en bellen vervolgens niet veel later om een zeilboot te huren.

Als ze bellen vraagt de verhuurder in eerste instantie wat voor een boot er gehuurd wilt worden door Jan en Anja; hij verteld hun dat er twee categorieën zijn, spierkracht aangedreven boten en motorboten, van motorboten verhuren ze op dit moment één soort boot, van het type Kruiser. Van spierkracht aangedreven boten hebben ze drie verschillende boten, een kano, en twee type zeilboten van het type Laser en Valk. Door de verhuurder wordt het verschil uitgelegd tussen de verschillende type boten, en met welke boten je waar mag varen. Uit eindelijk kiezen Jan en Anja ervoor om een zeilboot van het type Laser te huren. Ze geven dit door en de verhuurder vraagt hoeveel dagen de boot gehuurd moet worden, dit zijn er 3, van vrijdag t/m zondag. Daarna vraagt hij hoeveel zwemvesten, peddels, nog wat kampeerartikelen en dekzeilen er bij moeten. In totaal komt het op 5 extra bijbehorende artikelen. Als laatste vraagt de verhuurder wat hun totale budget is, zodat hij kan berekenen op hoeveel Friese meren ze mogen varen. Ze geven aan dat hun totale budget 52 euro is voor de boot huur. De verhuurder vult dit allemaal in op zijn applicatie en drukt op de bereken knop. Er komt uit dat in totaal 3 meren nog bevaren mogen worden met dit budget. Hij vraagt of dit goed is. Jan en Anja gaan akkoord, en zeggen dat ze deze boot graag willen huren.

Voor het huren vraagt de verhuurder naar de volledige naam en het e-mail adres van Jan, want hij is diegene die belt. Dit noteert hij ook in de applicatie en drukt vervolgens op de knop Huurcontract genereren. De applicatie genereerd een huurcontract en slaat deze op op de harde schijf van de verhuurder, tevens worden alle gegevens over deze huur vastgelegd in een database. De verhuurder zal Jan en Anja het contract ook per e-mail toesturen en Jan en Anja moeten als ze de boot op komen halen betalen.

Nadat er opgehangen is kijkt de verhuurder nog een keer in zijn applicatie, en vraagt daar de lijst op met huurcontracten die nog uit moeten worden gevoerd. Hij kijkt in deze lijst welke boten er morgen weg moeten en begint met het voorbereiden van deze boten zodat de klanten morgen de boten in een goede staat gelijk mee kunnen nemen.

# User requirements

Hieronder staan alle requirements voor de applicatie, gegroepeerd volgens de MoSCoW methode. Aan elke requirement zal een code worden gehangen, zodat er later makkelijk naar deze requirement kan worden verwezen.

## Must haves:

|  |  |
| --- | --- |
| Code | Omschrijving user requirement |
| M1 | Van een huurder moeten naam en e-mail adres opgeslagen worden. |
| m2 | Elke boot moet een unieke naam hebben. |
| m3 | Als een boot een motorboot is dan moet de tankinhoud en de actieradius van de boot opgevraagd kunnen worden. De actieradius is vastgesteld op de literinhoud vermenigvuldigd met 15. |
| m4 | Er moeten bijkomende artikelen gehuurd kunnen worden. |
| m5 | Alle gegevens in een huurcontract moeten in de database op worden geslagen. |
| m6 | Er moeten nieuwe huurcontracten aan kunnen worden gemaakt. |
| m7 | Een lijst met alle huurcontracten kunnen worden weergegeven op het scherm. |
| m8 | In het systeem moet het type boot, het aantal bijkomende artikelen, het budget en of een klant ook op de Noordzee of het IJsselmeer wilt varen ingevoerd worden waarna de applicatie terug geeft op hoe veel Friese meren er gevaren mag worden. |
| m9 | Het gemaakte huurcontract moet kunnen worden geëxporteerd naar een leesbaar tekst (txt) bestand. |
| m10 | Er moeten verschillende type boten gehuurd kunnen worden. |
| m11 | Er moeten twee categorieën zijn waar deze boten onder kunnen vallen, motorboot en spierkracht aangedreven boot. |
| M12 | De applicatie moet eenvoudig uitgebreid kunnen worden met extra vaarwateren. |
| M13 | De applicatie moet door een programmeur eenvoudig uitgebreid kunnen worden met extra type boten. |
| M14 | Met een spierkracht aangedreven boot mag alleen maar op de Friese meren gevaren worden. |

## Should haves:

|  |  |
| --- | --- |
| Code | omschrijving user requirement |
| s1 | Een klant kan meerdere boten gelijk huren. |
| s2 | De gevoelstemperatuur moet weergegeven worden door de applicatie voor op de data waarop de boot gehuurd kan worden. |
| S3 | Een klant kan een “overnachtingspakket” aan bijbehorende artikelen selecteren, waardoor in een keer de juiste extra spullen worden toegevoegd voor een of meerdere overnachtingen. |

## Could haves:

|  |  |
| --- | --- |
| Code | omschrijving user requirement |
| C1 | Er moet een navigatie mini game gespeeld kunnen worden |
| c2 | In deze mini game moet men met de cursortoetsen een bootje door een vaart kunnen navigeren zonder de wal te raken |
| c3 | Deze minigame is alleen maar speelbaar als iemand een zeilboot huurt |
| c4 | De wind komt uit het westen in de minigame, en deze kant moet dus niet op kunnen worden gevaren |
| c5 | Er moeten verschillende hindernissen(boeien, dukdalven, meerpalen en overige obstakels) ontweken worden; worden deze objecten geraakt dan moet er een bijpassend geluid afgespeeld worden |
| C6 | Het huurcontract moet ook als een html bestand kunnen worden geëxporteerd. |
| C7 | Er moet gekozen kunnen worden tussen een HTML export en de tekst file export. |
| C8 | Een admin gebruiker kan inloggen en kan bijbehorende schepen toevoegen, aanpassen en verwijderen |
| C9 | Een admin gebruiker kan bijbehorende materialen toevoegen, aanpassen en verwijderen. |

## Wont haves:

Alles wat niet beschreven is als een Must, Should of Could have zal niet worden geïmplementeerd.

# ER-Diagram

Dit is het entiteit relatie diagram voor de applicatie, hierin wordt de algemene structuur en relaties duidelijk die in de database moeten worden aangemaakt.



Een huurcontract heeft altijd een klant, maar het kan zo zijn dat een klant nog geen huurcontracten heeft, daarom dat dit een nul op veel relatie is aangezien een klant ook vaker terug kan komen om een boot te huren. In een huurcontract zit een boot, maar een boot kan meerdere keren voorkomen op verschillende huurcontracten.

Verder kunnen er meerdere artikelen op een huurcontract zitten, en kan een artikel ook een relatie met een ander artikeltype als het artikel bij een pakket hoort. In de tabel Vaarwater worden de verschillende vaarwateren opgeslagen buiten de Friese meren om, aangezien elke boot op een Fries meer mag varen, daarom dat het een nul op meer relatie is want zeilboten hebben hier niks in staan. Verder wordt er voor de boten nog een type onderscheiding gemaakt tussen motorboot en spierkrachtboot.

# UseCase diagram

Hier staat het UseCase diagram, in dit diagram wordt duidelijk wat de gebruiker van het systeem moet kunnen doen, wat een administrator moet kunnen doen en wat de klant moet kunnen doen met het systeem. Een klant heeft heel weinig interactie met het systeem aangezien het een beheersysteem is voor ’t Slopke en er dus achter de schermen van de klant mee gewerkt wordt, alleen als deze een minigame moet spelen heeft de klant interactie met het systeem.



# UseCases

|  |  |
| --- | --- |
| Naam | Boot verhuren |
| Samenvatting | Een actor moet een boot kunnen verhuren. |
| Actor | Gebruiker, Admin |
| Aanname | Alle gegevens die ingevuld moeten worden zijn bekend |
| Beschrijving | 1. De actor vult het budget, het type boot, de te huren materialen en de begin en einddatum in. 2. De actor selecteert de vaarwateren waarin de boot gaat varen[1]. 3. De actor klikt op bereken, de applicatie berekent vervolgens uit hoeveel meren er bevaren kunnen worden 4. De actor vult de naam en het e-mail adres in van de huurder. 5. De actor klikt op Huren |
| Uitzondering | 1. Als het een zeilboot is dan kan er niks worden aangegeven qua vaarwateren, deze lijst is dan ook leeg en de applicatie gaat gewoon door. |
| Resultaat | De huur wordt opgeslagen in de database en de actor krijgt een message box te zien met dat de huur geslaagd is. |

|  |  |
| --- | --- |
| Naam | Lijst van verhuurde boten opvragen |
| Samenvatting | Een actor moet kunnen zien welke verhuren er aan staan te komen en moet hiervan een lijst kunnen raadplegen |
| Actor | Gebruiker, Admin |
| Aanname | - |
| Beschrijving | 1. Gebruiker klikt op “Toon huurcontracten” 2. De applicatie haalt de huurcontracten op uit de database en laat deze zien in een listbox[1]. |
| Uitzondering | 1. Als er geen huurcontracten in de database staan dan is de listbox leeg. |
| Resultaat | De gebruiker krijgt een overzicht van de boten die nog verhuurd moeten worden |

|  |  |
| --- | --- |
| Naam | Vaarlocatie toevoegen |
| Samenvatting | De actor moet een vaarlocatie toe kunnen voegen |
| Actor | Gebruiker, Admin |
| Aanname | - |
| Beschrijving | 1. De actor klikt op “Vaarlocatie toevoegen” in de applicatie. 2. Een nieuw scherm komt naar voren waarin gevraagd wordt naar de naam van de nieuwe vaarlocatie. 3. De gebruiker drukt op OK[1]. |
| Uitzondering | 1. Als de actor een ongeldige naam heeft opgegeven wordt een error weergeven in een messagebox en moet de actor deze opnieuw invoeren. |
| Resultaat | Er is een nieuwe vaarlocatie opgegeven in de database en deze kan gebruikt worden in nieuwe verhuren. |

|  |  |
| --- | --- |
| Naam | Huurcontract exporteren |
| Samenvatting | Een actor moet een huurcontract naar een tekstbestand kunnen exporteren. |
| Actor | Gebruiker, Admin |
| Aanname | Er is een geldig huurcontract aangemaakt |
| Beschrijving | 1. Actor drukt op “Exporteer huurcontract”. 2. Actor krijg een scherm te zien waarin hij kan kiezen waar het bestand opgeslagen wordt. 3. Actor drukt op OK. |
| Uitzondering | - |
| Resultaat | Een bestand wordt opgeslagen op de harde schijf waarin de gegevens over het huurcontract staan. |

|  |  |
| --- | --- |
| Naam | Inloggen |
| Samenvatting | Een administrator moet kunnen inloggen |
| Actor |  |
| Aanname |  |
| Beschrijving |  |
| Uitzondering |  |
| Resultaat |  |

|  |  |
| --- | --- |
| Naam |  |
| Samenvatting |  |
| Actor |  |
| Aanname |  |
| Beschrijving |  |
| Uitzondering |  |
| Resultaat |  |

|  |  |
| --- | --- |
| Naam | Boot verhuren |
| Samenvatting |  |
| Actor |  |
| Aanname |  |
| Beschrijving |  |
| Uitzondering |  |
| Resultaat |  |

# Planning

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Taak | Begin geplant | Eind geplant | Begin reëel | Eind reëel |
| Analysedocument | 09:00 | 09:15 | 09:00 | 09:10 |
| User requirements | 09:15 | 09:45 | 09:10 | 09:35 |
| Scenario | 09:45 | 10:15 | 09:35 | 09:56 |
| ERD | 10:30 | 11:00 | 10:00 | 10:43 |
| Use-case diagram | 11:00 | 12:00 | 10:45 | 10:56 |
| Use cases | 12:45 | 13:30 | 10:56 |  |
| UI schetsen | 13:30 | 14:00 |  |  |
| Ontwerpdocument | 14:00 | 14:15 |  |  |
| Datamodel | 14:30 | 15:15 |  |  |
| Klassendiagram | 15:15 | 15:45 |  |  |
| Testplan | 15:45 | 16:30 |  |  |
| SQL scripts | 16:30 | 09:30 |  |  |
| DAL | 09:30 | 10:15 |  |  |
| Klassendiagram implementeren | 10:30 | 11:45 |  |  |
| Overige functionaliteiten programmeren en gelijktijdig unit tests aanmaken | 11:45 | 15:30 |  |  |
| Documentatie in orde maken | 15:30 | 17:00 |  |  |