Afstudeerverslag

DE HAAGSE HOGESCHOOL

Gegevens

Naam: Shawn Steinz
Studentnummer: 10015493
Bedrijf: NewCompliance
Studie: Informatica
School: De Haagse Hogeschool
Stagebegeleider: A.A.A.M Jacobs
Bedrijfsbegeleider: H. van den Dries
Periode: 4 mei 2015- 5 oktober 2015
Plaats: Zoetermeer

Referaat

Afstudeerverslag, Shawn Steinz, NewCompliance te Zoetermeer, Haagse Hogeschool, 2 oktober 2015.

Dit afstudeerverslag beschrijft het proces rondom de afstudeeropdracht, die gedurende de periode van mei 2015 tot en met oktober 2015 is uitgevoerd. De opdracht is uitgevoerd voor NewCompliance. Deze opdracht is gerelateerd aan de opleiding Informatica van de Haagse Hogeschool.

De afstudeeropdracht betreft het maken van een Medische Cockpit voor ziekenhuizen. De cockpit is ontworpen volgens de opgestelde requirements. De cockpit is daarna gebouwd en getest. De cockpit zorgt ervoor dat de verkregen proces data inzichtelijk getoond kan worden aan de ziekenhuizen tijdens hun processen. Met behulp van deze proces data kunnen de processen in het ziekenhuis gestuurd worden. Dit zal er voor zorgen dat de zorg veiliger wordt.

Descriptoren:
- Ziekenhuizen
- Mediche proces
- PHP
- Mappings
- MoSCoW
- Phpstorm
- Laravel

Voorwoord

Het afstudeerverslag is geschreven in het kader van mijn studie Informatica aan de Haagse Hogeschool. De opdracht is uitgevoerd voor NewCompliance. Tijdens de opdracht wordt er een Medische Cockpit ontwikkeld. Deze Cockpit zal belangrijke proces informatie gaan tonen in ziekenhuizen.

Dit proces en de kennis die ik heb opgedaan wil ik door middel van dit verslag met u delen. Verder wil ik in dit voorwoord graag een aantal mensen bedanken. Allereerste bedanken ik Henk van den Dries voor het intern begeleiden van de opdracht. Daarnaast wil ik Bo Wiesman bedanken voor het beschikbaar stellen van deze opdracht. Verder wil ik graag Arno Nederend en Nanny Jacobs bedanken voor het begeleiden van de opdracht en het leveren van bruikbare kritiek gedurende de opdracht.

Shawn Steinz
Pijnacker, 1-10-2015

Inhoudsopgave

Inleiding 1

1. NewCompliance 2

1.1 Bedrijf 2

1.2 Missie NewCompliance 3

1.3 Interne organisatie 4

2. Opdracht 5

2.1 Initiële opdracht 5

2.2 Probleemstelling 5

2.3 Doelstelling 6

2.4 Gewenste resultaat 6

3. Aanpak en Methode 7

3.1 OK Cockpit 7

3.2 Methodiek 9

3.2.1 Keuze 9

3.2 Gesprekken 11

4 Werkzaamheden 12

4.1 Sprint 0 project opzet 12

4.1.1 Aangetroffen staat 12

4.1.2 Planning 13

4.1.3 Requirements 14

4.1.4 Laravel 19

4.1.5 ERD (Entity–relationshipmodel) diagram huidige situatie 21

4.1.7 Evaluatie 22

4.2 Sprint 1 basis cockpit 23

4.2.1 Taken 23

4.2.2 Ontwerpen 23

4.2.3 Bouwen 25

4.2.4 Testen 26

4.2.5 Evaluatie 27

4.3 Sprint 2 Database wijzigingen 28

4.3.1 Taken 28

4.3.2 Ontwerpen 28

4.3.3 Bouwen 30
4.3.4 Testen 31
4.3.5 Evaluatie 31
4.4 Sprint 3 modules 32
4.4.1 Taken 32
4.4.2 Ontwerpen 33
4.4.3 Bouwen 33
4.4.4 Testen 35
4.4.5 Evaluatie 36
4.5 Sprint 4 Modules deel 2 37
4.5.1 Taken 37
4.5.2 Bouwen 38
4.5.3 Testen 39
4.5.4 Evaluatie 39
4.6 Sprint 5 gevaarlijke situatie en klachten formulier 40
4.6.1 Taken 40
4.6.2 Ontwerpen 41
4.6.3 Bouwen 42
4.6.4 Testen 43
4.6.5 Evaluatie 43
4.7 Sprint 6 Checklists 44
4.7.1 Taken 44
4.7.2 Ontwerpen 45
4.7.3 Bouwen 45
4.7.4 Testen 46
4.7.5 Evaluatie 46
4.8 Sprint 7 Daglijsten 47
4.8.1 Taken 47
4.8.2 Ontwerpen 48
4.8.3 Bouwen 48
4.8.4 Testen 49
4.8.5 Evaluatie 49
4.9 Sprint 8 afronding 50

5 Evaluatie 51
5.1 Proces evaluatie 51
5.2 Product evaluatie 52
6 Beroepstaken 53
6.1 Competentie 1.4: Uitvoeren analyse door definitie van requirements 53
6.2 Competentie 3.2: Ontwerpen systeemdeel 53
6.3 Competentie 3.3: Bouwen applicatie 53
6.4 Competentie 3.5: Uitvoeren en rapporteren over het testproces 54
7 Literatuurlijst 55
8 Bijlagen 56
8.1 Woordenlijst 56
8.2 Bijlagenlijst 56

Inleiding

Voor u ligt het afstudeerverslag van de 17 weken durende afstudeeropdracht. Deze opdracht, in het kader van het afstuderen bij aan de opleiding Informatica aan De Haagse Hogeschool te Zoetermeer, is uitgevoerd bij het bedrijf NewCompliance.
Het doel van dit afstudeerverslag is om u inzicht te geven over het uitgevoerde afstudeerproject. Door middel van dit afstudeerverslag worden de examinatoren en gecommitteerden in staat gesteld een positief oordeel over de afstudeeropdracht te kunnen geven.

Om dit mogelijk te maken begint dit document met het beschrijven van de betrokken organisatie, NewCompliance. In hoofdstuk 2 wordt de opdracht besproken en de doelstelling en het gewenste resultaat. In hoofdstuk 3 wordt de gekozen aanpak naar voren gebracht. Dit wordt gedaan door eerst de huidige situatie te schetsen. Daarna wordt er een software ontwikkelmethode gekozen. Hierbij worden gekeken naar interactief ontwikkelen of ontwikkelen via waterval methode. Vervolgens worden Scrum en RUP tegen elkaar afgewogen. In hoofdstuk 4 worden de uitgevoerde sprints beschreven. Hierin wordt het proces beschreven en de uitgevoerde taken in de sprints. In hoofdstuk 5 wordt het proces en de opgeleverde producten geëvalueerd. In hoofdstuk 6 wordt het behalen van de beroepstaken besproken. Het document eindigt met een bronnenlijst en de bijlagen.

Als er in de tekst de volgende leestekens (..) wordt gevonden, dan wordt er naar een bron in de bronnenlijst die vermeld staat in hoofdstuk 7 verwezen. Verder is er in de bijlage een verklarende woordenlijst te vinden.

1. NewCompliance

1.1 Bedrijf

NewCompliance is een jong bedrijf opgericht in 2006 dat innovatieve producten, gericht op het verbeteren van patiëntveiligheid en productiviteit, ontwikkelt en vermarkt. NewCompliance heeft 14 fulltime medewerkers in dienst en er wordt hard gewerkt aan de ontwikkeling en vermarkting van verschillende nieuwe technologische producten.

NewCompliance richt zich volledig op het verbeteren van de patiëntveiligheid en productiviteit met behulp van innovatieve producten die gebruiksvriendelijk en toekomstgericht zijn. Hierbij ligt de nadruk op technologische oplossingen die gerealiseerd worden door een combinatie van hardware en software.

Het meest verkochte product van NewCompliance is het OK Cockpit. Hierop kunnen artsen en doctoren de variabelen (Bijv. luchtdruk, aantal openingen van de deur, temperatuur van de patiënt) zien in een oogopslag. Hiermee kan het slagingspercentage van operaties worden verhoogd en op een simpele wijze binnen de afgesproken normen worden gehouden. De gegevens die vergaard zijn via de OK Cockpit kunnen worden bekeken in een analyse tool. In de analyse tool kunnen een groot aantal filters worden toegepast op de gegevens.

In 2012 is NewCompliance een distributeur geworden van het Vernacare Systeem. NewCompliance is op dit moment de enige distributeur in Nederland van Vernacare. De producten van Vernacare zijn alternatieven voor de traditionele bedpan spoeler. Traditionele producten die worden gebruikt in ziekenhuizen zijn vaak metalen bedpannen en plastic urinalen. De producten van Vernacare zijn ‘pulp disposable’. Dit betekent dat de producten eenmalig te gebruiken zijn. Om de producten van Vernacare op te kunnen slaan is NewCompliance verhuist van Delft naar Zoetermeer. Dit is het huidige hoofdkantoor geworden van NewCompliance.

NewCompliance is begonnen met het ontwikkelen van de CSA (Centrale Sterilisatie Afdeling) Cockpit. Hierop kunnen medewerkers van de centrale sterilisatie afdeling net als bij de OK Cockpit belangrijke variabelen bekijken in een oogopslag. Dit moet de veiligheid en productie van de CSA verbeteren. (NewCompliance, n.d.)

1.2 Missie NewCompliance

De missie van NewCompliance ICT is volgens Bo Wiesman (CEO NewCompliance) als volgt: ”Het verbeteren van de patiëntveiligheid en de kwaliteit van de zorg door het invoeren van innovatieve systemen.”

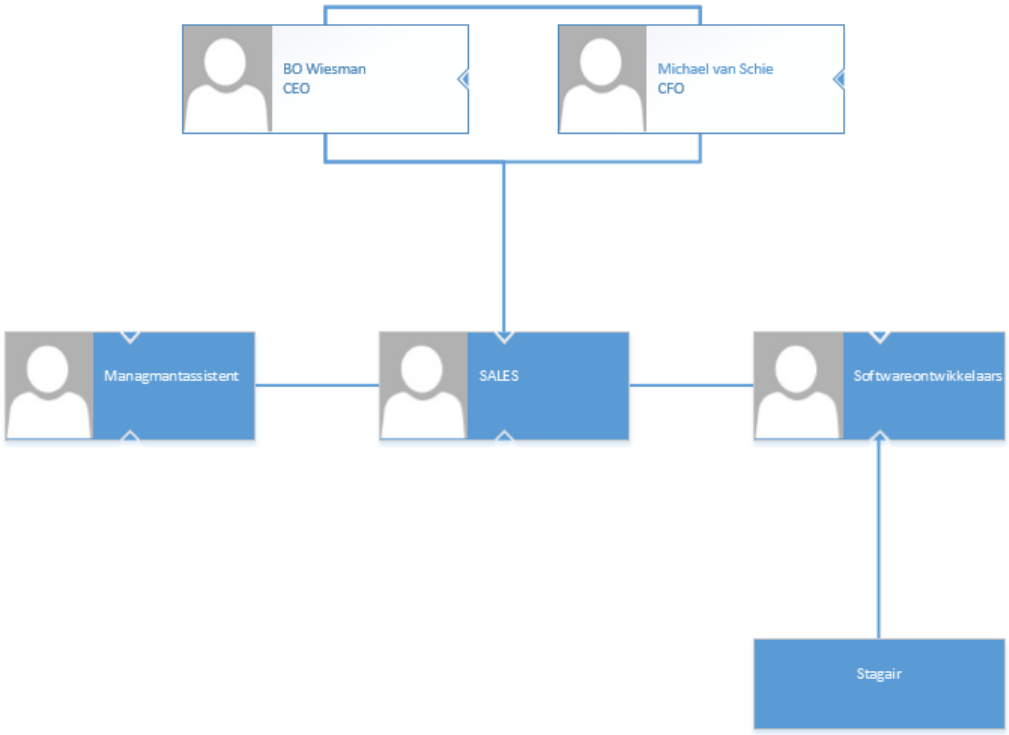
Dit missiestatement heeft betrekking op de twee soorten producten die NewCompliance aanbied aan zorginstellingen. Onder de innovatieve systemen worden de Ok Cockpit, Ok Monitoring en Ok Deurtellingen systeem genoemd. Dit zijn systemen die de veiligheid van de patiënt op de operatiekamer kunnen vergroten.

De doelstelling op korte termijn van NewCompliance is verder groeien. Op dit moment worden er in vijf ziekenhuizen het OkMonitering systeem geïnstalleerd. NewCompliance wil in het komende jaar nog eens tien ziekenhuizen voorzien van een OK systeem. Verder wil NewCompliance een grote update doen van de huidige OK systemen. Deze zullen dan worden aangeboden aan de huidige klanten.

Op de lange termijn is het doel van NewCompliance om marktleider te worden in het verbeteren van de zorg en de patiëntveiligheid. De manier waarop dit zal worden bereikt is door de bestaande systemen verder te innoveren.

1.3 Interne organisatie

NewCompliance is een vrij jong bedrijf. Het bedrijf heeft op dit moment 10 werknemers. Hierdoor is het organogram niet erg groot. De functie CFO en Sales worden door dezelfde persoon bekleed, namelijk Michael van Schie. De functie CEO en ook een deel van de sales worden bekleed door Bo Wiesman. Bo Wiesman heeft de verantwoording over de ict afdeling en dus het Ok Cockpit systeem en geeft daardoor leiding aan de softwareontwikkelaars. Michael van Schie is verantwoordelijk voor de Vernacare afdeling, de verkoop van Vernacare producten en de financiële administratie van NewCompliance.



De stagiair wordt begeleid door één van de softwareontwikkelaars. Binnen NewCompliance is het afgesproken dat er elke maandag een stand up meeting gehouden wordt. In deze meeting worden de huidige werkzaamheden besproken. Hieronder valt ook de vooruitgang van de stagiair. In deze meeting kunnen werknemers en stagairs ideeën naar voren brengen. Ook wordt er samen met de stagairs wekelijks een code review uitgevoerd. In deze code review wordt samen met de hoofdprogrammeur de code te besproken van die week. Hiermee kan worden bepaald of de stagiair op het juiste pad is of dat er bijgestuurd moet worden. Als de stagiair buiten deze afspraken nog vragen heeft kan hij altijd naar een softwareontwikkelaar lopen om zijn vragen te stellen.

2. Opdracht

In dit hoofdstuk wordt de uitgevoerde opdracht besproken. Hierbij wordt de aanleiding van de opdracht naar voren gebracht. Ook wordt de huidige situatie met de ontstane probleem geschetst. Verder wordt de doelstelling van de opdracht, het uiteindelijke resultaat en de gewenste situatie beschreven. Verder zal het hoofdstuk afsluiten met een opsomming van de belangrijkste werkzaamheden.

2.1 Initiële opdracht

Voor de start van het afstudeertraject in mei 2015 is de volgende opdrachtomschrijving voorgelegd. Deze opdrachtomschrijving is te vinden in de bijlage van het afstudeerplan:

Het huidige product OK Cockpit is ontwikkelt voor ziekenhuizen om in de OK alle medische data snel toegankelijk te hebben. Nu willen de ziekenhuizen ook een soort gelijk systeem voor de sterilisatie afdeling, zodat er gemakkelijk gezien kan worden wat er gedaan moet worden, wat er gedaan is en of een bepaalde opdracht spoed heeft. De ziekenhuizen willen het systeem over een half jaar in gebruik nemen. De huidige OK Cockpit is niet aanpasbaar genoeg om hergebruikt te worden voor de nieuwe Cockpit. Er kunnen delen van de Cockpit hergebruikt worden maar een groot deel moet herschreven worden om toepasbaar te zijn voor de nieuwe Cockpit. Voor de nieuwe Cockpit moeten ook een paar belangrijke algoritmes bedacht worden om de grote hoeveelheid data juist weer te geven in de Cockpit met gebruik van tabellen.

2.2 Probleemstelling

Het huidige product OK Cockpit is ontwikkelt voor ziekenhuizen om in de OK alle medische data snel toegankelijk te hebben. De OK Cockpit is specifiek gemaakt voor de OK afdeling. Nu wil het Erasmus MC ook een soort gelijk systeem voor de CSA (Centrale Sterilisatie Afdeling), zodat er gemakkelijk gezien kan worden wat er gedaan moet worden, wat er gedaan is en of een bepaalde opdracht spoed heeft. Het ziekenhuis wil het systeem over een half jaar in gebruik nemen.

De huidige OK Cockpit moet aangepast worden naar een MED (medische) Cockpit. Deze MED Cockpit bevat de functionaliteiten van de OK cockpit en de nieuwe CSA Cockpit. Als eerste zal de MED Cockpit gemaakt moeten worden. De OK Cockpit wordt als basis gebruikt om de MED Cockpit te maken. In de OK Cockpit staat het uitvoeren van een operatie centraal en daar is de software naar toe gebouwd. In de MED Cockpit moet elke module modulair toepasbaar zijn in elke kamer. Alleen waar nodig een koppeling hebben met een operatie. Dit zorgt ervoor dat de MED Cockpit aanpasbaar wordt voor meerdere ziekenhuis afdelingen.

De huidige software zal dus deels opnieuw ontworpen moeten worden. Vervolgens zullen er wijzigingen in de software gemaakt moeten worden.

Nadat de OK Cockpit omgebouwd is naar de MED Cockpit zullen er nieuwe functionaliteiten gemaakt moeten worden voor de CSA. Deze nieuwe functionaliteiten zullen met het Erasmus MC tijdens het project vastgesteld worden.

2.3 Doelstelling

De afstudeeropdracht heeft twee doelstellingen. De eerste doelstelling is het aanpassen van de OK Cockpit. De OK Cockpit moet niet meer afhankelijk zijn van het uitvoeren van een operatie zoals hierboven beschreven is. De tweede doestelling is het toevoegen van functionaliteiten die samen met de CSA van het Erasmus MC vastgesteld zullen worden. De eerste doelstelling moet volbracht zijn om de tweede uit te kunnen voeren.

2.4 Gewenste resultaat

Het gewenste resultaat is een MED Cockpit. Deze Cockpit is ontworpen en gebouwd naar de eisen en wensen van de opdrachtgever. Met deze Cockpit kan ook nog de oude functies van de OK Cockpit uitgevoerd worden. Ook zal de MED Cockpit de functie moeten kunnen uitvoeren die met de CSA zijn afgestemd. De functionaliteiten zullen worden getest aan de hand van de requirements. Ook wordt aan de Haagse Hogenschool een afstudeerverslag opgeleverd. De volgende producten zullen opgeleverd worden.

- Plan van aanpak
- Requirements rapport
- Functioneel ontwerp
- Techniche ontwerp
- MED Cockpit
- Testrapport

3. Aanpak en Methode

In dit hoofdstuk wordt de werking van de OK Cockpit beschreven. Deze uitleg beschrijft de bestaande software waar in dit project op verder gebouwd gaat worden. In de tweede paragraaf wordt de gekozen software ontwikkelmethode naar voren gebracht. Ook wordt de argumentatie gegeven over de gekozen methode.

3.1 OK Cockpit

In dit hoofdstuk zal de werking van de Ok Cockpit beschreven worden. Dit is noodzakelijk om de rest van het verslag te kunnen begrijpen.

Om de software uit te leggen neem ik een voorbeeld van een OK(Operatie kamer). Deze kamer is deel van een OK afdeling. Elke van deze kamers zijn in de database ingevoerd. Voor het tonen van een op maat gemaakt Dashboard voor één of meerdere kamers wordt er een layout ingesteld. Deze layout bevat de modules die getoond moeten worden op het scherm en op welke plek deze staan. De modules zijn modulaire blokjes die bepaalde informatie tonen. Voor elke layout die is ingesteld toont het systeem alle ingevoerde kamers in combinatie met het aantal layouts. Dit is te zien in het onderstaande plaatje.

Er is op het bovenstaande plaatje te zien dat er 6 kamers zijn ingesteld met 2 verschillende layouts. Bovenstaand plaatje presenteert dus 12 mogelijke dashboards. Er wordt op de gewenste layout van de gewenste kamer gedrukt. Voor OK-2 zal het scherm er dan als volgt uitzien.

Op het bovenstaande plaatje is een dashboard te zien met een layout die 22 modules bevat. Om de modules waardes te laten tonen is er connectie met het ziekenhuis systeem. Uit het systeem wordt periodiek data opgehaald om die te kunnen tonen op het dashboard.

Als er een operatie gestart wordt in OK-2 wordt er op het scherm de gewenste informatie getoond. Deze informatie helpt de OK medewerkers tijdens een operatie. Dit omdat real time informatie over de patiënt en de kamer overzichtelijk toegankelijk is. Beschikbare informatie is bijvoorbeeld allergieën en lichaamstemperatuur van de patiënt, maar bijvoorbeeld ook de lucht kwaliteit in de OK.
Aan het eind van een operatie wordt alle data van de operatie verzameld. Een rapport wordt opgebouwd nadat alle data verzameld is. Dit rapport toont de belangrijkst punten van een operatie. Dit is bijvoorbeeld de duur van de operatie, de toegediende medicatie en het aantal deuropeningen tijdens de operatie. Hieruit wordt een cijfer berekend naar een vooraf afgestelde formule. Dit cijfer toont de OK medewerkers waar mogelijk verbeteringen toegepast kunnen worden. Het doel van dit cijfer is om de zorg te verbeteren en mogelijk ook in de toekomst te kunnen gaan vergelijken tussen ziekenhuizen. In de onderstaand afbeelding is een voorbeeld rapport te zien.

3.2 Methodiek

NewCompliance heeft niet een standaard ontwikkelmethode. Ze gebruiken wel enkele best practices uit bekende ontwikkel methodes. Zoals het maken van een plan van aanpak om met ziekenhuizen het doel van het project te bepalen. Ook worden er binnen NewCompliance wekelijkse stand up meetings gehouden om de voortgang te bespreken.

Voor een Ontwikkelmethode was ik vrij om te kiezen. Daarom heb ik enkele ontwikkelmethodes vergeleken om de juiste te kunnen kiezen. Hieronder staat de keuze voor ontwikkelmethode beschreven.

3.2.1 Keuze

In dit hoofdstuk beschrijf ik de door mij gekozen ontwikkelmethode. Hierin heb ik eerst de overweging gemaakt of een iteratieve methode of een waterval methoden het beste bij het project paste. Vervolgens ben ik gaan kijken binnen de methode welke soort het beste zou passen. Om deze keuzes te maken heb ik de risico's van het project gebruikt. Dit omdat een juiste keuze van ontwikkeltechniek de risico's verminderen.

Uit de opdracht omschrijving werd duidelijk dat er nog niet veel vast staat in dit project. In de opdracht omschrijving staat dat als eerste het systeem aangepast moet worden en vervolgens er uitbreidingen gemaakt moeten worden. Het is nog onduidelijk welke uitbreidingen er op het systeem gedaan moeten worden

Waterval methodes zijn methodes die lineair uitgevoerd dient te worden (Waterfall Model, 2015). Bij waterval methodes wordt iedere fase eenmalig uitgevoerd. Daardoor is het niet mogelijk om eerst de huidige software aan te passen en daarna functionaliteiten bij te bouwen voor de CSA. De waterval methode is uitermate geschikt voor projecten met een grote kostenrisico. Het project is een experiment om de klanten basis te vergroten. Het kostenrisico is niet hoog omdat het project zal worden uitgevoerd door een stagiair. De opdrachtgever verwacht meerdere oplever momenten om deze met het ziekenhuis te kunnen bespreken en omdat het project een laag kostenrisico heeft is het gebruik van de watervalmethode afgevallen.

Iteratieve methodes zijn methodes die iteratief uitgevoerd dient te worden. Dit betekent dat elke fase meerdere keren herhaald kan worden. Daardoor wordt het mogelijk om eerst de huidige software aan te passen en daarna functionaliteiten bij te bouwen voor de CSA. Dit is een belangrijke eis voor dit project. Daarom is er gekozen om een iteratieve methode te gaan gebruiken.

Hieronder zal ik twee iteratieve methodes doorlopen die ik overwogen heb om te gebruiken. Dit zijn RUP en Scrum.

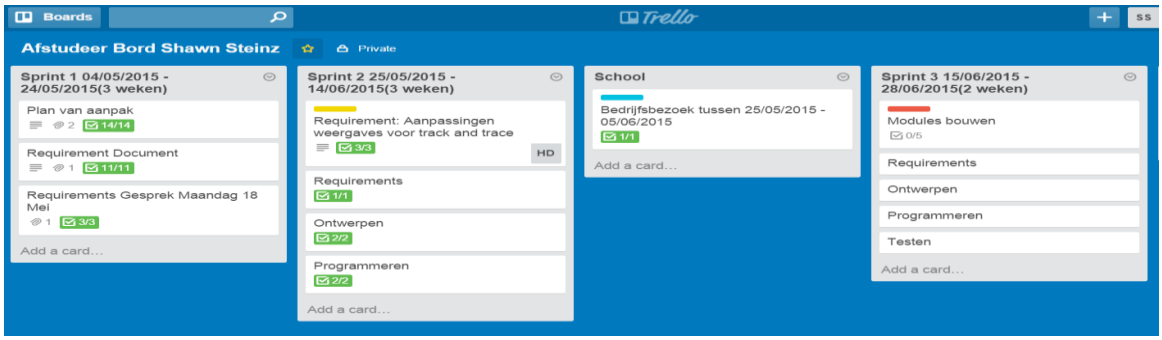
RUP is een iteratieve software methode (Temnenco, 2006). In deze methode wordt van te voren de iteraties gepland. Uit de opdracht omschrijving zijn er minimaal twee iteraties naar voren komen. De opdrachtgever heeft aangegeven tijdens het project betrokken te willen zijn. De mogelijkheid tot meerdere iteraties, waarbij er steeds een stuk software wordt gemaakt lijkt hier goed op aan te sluiten. Bij RUP is wel een vereiste dat de requirements van het project niet (veel) verschuiven. De requirements zullen tijdens het project waarschijnlijk nog veel gaan veranderen omdat de opdrachtgever en het ziekenhuis nog niet duidelijk weten wat zij willen. Dit sluit RUP uit omdat er niet van te voren de iteraties gepland kunnen worden.

Een andere software ontwikkelmethode, die ik heb overwogen, is Scrum (scrumalliance, n.d.). Scrum geeft ruimte om het project aan te passen aan veranderende eisen. Uit de opdrachtomschrijving blijkt dat er nog geen duidelijk beeld is van de applicatie bij de stakeholders. Dit vormt een risicofactor. Deze kan niet worden afgedekt omdat de requirements tijdens het project nog zullen gaan veranderen. Ik heb daarom gekozen om Scrum te gebruiken als software ontwikkelmethode.

De keuze is gevallen op Scrum omdat er veel contact met de opdrachtgever is en het ziekenhuis betrokken wil zijn bij het ontwikkel proces. Het bedrijf eist geen documentatie daarom is een andere ontwikkelmethode kijkend op documentatie overbodig. Er was bij aanvang van het project nog geen vastgestelde scope voor het project. Veel keuzes lagen nog open. Dit maakte het plannen van het project moeilijk omdat er nog niet helemaal duidelijk was wat er gedaan moest worden.

Er zijn wel enige aanpassingen gemaakt aan de standaard van Scrum. De dagelijkse meetings van Scrum zijn omdat dit project maar één lid heeft van het ontwikkelteam overbodig. In plaats daarvan wordt 4 keer per sprint een “Daily Scrum” meeting gehouden. Wel wordt de product-owner nauw bij het project betrokken gehouden. Omdat er bewijzen nodig zijn voor de door de school opgelegde competenties zijn de in hoofdstuk 2.4 genoemde documenten gemaakt.

Als Scrum product en sprint backlog heb ik gebruik gemaakt van een Trello bord. Hieronder is een stuk van dat bord te zien.



Zoals te zien is staat per sprint aangeven wat ik gedaan heb en wat ik die sprint ga doen.

3.2 Gesprekken

Zoals eerder beschreven heb ik vier Scrum meetings per sprint gedaan met het ontwikkelteam. Ook werd er per sprint enkele keren overleg gepleegd met de opdrachtgever, door middel van een vergadering of telefonisch overleg.

In Scrum zijn meetings ge-timeboxed zodat deze meetings niet kunnen uitlopen. Dit zorgt ervoor dat meetings duidelijk en gestructureerd zullen gaan.

Na elke sprint hebben we een Sprint meeting gehouden om de werkende functionaliteiten met de klant te bespreken en te bepalen wat er de komende sprint gedaan zou worden. Dit zorgt voor een goede sturing voor het proces. De meetings zijn ge-timeboxed naar onderstaande tabel.

Activiteit	**Regelmaat**	**Tijd**
Sprint meeting	1X per sprint	1 uur
Voortgang meeting		4X per sprint

4 Werkzaamheden

In dit hoofdstuk worden alle werkzaamheden die tijdens de afstudeerstage zijn uitgevoerd beschreven. De werkzaamheden worden per sprint beschreven.

4.1 Sprint 0 project opzet

In sprint 0 ben ik begonnen met het maken van een plan van aanpak. In dit plan van aanpak heb ik het afstudeerplan uitgewerkt tot een werkwijze. Tevens heb ik hiermee de scope van het project samen met de product-owner vastgesteld. In dit document staat ook beschreven welke methodiek ik heb gebruikt voor het project. De risicofactoren die een risico vormen voor het slagen van het project staan hier ook beschreven. Om een goed plan van aanpak te maken moest ik mijzelf eerst inlezen in de Scrum methodiek. Vervolgens heb ik een interview met Henk van den Dries (mijn stagebegeleider) gehouden om een globale indruk te krijgen van de bestaande software. Tevens heb ik mij ingewerkt in het PHP framework Laravel.

4.1.1 Aangetroffen staat

Omdat ik een toevoeging ga maken op bestaande software ben ik begonnen met het analyseren wat er al aanwezig was. NewCompliance is een bedrijf die niet veel documenteert. Daarom was er vrij weinig documentatie aanwezig bij de start van mijn afstuderen. Wat wel aanwezig was waren verouderde installatie handleidingen en een oud bedrijfsoriëntatie document. Deze heb ik wel kunnen gebruiken om een beter inzicht te krijgen in het bedrijf en de benodigde software voor de Cockpit applicatie. Vervolgens ben ik in gesprek gegaan met Henk van den Dries om een beeld te krijgen van de werkzaamheden van de Cockpit.

Hierna heb ik de broncode van Cockpit geïmporteerd in mijn programmeeromgeving. De programmeeromgeving binnen NewCompliance is PhpStorm. Zij gebruiken PhpStorm omdat deze programmeeromgeving gemaakt is voor PHP waar de Cockpit in geschreven is. PhpStorm zorgt ook voor een goede ondersteuning voor HTML, CSS en javascript. Dit zijn andere componenten van de Cockpit. Tevens ondersteund PhpStorm GitHub. Dit is een Git Cloud opslag voor bron code. NewCompliance gebruikt GitHub om in een online omgeving hun broncode op te slaan zodat hun broncode veilig is opgeslagen en overall bereikbaar is.

In de installatie handleiding stonden enkele applicatie die geïnstalleerd moesten worden om de Cockpit draaiende te krijgen. Dit zijn PHP, Apache, PostgreSQL, Mirth en tevens de bovenstaande programmeeromgeving PhpStorm.

4.1.2 Planning

In de eerste sprint heb ik een sprint planning in het plan van aanpak opgesteld. Deze planning was bedoeld om overzichtelijk de sprint te kunnen plannen en ook de gesprekken die daarbij horen te kunnen inplannen. Ik ben begonnen met sprints van 3 weken en vervolgens naar 2 weken gegaan. Dit is niet gebruikelijk voor Scrum. De reden dat ik hier toch voor gekozen heb, was omdat er tijdens de opzet van het project nog zoveel onduidelijkheden waren, dat 3 weken een logische keuze was. Maar omdat het ziekenhuis en NewCompliance om de 2 weken voortgang wilden zien heb ik ervoor gekozen om tijdens de sprint planning af te wijken van de standaard. Daarom zijn de eerste twee sprints drie weken lang en sprint twee tot zeven twee weken lang. Sprint 8 is maar 1 week lang, omdat dat de laatste week van mijn afstuderen was. Zie bijlage B voor verdere informatie van het plan van aanpak.

Sprint	**Start**	**Einde**	**Beschrijving**
0 project opzet	04/05/2015	24/05/2015(3 weken)	Het opzetten van de afstudeerstage inwerken in de code, scrum en opstellen van plan van aanpak.
1	25/05/2015	14/06/2015(3 weken)	Het bouwen van de basis van de Cockpit
2	15/06/2015	28/06/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
3	29/06/2015	12/07/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
4	13/07/2015	26/07/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
5	27/07/2015	09/08/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
6	10/08/2015	23/08/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
7	24/08/2015	06/09/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
8	07/09/2015	13/09/2015(1 week)	Het afronden van het afstudeerproject

4.1.3 Requirements

Aan het begin van het project is het probleem geschetst. Hierbij is naar voren gekomen dat de huidige software een belemmering geeft in het verdere uitbreiden van de ICT diensten. Ook zijn er wensen naar nieuwe functionaliteiten. Om de exacte functionaliteiten te kunnen vaststellen worden de eisen en wensen, vanaf nu requirements genoemd, achterhaald. Dit wordt gedaan door contact te leggen met de stakeholders. Dit zijn de opdrachtgever en het Erasmus MC ziekenhuis.

De requirements worden geprioriteerd aan de hand van de MoSCoW methode. Aan het begin van elke sprint wordt de prioritering van de requirements door de opdrachtgever bepaald. Door prioriteit te geven aan de requirements kunnen de meest essentiële functionaliteiten worden vastgesteld, zodat er bij elke sprint de belangrijksten functionaliteiten gebouwd kunnen worden. In de MoSCoW methode worden vier niveaus van prioriteit gehanteerd. Dit zijn de volgende niveaus:

- M - must haves: deze requirements moeten in het eindresultaat terugkomen, zonder deze requirements is het product niet bruikbaar;
- S - should haves: deze requirements zijn zeer gewenst, maar zonder is het product wel bruikbaar;
- C - could haves: deze requirements zullen alleen aan bod komen als er tijd genoeg is;
- W – won't haves: deze requirements zullen in dit project waarschijnlijk niet aan bod komen maar kunnen in de toekomst, bij een vervolgproject, interessant zijn.

Bij aanvang van het project waren er op een trello bord enkele eisen opgeschreven. Deze eisen heb ik verwerkt in de onderstaande requirements:

Requirement ID	Requirement
R1	Een CSA medewerker moet een gevaarlijke situatie formulier kunnen invullen in het systeem.
R2	Een OK medewerker moet klachtformulier kunnen invullen in het systeem.
R3	De Dagoudste moet een checklist kunnen uitvoeren genaamd RIC controle.
R4	Een medewerker moet een daglijst kunnen invullen en wijzigen.
R5	Er moeten views gemaakt worden voor de afdelingen van de CSA.

Vervolgens ben ik in gesprek gegaan met het Erasmus MC en de opdrachtgever. Ik heb ze toen deze requirements voorgelegd en gevraagd of deze requirements nog van toepassing waren. Vervolgens zijn we gaan brainstormen om alle nieuwe ideeën naar voren te brengen. Ook hebben we de bestaande requirements besproken en deze specifieker gemaakt.

Na het gesprek ben ik de aantekening gaan verwerken in de requirements. De hieronder gepresenteerde requirementslijst was het resultaat:

Requirement ID	Requirement
R1	Een CSA medewerker moet een gevaarlijke situatie formulier kunnen invullen in het systeem.
R2	Een OK medewerker moet klachtformulier kunnen invullen in het systeem.
R3	De Dagoudste moet een checklist kunnen uitvoeren genaamd RIC controle.
R4	Een medewerker moet een daglijst kunnen invullen en wijzigen. Omdat de werklast binnen deze afdelingen kan verschillen van uur tot uur is het belangrijk dat binnen deze invoer wijzigingen aan te brengen zijn.
R5	Vuile ruimte dit scherm bevat informatie over Flex scopen, capaciteit apparatuur t.o.v. volume in aantocht, spoed netten, spoed plan en de werklast.
R6	Verpakruimte 1 dit scherm bevat informatie over verlooptijd Flex scopen, klachten van de ok, spoed net, spoed plan, productie % tegen over klachten, werklast en prestatie cijfer.
R7	Verpakruimte 2 dit scherm bevat informatie over netten in reparatie, capaciteit apparatuur t.o.v. volume in aantocht en bezetting
R8	Uitgifte CSA dit scherm bevat informatie over flex scopen, netten, spoed netten en spoed plan
R9	Dagoudste dit scherm bevat informatie over verlopen eenheden, bd-waarde, OK klachten, gevaarlijke situatie meldingen en de ingevulde checklists.
R10	OK's deze schermen tonen per OK de volgende informatie verlopen eenheden, netten uitgeleend, netten reparatie, spoed plan, spoed net en gevaarlijke situatie meldingen.
R11	In de bestaande analyse tool moet er een rapport gemaakt kunnen worden met betrekking op netten, tijdsduur, kwaliteit, ziekteverzuim, gebruiksintensiteit van netten, prestatie van medewerkers, kostenplaats en verschillen afwijkend gewicht tussen OK en CSA.
R12	De medewerkers van de CSA moeten een apparaat checklist kunnen invullen.

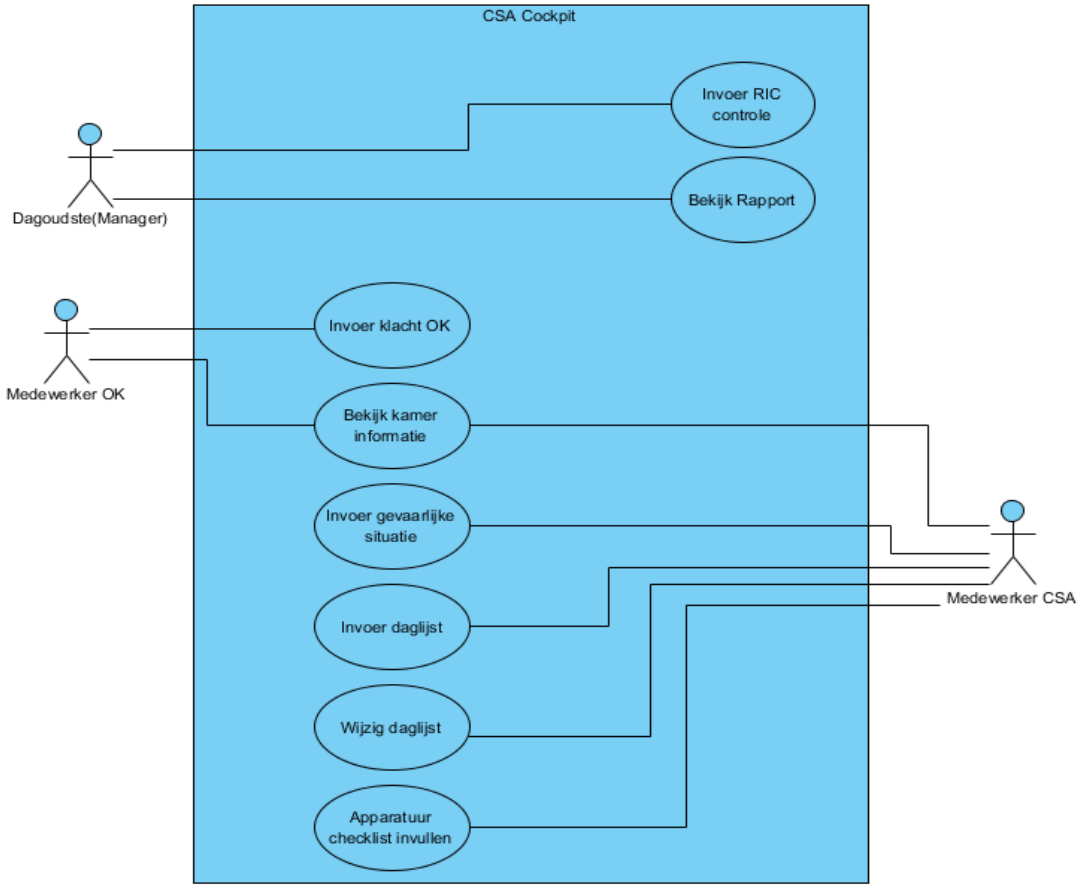
De requirements zijn nog niet heel specifiek omdat de opdrachtgever en het ziekenhuis nog niet helemaal weten wat ze willen. Na het opstellen van de requirementslijst ben ik de niet-functionele software requirements en functionele software requirements gaan opstellen. De niet-functionele software requirements beschrijven de eisen die gebruikt kunnen worden om het gedrag van het systeem te kunnen beoordelen. De functionele software requirements geeft gewenst gedrag aan.
Uit het plan van aanpak en het gesprek zijn de volgende functionele software requirements en niet-functionele software requirements naar voren gekomen:

Requirement ID	Requirement
NFSR1	Er moet een connectie met T-DOC en 3M gemaakt worden om data input te krijgen voor de CSA Cockpit.
NFSR2	De Cockpit moet real time data kunnen tonen
NFSR3	De Cockpit moet gebruik maken van een MSSQL database
FSR1	De werklast van de CSA moet berekent kunnen worden door het systeem. Dit geld voor alle afdelingen apart en ook gezamenlijk.

Na het opstellen van alle requirements heb ik een actor list gemaakt, waarin alle gebruikers van het systeem en hun rol beschreven worden. Hierdoor konden de actoren in het use case diagram gebruikt worden. Hieronder is deze tabel te zien.

Actor	Role	Description
CSA Medewerker	User	De actor kan daglijsten invoeren en wijzigen ook kan hij gevaarlijke situatie formulier invoeren en ook kan hij de Dashbord module bekijken.
OK medewerker	User	De actor kan een OK klacht invoeren en hij kan de Dashbord module bekijken.
Dagoudste (Manager)	Superuser	De actor kan RIC controle invoeren en data bekijken in de analyse tool.

Nadat de actor list gemaakt was ben ik begonnen met het maken van een use case diagram. In deze use case diagram worden de gebruikersscenario's opgesteld. Deze ontstaan door het categoriseren van de requirements.



De use case diagram laat duidelijk zien welke actoren welke taken in het systeem moeten kunnen uitvoeren.

Na het opstellen van het use case diagram ben ik met behulp van de hierboven uitgelegde MoSCoW methode samen met de opdrachtgever de requirements gaan prioriteren. Hieronder is het resultaat gepresenteerd:

Use Case ID	Use Case	Description	Requirement nr	Prioriteit
U1	Invoer gevaarlijke situatie	De CSA medewerker vult het gevaarlijke situatie formulier in.	R1	M
U2	Invoer klachten OK	De OK medewerker vult het klachten formulier in.	R2	M
U3	Invoer RIC controle	De Dagoudste op de CSA vult het RIC formulier in.	R3	M
U4	Invoer daglijst	De CSA medewerker vult de daglijst in.	R4	M
U5	Wijzig daglijst	De CSA medewerker wijzigt een daglijst.	R4	M
U6	Bekijk kamer Module	De actor ziet een informatie scherm die is gebaseerd op de kamer waar hij zich bevind.	R5, R6, R7, R8, R9, R10	M
U7	Bekijk Rapport	De Dagoudste kan hier alle opgeslagen gegevens in een rapport weergeven.	R11	C
U8	Apparatuur checklist invullen	De CSA medewerker vult de apparatuur checklists in.	R12	M

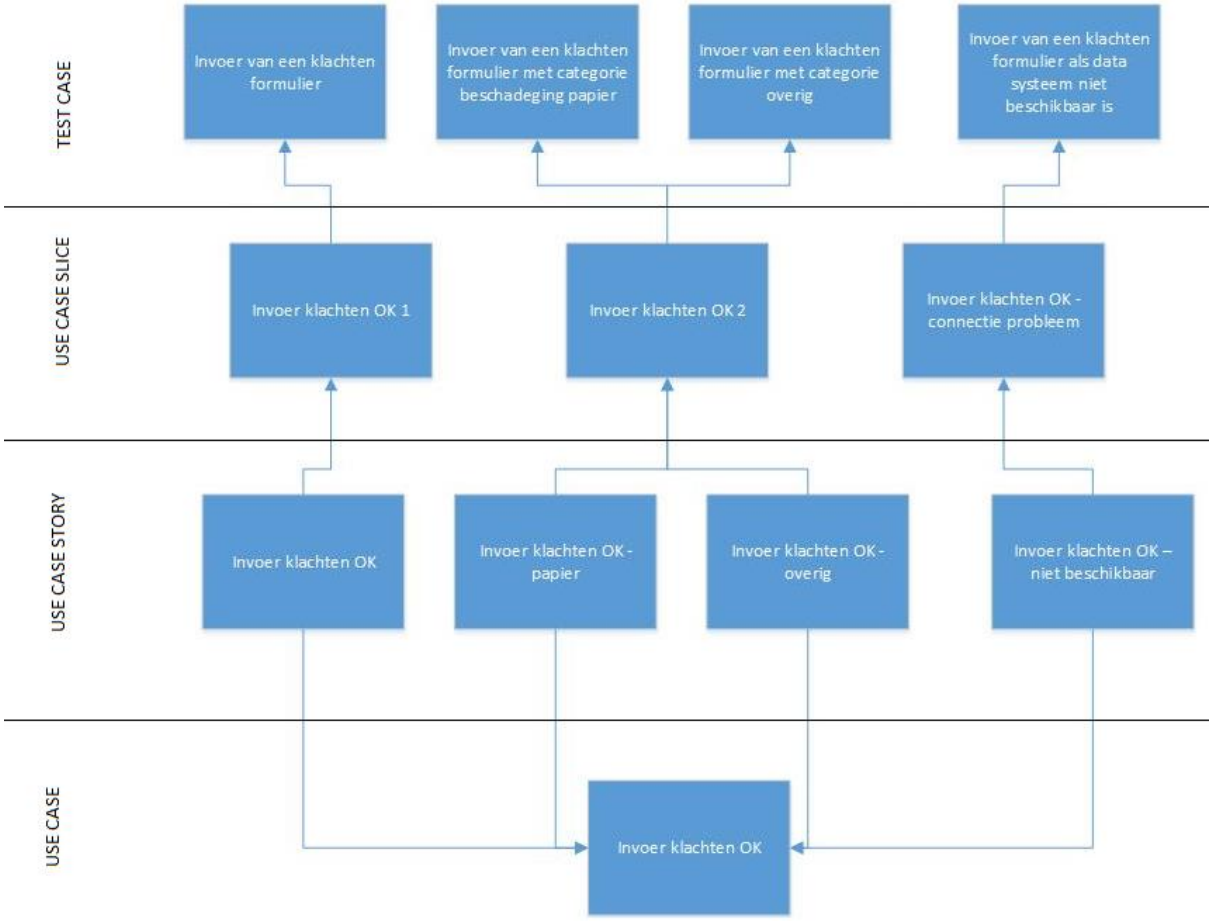
In de bovenstaande tabel is te zien dat bijna alle use cases nodig zijn om een bruikbaar programma te kunnen afleveren. Na het uitvoeren van de prioriteitssessie ben ik met het ziekenhuis in gesprek gegaan om op een dieper niveau de use case beschrijvingen te kunnen gaan maken. Van de Must have requirements heb ik vervolgens use case beschrijvingen gemaakt.

Na het maken van de use case beschrijvingen kwam ik tot de conclusie dat de use cases te groot waren om later te kunnen gaan uitvoeren in sprints. Daarom heb ik ervoor gekozen om de use cases op te delen in use case storys en vervolgens in use case slices. Een use case story is een use case uitgewerkt met al zijn alternate flows. Een alternate flow is een alternatieve uitkomst van een actie in het systeem. Vervolgens heb ik de use case storys omgezet in use case slices door de use case story te sorteren onder verzamelnamen per use case. Na dit proces waren er kleinere use case slices die aan het begin van de sprint verdeeld konden worden om uitgevoerd te kunnen worden.

Bron: Use case proces (Schaaf, 2012) terug te vinden in de bronnenlijst.

Van de gemaakt use case slices kunnen gemakkelijk test cases gemaakt worden, wat er voor zorgt dat er een goede basis voor het testen is.

In de onderstaande afbeelding is één zo'n opsplitsing te zien:



In de bovenstaande afbeelding is te zien dat de use case Invoer klachten OK opgesplitst wordt. In iedere sprint zullen de use case slices geprioriteerd worden. De use case slices met de hoogste prioriteit zullen gedurende die sprint uitgevoerd en getest worden volgens de bijbehorende test cases. Voor meer informatie verwijs ik naar bijlage C: het requirements rapport.

4.1.4 Laravel

Bij aanvang werd mij verteld dat ik zou gaan werken met het framework Laravel. Daarom ben ik online gaan kijken wat dit inhield en hoe ik het best ermee zou kunnen gaan werken.

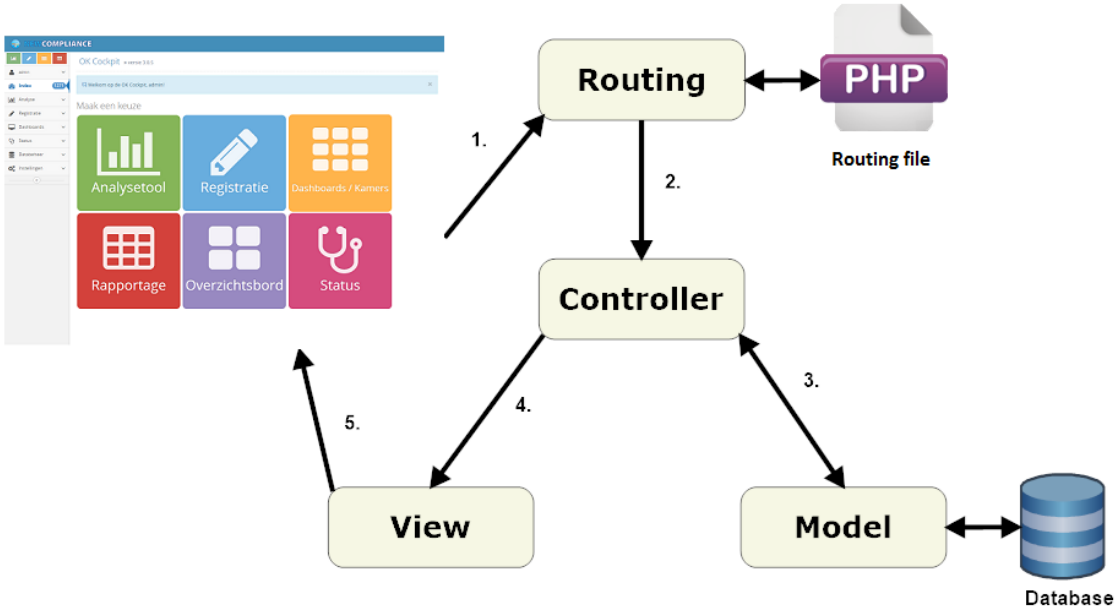
Laravel is een framework voor PHP projecten. Het geeft het project structuur, zorgt voor gebruik van design pattern en gebruikt zijn eigen ORM (Object-relational mapping) genaamd Eloquent. Laravels motto is: “Convention over configuration”. Hiermee wordt bedoelt dat het aantal beslissingen voor de ontwikkelaar wordt verminderd, waardoor meer eenvoud ontstaat, maar niet perse aan flexibiliteit wordt ingeleverd. Hieronder zal ik per onderdeel een beschrijving geven.

Een Laravel project gebruikt altijd dezelfde projectstructuur. Hierdoor kunnen ontwikkelaars aan meerdere projecten werken en ze kunnen nog steeds alles vinden. Dit zorgt voor een goede onderhoudbaarheid van de code. Dit zorgt ervoor dat een nieuwe ontwikkelaar sneller kan instromen.

Laravel werkt met het MVC (Model View Controller) design pattern. Dit is ook doorgevoerd in de project structuur. MVC design pattern zorgt voor een duidelijke scheiding tussen Model en View door gebruik van controllers. Het model definieert de representatie van de gegevens waar de applicatie mee werkt. De view toont de informatie aan de gebruiker. De controller zorgt voor de handeling van events die meestal voortkomen uit handelingen van de gebruiker. Het gebruik van MVC zorgt voor een betere leesbaarheid en herbruikbaarheid van de applicatie. Bijvoorbeeld veranderingen in de View hebben geen invloed om de model.

Laravel werkt met zijn eigen ORM (Object-relational mapping) genaamd Eloquent. Een ORM bestaat uit mappings die een brug vormen tussen een datasource en een applicatie. In deze mappings wordt de data gemapped naar object. Dit wordt gedaan om object georiënteerd te kunnen programmeren en om een modulariteit toe te kunnen voegen. Zo kan de programmatuur gemakkelijk hergebruikt en uitgebreid worden.

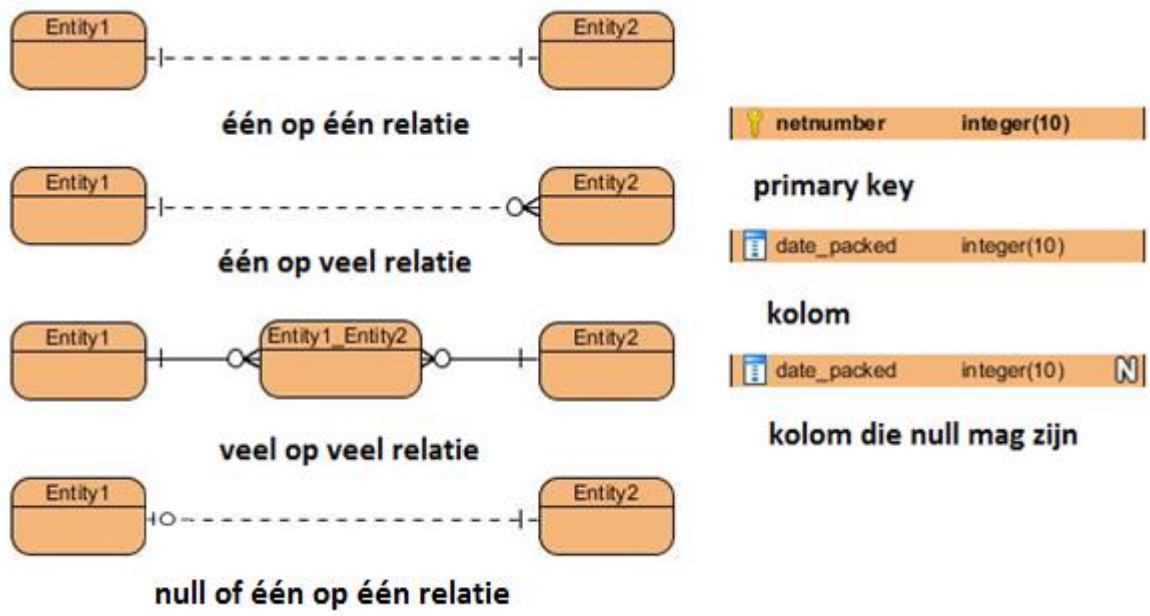
Om de werking van Laravel beter te beschrijven heb ik het onderstaande plaatje gemaakt die het proces van Laravel toont. Daaronder zal ik het proces tekstueel beschrijven.



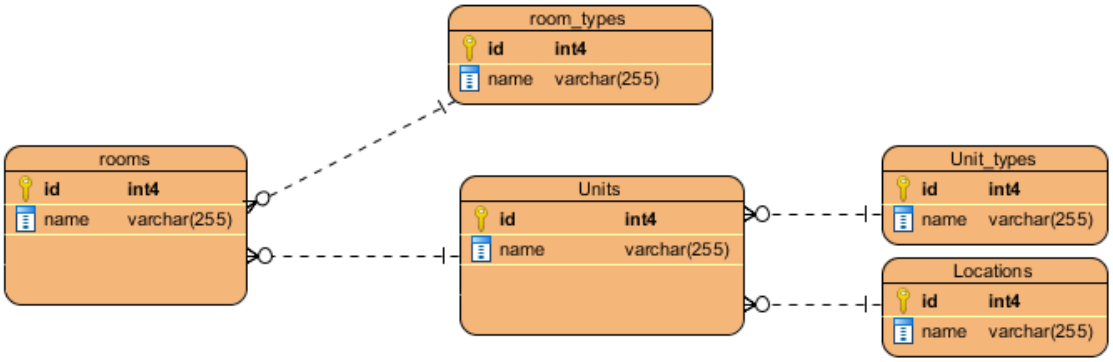
Stap 1: de user vraagt via de browser een pagina aan.
Stap 2: De routing roept de juiste controller aan. Laravel maakt gebruik van een Routing file om dit gemakkelijk te kunnen bewerkstelligen.
Stap 3: de controller heeft interactie met de model. Hier wordt met behulp van eloquent mapping de data uit de database gehaald. Vervolgens wordt in de aangeroepen model class de data getransformeerd om klaar gezet te worden voor de view.
Stap 4: de Controller roept de view op die bij de user aanvraag hoort en geeft de data mee.
Stap 5: de View wordt opgebouwd met de data en getoond in de browser.

4.1.5 ERD (Entity–relationshipmodel) diagram huidige situatie

In dit document zal ik veelvuldig gebruik maken van ERD diagrammen. Voor deze diagrammen heb ik een legenda gemaakt die de semantique beschrijft. Deze is hieronder te zien.



Tijdens dit project zal ik gebruik maken van de database die binnen NewCompliance al bestaat. Deze bestaat uit 79 tabels. Deze zijn nodig voor de werking van de OK Cockpit. In de Cockpit staat een kamer centraal. Vanuit de kamer wordt de rest van de data opgehaald. Omdat dit een belangrijk deel van de database is, heb ik het onderstaande ERD gemodelleerd.



In de rest van het document zal het bovenstaande ERD centraal staan en zal daar verder op gemodelleerd worden.

4.1.7 Evaluatie

In deze sprint was het opzet van het project en het vaststellen van de requirements de hoofdzaak. Het maken van het plan van aanpak ging voorspoedig. Het vaststellen van de requirements ging niet voorspoedig. De opdrachtgever en het ziekenhuis zijn er nog niet helemaal uit wat zij willen. Dit zorgt voor nog veel vragen die onbeantwoord blijven. In de volgende sprint ga ik dieper in op bepaalde requirements. Ik zal dan ook meer informatie moeten vergaren in de gesprekken. Dit ga ik doen door van tevoren zelf na te denken over de mogelijkheden en voorstellen voor doen indien de opdrachtgever en het ziekenhuis twijfelt over bepaalde zaken. Per sprint zal ik voor de taken de eisen bepalen, zodat het opgeleverde resultaat het beste past bij de wens van de opdrachtgever en het ziekenhuis.

4.2 Sprint 1 basis cockpit

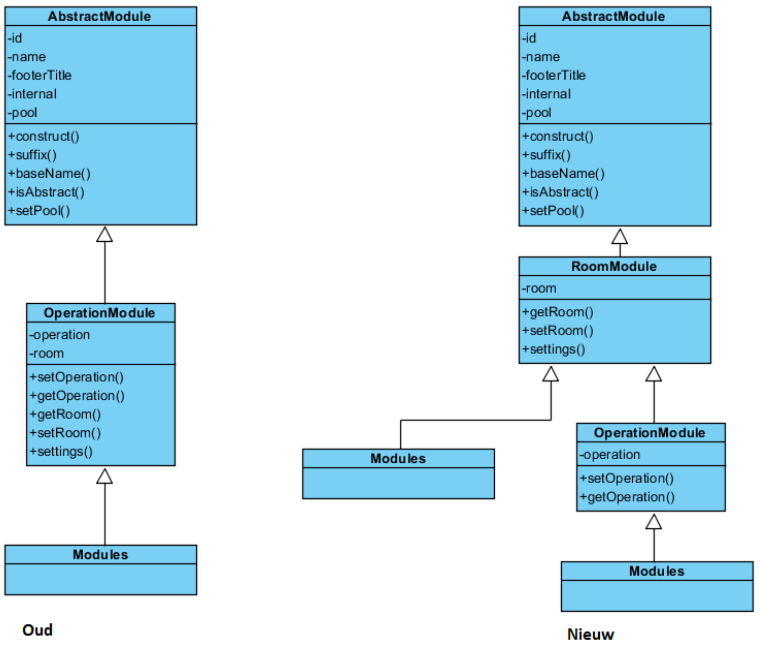
4.2.1 Taken

Bij de start van sprint één ben ik samen met de opdrachtgever in gesprek gegaan om de taken van deze sprint vast te stellen. Omdat de software eerst aangepast moest worden voor er uitbreidingen gemaakt kunnen worden was de keuze gemakkelijk te maken. De onderstaande niet-functionele software requirement is hetgene wat uitgevoerd zal worden in deze sprint. In deze sprint zijn niet meer taken ingepland, omdat dit de eerste keer was dat ik met de OK Cockpit code en met de PHP taal ga werken.

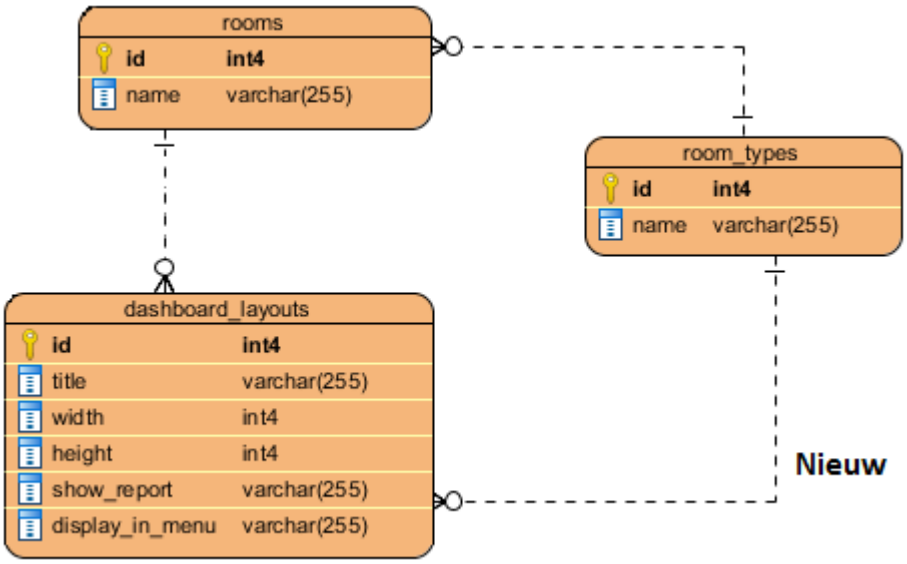
Requirement ID	Requirement Uitleg
NFSR2	De Cockpit moet real time data kunnen tonen

4.2.2 Ontwerpen

De software gebruikte voor elke module een operatieparameter. Voor de CSA Cockpit wordt er geen gebruik gemaakt van de operatieparameter. Daarom heb ik uit de Operation Module de kamerparameter gehaald en deze los gezet in Room Module. De Operation Module overerft nu de Room parameter vanuit de Room Module. Modules die geen gebruik maken van de operatieparameter kunnen nu Room Module overerven. Voor de modules die wel de operatieparameter nodig hebben kunnen Operation Module overerven. Met deze wijziging blijven de oude modules werken en kunnen er ook nieuwe modules gemaakt worden die geen operatieparameter bezitten. Dit is één van de plekken in de software waar deze wijziging door gevoerd moest worden. Ik heb onderstaande oude en nieuwe klassendiagram gemaakt om een duidelijk beeld te scheppen van deze wijziging:



De tweede aanpassing was dat voor elke gemaakte layout aangeven kon worden bij welke type kamer deze hoorde. Voorheen was het zo dat als er een layout gemaakt werd deze voor elke kamer bestond. Dat was goed omdat er alleen met OK kamers gewerkt werd. Maar omdat er nu ook ander soort kamers zijn, was dit een noodzakelijke aanpassing om het aantal layouts flink te beperken. De database aanpassing is hieronder in het plaatje te zien.

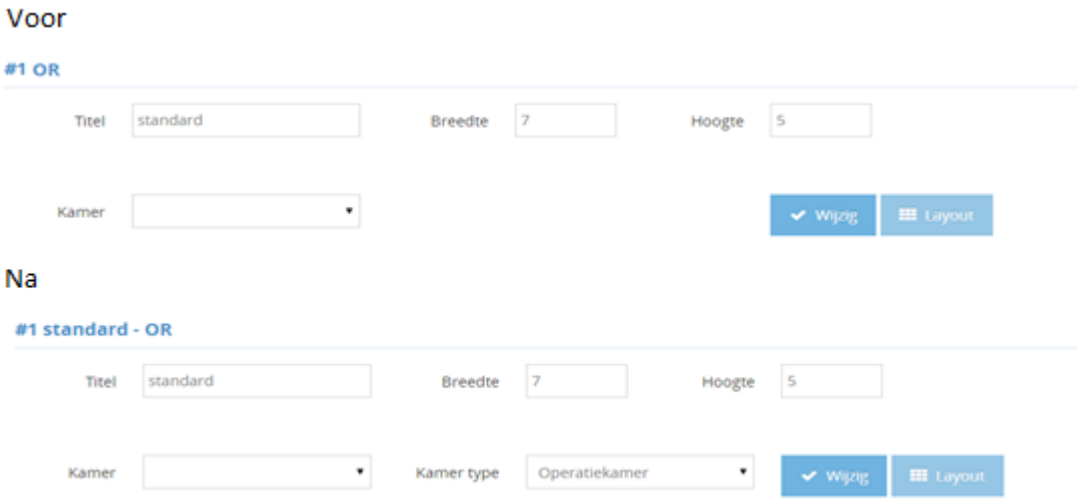


4.2.3 Bouwen

De twee wijzigingen die in de ontwerp fase naar voren zijn gekomen zijn geprogrammeerd. De room type aanpassing in de database zorgde ervoor dat wanneer de kamers opgehaald worden er ook gekeken moest worden welk type kamer deze is. Hieronder is een stukje van deze code te zien. Het rode deel is de oude code en het groene deel is de nieuwe code.

```
19  19      public function chooseRoom($location = null) {
20      -       $query = DB::table('rooms')->join('units', 'rooms.unit_id', '=', 'units.id')
21      -                   ->select('units.name AS unit', 'rooms.id', 'rooms.name')->where('active', true)->orderBy('rooms.id');
        20  +       $query = DB::table('rooms')
        21  +                   ->join('units', 'rooms.unit_id', '=', 'units.id')
        22  +                   ->select('units.name AS unit', 'rooms.id', 'rooms.name', 'rooms.type_id')
        23  +                   ->where('active', true)
        24  +                   ->orderBy('rooms.id');
```

In de layout van het layout opbouwscherm is eveneens deze wijziging aangebracht, zodat bij het maken van een layout aangeven kan worden bij welke type kamer deze hoort. Dit is te zien in de onderstaande afbeelding van voor en na deze aanpassing.



4.2.4 Testen

Nadat de aanpassingen in deze sprint gemaakt waren ben ik begonnen met het testen of de bestaande functionaliteiten nog werkten. Dit heb ik gedaan door test cases op te stellen. Deze test cases zijn gebaseerd op de eisen van de hoofdprogrammeur Sylvester Oosterbos. De uitgevoerde test cases zijn Kamer bekijken en Type toewijzen. Hieronder staan de logische test case en fysieke test van type toewijzen.

LOGISCHE CASE type toewijzen	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘Layouts’	Systeem toont layouts pagina.
Medewerker selecteert het drop down menu kamer type en selecteert ‘operatiekamer’.	Systeem toon bij type ‘operatiekamer’
Medewerker klikt op de knop ‘opslaan’	Systeem slaat de layout wijziging op.

FYSIEKE TEST type toewijzen			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop ‘Layouts’	Systeem toont layouts pagina.	√	
Medewerker selecteert het drop down menu kamer type en selecteert ‘operatiekamer’.	Systeem toon bij type ‘operatiekamer’	X	Toonden geen types
Medewerker klikt op de knop ‘opslaan’	Systeem slaat de layout wijziging op.	√	

Uit de test bleek dat de kamertype drop down menu niet de juiste types toonden. Deze bug is na de test sessie opgelost.

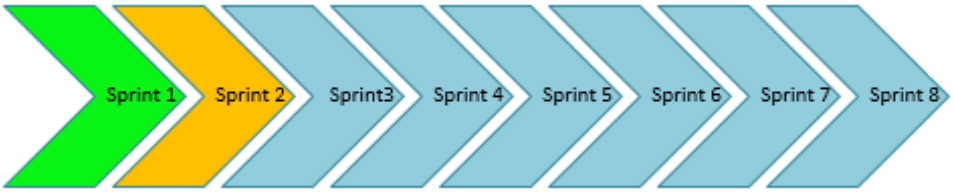
4.2.5 Evaluatie

In deze sprint heb ik deels leren werken met de Ok Cockpit software. De Ok Cockpit software is vergeleken met de programma's waar ik aan gewerkt heb een groot programma. Daarom zal ik er nog wel een tijd mee bezig zijn om de volledige structuur en werking te kunnen begrijpen. Dit komt mede omdat er weinig tot geen documentatie binnen het bedrijf aanwezig was. De door mij gemaakte documentatie zal ik aan het eind van mijn stage aanleveren aan NewCompliance. Dientengevolge zal in de toekomst een volgende stagiair wel documentatie hebben om verder te kunnen werken. Ook ben ik bezig geweest mijzelf PHP aan te leren. Dit heb ik gedaan door het gebruiken van een cursus boek(Kassenaar, 2004). Dit ging redelijk voorspoedig.

Samen met Sylvester Oosterbos hebben we wekelijkse de gemaakt code doorgelopen. Hieruit bleek dat ik geen gebruik maakte de PHP standaarden in code format. Deze format is de PSR standaard (php-fig, n.d.). Ook kwam naar voren dat ik gebruik maakten van Nederlandse variabelen en methodenamen waren soms ook verkeerd gespeld. Binnen het bedrijf wordt Engels geprogrammeerd. We hebben toen afgesproken dat ik de code zou aanpassen naar Engels. Ook zou ik gaan opletten dat ik automatische alles in het Engels zou gaan zetten. De reden was dat ik op school en huis altijd met Nederlandse variabelen en methodenamen heb gewerkt.

De OK Cockpit software is aanpasbaar gemaakt en er kunnen nu CSA uitbreidingen gemaakt worden. Dit zal dan ook in de komende sprints gaan gebeuren.

4.3 Sprint 2 Database wijzigingen



4.3.1 Taken

Aan het begin van deze sprint heb ik met de opdrachtgever besproken welke werkzaamheden er uitgevoerd moesten gaan worden in deze sprint. In dit gesprek kwam naar voren dat we een afspraak zouden gaan hebben met de eigenaren van de bronsystemen van het ziekenhuis. Ik heb toen naar voren gebracht dat het maken van een ERD ons duidelijk zou maken welke data we minimaal nodig hebben van die bronsystemen. We hebben toen gekeken naar welke taken uitgevoerd moeten worden voordat er begonnen kan worden met het maken van de CSA uitbreiding requirements. Dit waren Laravel mappings en een Mock-up Dashboard van de schermen die in de CSA geplaatst zullen worden. Hierdoor kon het ziekenhuis een beter beeld krijgen van hoe het er uit zal komen te zien. Hierdoor ontstonden nieuwe ideeën of aanpassingen. De activiteiten voor deze sprint zijn:

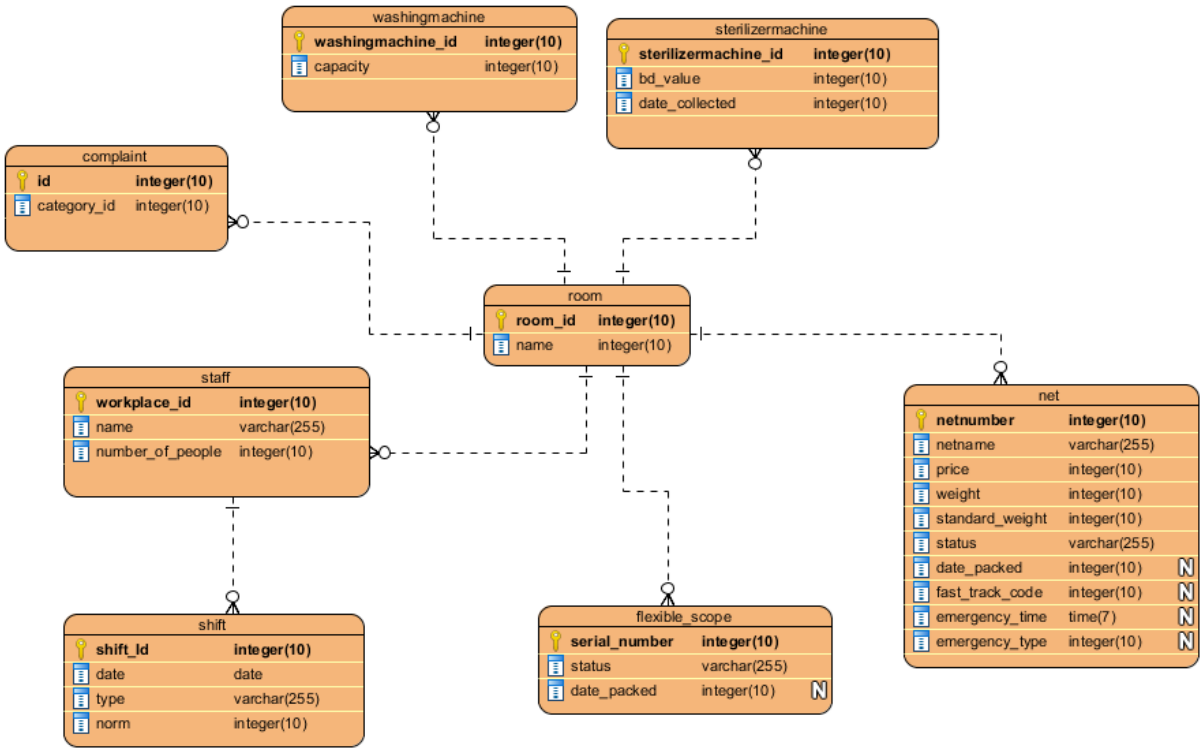
- Larvel mappings maken.
- Verwerken van requirements naar ERD.
- Verwerken van requirements en Trello bord naar mock-up van de Dashboards

4.3.2 Ontwerpen

In de ontwerp fase ben ik begonnen met het maken van het ERD. Dit diagram is opgebouwd uit de opgestelde requirements en een interview met de data leveranciers T-doc en 3M. Deze twee bedrijven zijn verantwoordelijk voor het opslaan van alle ziekenhuis data van het EMC. NewCompliance heeft maar een klein deel hiervan nodig. Daarom zijn we in gesprek gegaan om te bepalen welke data wij aangeleverd moeten krijgen en welke structuur deze data zou moeten hebben.

Er kwam al snel een probleem naar voren met de zogenoemde ‘netten’ (net is een pakket van OK gereedschappen). Een net kan namelijk uit twee soorten bestaan: een fast track net of een niet fast track net. De vraag was hoe ik dit het beste in de database kan gaan opslaan. Er zijn twee keuzes naar voren gekomen. Keuze één is een specialisatie maken van netten of het een fast track net is of niet. Dit zou uitkomen op 3 table’s. De tweede keuze was om de net table een fast track code column te geven en deze alleen invullen als er een fast track code is. Het voordeel van keuze één is dat er geen lege velden in de database staan. Een nadeel is wel dat netten van table kunnen veranderen omdat één net een fast track code kan krijgen of weggehaald kan worden. Bij keuze 2 kan door het invullen of leeg laten van de fast track code aangeven worden wat voor een net het is. Omdat bij keuze 1 meer data modulatie nodig is dan bij keuze 2 is voor keuze 2 gekozen. Dit is vervolgens ook met het stagebedrijf besproken. Het stagebedrijf vond het een terechte keuze. Als een fast track code ingevoerd wordt, worden ook de emergency_time en type ingevoerd vanuit het bronsysteem.

Met deze informatie is er het volgende ERD diagram gemaakt:



In dit ERD staan de kernvariabele van de CSA afdeling waar de Cockpit mee gaat werken. Bij packed_date is null toegestaan omdat een net of flexibele scope niet ingepakt hoeft te zijn.

Om de klant resultaat te kunnen laten zien zijn er een mock-ups gemaakt van alle Dashboards die gewenst waren vanuit het ziekenhuis. De onderstaande mock-up toont een CSA scherm met een voorbeeld layout.

4.3.3 Bouwen

In deze sprint is er nog niet veel geprogrammeerd. Deze bouwfase is gebruikt om mappings te maken in eloquent van laravel, kijkend naar het gemaakte ERD. Wanneer de mappings klaar zijn kan begonnen worden met het maken van de modules voor de dashboards. Hieronder is één van de mappings te zien.

```
37  class Room extends Model {
38
39
40      public function flexibleScope() {
41          return $this->hasMany('NewCompliance\Models\FlexibleScope');
42      }
43
44      public function net() {
45          return $this->hasMany('NewCompliance\Models\Net');
46      }
47
48      public function sterilizerMachine() {
49          return $this->hasMany('NewCompliance\Models\SterilizerMachine');
50      }
51
52      public function washingMachine() {
53          return $this->hasMany('NewCompliance\Models\WashingMachine');
54      }
55
56      public function complaint() {
57          return $this->hasMany('NewCompliance\Models\RoomComplaint');
58      }
59
60      public function staff() {
61          return $this->hasMany('NewCompliance\Models\Staff');
62      }
63  }
```

4.3.4 Testen

Voor het project was het noodzakelijk dat alles goed getest is, omdat het een ziekenhuissysteem betreft. Binnen NewCompliance worden er normaliter geen testen uitgevoerd. NewCompliance voert wel samen met het ziekenhuis een acceptatietest uit. Omdat dit niet de beste manier van werken is en je liever een bug vrij applicatie oplevert, heb ik voorgesteld om wel te testen tijdens de ontwikkeling van de MED Cockpit.

Voor het testen hebben ik gekozen om de plugin PHPUnit te gebruiken. Hierdoor kunnen gemakkelijk test cases worden uitgevoerd. Voor deze sprint is er gekozen om alleen CRUD testen uit te voeren. Deze testen doe je om de werking van de database koppeling te testen. Door het uitvoeren van een Create, Read, Update en Delete statements. Nadat de testen gemaakt en uitgevoerd waren bleek dat de koppeling niet altijd werkte.

4.3.5 Evaluatie

In deze sprint waren er gesprekken met het ziekenhuis en de data leveranciers. Tijdens deze gesprekken heb ik veel belangrijke informatie gekregen. Maar omdat het veel informatie betrof en ik tijdens de gesprekken niet alles hebben kunnen notuleren is een deel van deze informatie mij ontschoten. Vervolgens heb ik dit per mail later opnieuw gevraagd. Ik heb daarom besloten om tijdens de gesprekken uitgebreider te gaan notuleren zodat dit niet opnieuw kan gebeuren.

Uit het testen bleek dat de gemaakte mappings en keuzes in het ERD niet werkte samen met Laravel, omdat Laravel standaarden heeft voor database class namen en primary key namen. Primary key namen zijn namelijk altijd ‘id’ of dit moet anders aangeven worden in de mappings. De namen van de classes mogen niet meervoud zijn en moeten in lowercase geschreven zijn. Ook moet de mapping naam het meervoud zijn van de table naam. Nadat ik hier achter kwam heb ik de wijzigingen doorgevoerd in het EER en in de mappings. Naderhand heb ik de testen nog een keer uitgevoerd. Toen was de test wel gelukt.

4.4 Sprint 3 modules

4.4.1 Taken

Sprint 3 ben ik begonnen door in gesprek te gaan met het Erasmus MC en de opdrachtgever. In dit gesprek heb ik de mock-ups laten zien die in de vorige sprint gemaakt waren. Vervolgens zijn we naar de requirements gaan kijken. Door het zien van de mock-ups en interne overleggen in het ziekenhuis konden we de requirements specifieker gaan formuleren. Na deze sessie heb ik de requirements aangepast. We hebben in dit gesprek ook besproken wat er in deze sprint gemaakt zal worden. Het ziekenhuis gaf aan dat zij graag werkende dashboards wilden zien. De opdrachtgever heeft toen het ontwikkelen van de modules de hoogste prioriteit gegeven.

Om de modules te gaan maken hebben we het bestaande trello bord waar alle wensen van het ziekenhuis op staan uitgebreid. De wensen en eisen van het ziekenhuis zijn specifieker opgeschreven. Vervolgens hebben we deze wensen en eisen omgezet in modules. Het trello bord is te vinden in bijlage G. Op dit bord is te zien per kamer van de CSA welke modules er gemaakt moeten worden en wat deze modules moeten laten zien. In het bord is onderscheid gemaakt tussen modules die een algoritme hebben of modules die geen algoritme hebben en of bij bepaalde waardes de kleur van de module moet veranderen.

Aan deze kenmerken heb ik vervolgens een tijdsindicatie gegeven. Voor een module heb ik als basis 6 uur genomen. Als er een algoritme gemaakt zou moeten worden heb ik hier 4 uur voor gerekend. Als er een kleur verandering mogelijke moest zijn bij een module heb ik hier 2 uur voor gerekend. Deze tijden zijn in samenspraak met NewCompliance afgesproken. De onderstaande tabel staan

Module	Tijd
Capaciteit wasmachine t.o.v volume in aantocht	12 uur
Netten in aantocht	6 uur
Netten in reparatie	6 uur
Netten voor uitgifte	6 uur
Netten uitgeleend	6 uur
BD-waarde sterilisatoren	12 uur
Verlopen Netten	12 uur
Verlopen flexible scopen	12 uur
Aantal flexible scopen	6 uur
Aantal netten	6 uur
Netten met een afwijkend gewicht	10 uur
Fast track netten	12 uur

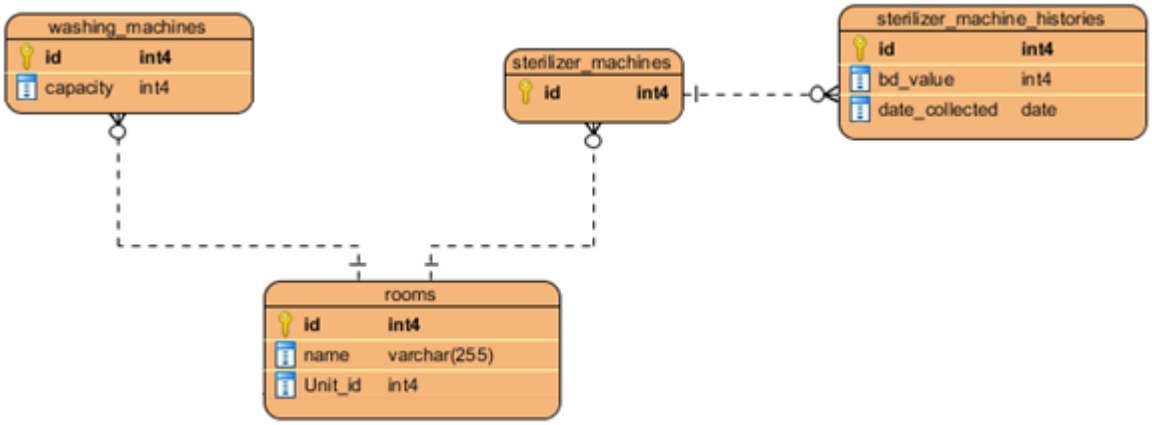
deze modules met hun tijdsindicatie:

Nu ik een tijdsindicatie had voor de modules ben ik met de opdrachtgever in gesprek gegaan en hebben we besloten dat we in deze sprint de eerste 6 modules van deze lijst zouden gaan maken en testen. De andere modules worden in de volgende sprint uitgevoerd.

4.4.2 Ontwerpen

In deze sprint bleek dat de aanpassing aan de overerving die ik gemaakt heb in sprint 1 goed van pas kwamen. Ik hoefde namelijk in deze sprint geen aanpassingen te doen aan de applicatie structuur omdat deze al gedaan was.

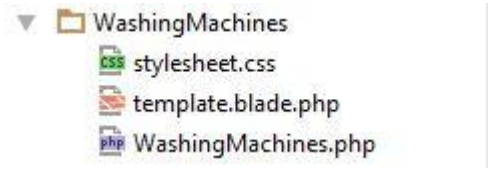
Wat wel bleek was dat er in het opgestelde ERD een deel miste over de machines die op de CSA afdeling staan. Dit zijn de wasmachines en de sterilisatoren. Ik ben toen in gesprek gegaan met het ziekenhuis welke informatie zij op het scherm wilde zien. Uit dit gesprek is de volgende toevoeging aan het ERD gekomen:



In dit ERD is te zien dat een kamer wasmachines en sterilisatoren kan bevatten. Bij de wasmachines was het gewenst om het laadvermogen op te kunnen slaan. Een wens was om de geschiedenis van een sterilisator op te slaan zodat deze later mogelijk teruggekeken kan worden. Een probleem was dat je in de sterilisator table niet meerdere keren dezelfde sterilisator met verschillende data op wil slaan. Ook zou het mogelijk moeten zijn om van een sterilisator meer gegevens op te kunnen slaan Daarom heb ik er voor gekozen om de geschiedenis in een aparte table op te slaan.

4.4.3 Bouwen

In de bouw fase ben ik begonnen met het maken van de modules. Een module heeft logica in PHP, een stylesheet in CSS en een View in een Laravel blade. Van de module Capaciteit wasmachine t.o.v volume in aantocht is hieronder de structuur te zien:


WashingMachines
stylesheet.css
template.blade.php
WashingMachines.php

Tijdens het maken van de modules kwam ik de volgende uitdaging tegen. De wens van de klant was dat er voor elke wasmachine een eigen module op het bord getoond wordt. Dus als een kamer vier wasmachines heeft dat er vier modules getoond worden met ieder informatie over 1 van de wasmachines.

De oplossing die ik hier heb gebruikt is het gebruiken van een Variance optie. Deze optie vertelt de controller dat hij deze module een bepaald aantal keer moet opbouwen. Door het gebruik van een associatieve array kan bij het aantal wasmachines ook het id van de machine meegegeven worden aan de controller. Zo weet de controller welke wasmachine hij moet tonen in welke module. Hieronder wordt deze code getoond:

```
public  function varianceOptions() {
    $options = [];
    $id = 0;
    $washingMachines = WashingMachines::whereRoomId($this->room)->all();
    foreach ($washingMachines as $washingMachine) {
        $id++;
        $options[$id] = $washingMachine->id;
    }

    return $options;
}
```

Een andere keuze zou zijn om meerdere modules te maken. Één voor elke wasmachine. Deze keuze zou betekenen dat er veel dubbele codes zouden komen. Ook maak het de code slecht aanpasbaar. De gekozen oplossing zorgt ervoor dat elke module meerdere keren hergebruikt kan worden.

4.4.4 Testen

Voor het testen van deze sprint heb ik gebruik gemaakt van Unittesten. Voor elke module heb ik een unit test gemaakt. Deze Unittests testen of een vooraf bepaalde invoer een vooraf bepaald resultaat oplevert en geeft dan een resultaat. Unittesten kunnen altijd uitgevoerd worden, zodat bij elke wijziging in het systeem gecontroleerd kan worden of alle functionaliteiten nog werken. Hieronder toon ik twee van de gemaakte unittesten:

```
public function testCountNets() {
    $nets[] = 1;
    $nets[] = 2;
    $count = (new NetsInfo(null, null))->countNets($nets);
    $this->assertEquals(2, $count);
}

public function checkpass() {
    $totalWachingMachineValue = 100;
    $totalNetValue = 50;
    $nets[] = 2;
    $pass = (new WashingMachines(null, null))->calculate($totalWachingMachineValue, $totalNetValue);
    $this->assertEquals('true', $pass);
}
```

In deze unittesten wordt gecontroleerd of countNets die het aantal netten in een array telt werkt. In de tweede test wordt gekeken of totale value van een wasmachine groter is dan het totaal aantal netten. Als dit waar is moet er ‘true’ uitkomen.

4.4.5 Evaluatie

De eerste fase van de sprint verliep trager dan verwacht. Het duidelijk krijgen van de requirements en deze dan vervolgens indelen kostte veel tijd. Ik begon dan ook later aan de ontwerp en bouw fase. Om deze tijd goed te maken ben ik thuis extra gaan werken. Zo heb ik de achterstand weg gewerkt om op tijd aan het testen te kunnen beginnen. Voor de volgende sprints moet ik zorgen dat de requirements van die sprint sneller op papier komen. Dit kan ik doen door zelf voorstellen te doen en na het gesprek met de klant de requirements tijdelijk vast te zetten tot het volgende overleg, zodat ik door kan werken met de voorlopige requirements. Daarentegen ging het maken van de modules snel en gemakkelijk omdat ik van tevoren mij bekent heb gemaakt met de code en laravel. Wel kwam in de code review naar voren dat er weer Nederlandse variabele namen zijn gebruikt. Dit was in de vorige sprints ook al zo en we hebben afgesproken dat ik hier in het vervolg beter op ga letten.

4.5 Sprint 4 Modules deel 2

4.5.1 Taken

In deze sprint zullen de modules aan bod komen die niet zijn uitgevoerd in de vorige sprint. Dit zijn de onderstaande modules:

Module	Tijd
Verlopen Netten	12 uur
Verlopen flexible scopen	12 uur
Aantal flexible scopen	6 uur
Aantal netten	6 uur
Netten met een afwijkend gewicht	10 uur
Fast track netten	12 uur

In het Sprint overleg met het ziekenhuis en de opdrachtgever kwam een nieuwe wens naar voren. De wens was dat bij de module BD-waarde sterilisatoren uit de vorige sprint een grafiek te zien zou zijn. Deze grafiek zou de huidige waarde laten zien en de geschiedenis hiervan. Ook wilden ze dat de module zijn kleur naar rood zou moeten veranderen als er niet aan bepaalde eisen voldeed. Deze eisen waren dat de waarde niet 0 of lager mag zijn. Ook wilde het ziekenhuis dat de module rood kleurt als de waarde meer dan 3 dagen achter elkaar stijgt of daalt of gelijk blijft. Deze waarde geeft namelijk een indicatie van het functioneren van de sterilisator.

Deze wens zal ook in deze sprint uitgevoerd worden. In de vorige sprint bleek dat de tijden die gegeven waren aan de modules te lang waren. De extra tijd zou genoeg ruimte moeten geven om deze extra functies toe te voegen in deze sprint.

In deze sprint zal de ontwerp fase overgeslagen worden omdat er geen wijzigingen aan het ontwerp deze sprint zijn uitgevoerd.

4.5.2 Bouwen

In de bouw fase ben ik begonnen met het bouwen van de modules. Vervolgens ben ik aan de extra wens van het ziekenhuis begonnen. Dit bleek uitdagender te zijn dan ik in eerste instantie had gedacht omdat namelijk in de huidige software nog geen grafieken in het dashboard werden gebruikt. Wel werd plugin highcharts gebruikt om een grafiek te maken. Deze plug-in tekent van een aangeleverde dataset een grafiek. Ik heb ook voor deze grafiek deze plug-in gebruikt. Vervolgens kwam ik bij het probleem dat een Module alleen één waarde kan tonen. Daarom heb ik er toen voor gekozen om een bestaande functionaliteit te gebruiken namelijk een popup scherm. Als er op een module geklikt wordt er een pop-up scherm getoond als deze ingesteld is. Als er op de module geklikt wordt zal een pop-up zoals hieronder weergegeven te zien zijn:

Ook was een wens dat de module rood zou kleuren als de waarden een bepaalde trend had. Hiervoor moest ik de data van de laatste 4 dagen ophalen. Vervolgens worden deze waardes met elkaar vergeleken met behulp van de onderstaande code

```
foreach ($input as $currentValue) {
    $trend = 'plat';
    if ($previousValue !== false) {
        if ($currentValue > $previousValue) {
            $trend = 'boven';
        } elseif ($currentValue < $previousValue) {
            $trend = 'beneden';
        }
        $trends[] = $trend;
    }

    $previousValue = $currentValue;
}

return $trends;
```

Hierna wordt gekeken of de array trends 3 keer dezelfde waarde bevat namelijk plat, boven of beneden. Als dit zo is kleurt de module rood en verschijnt er in beeld dat de trend van de grafiek in een van die richtingen loopt en dat er naar de machine gekeken moet worden.

4.5.3 Testen

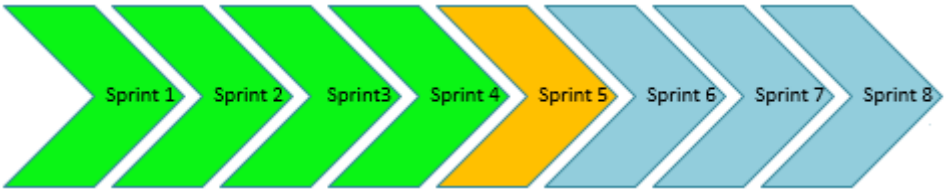
De modules in deze sprint zijn net als in de vorige sprint getest met unittesten. Vervolgens heb ik aan het eind van de sprint een performance test uitgevoerd. Dit heb ik gedaan omdat de response tijd van de applicatie steeds langer werd. In de test bleek dan ook dat de responsetijd naar 9 seconden was gegaan. NewCompliance heeft als regel dat zij de response tijd onder de 3 seconden willen houden.

4.5.4 Evaluatie

Deze sprint leek veel op de vorige sprint als je kijkt naar de werkzaamheden. Voor de grafiek heb ik gebruik gemaakt van Highcharts (Highcharts, n.d.). Highcharts is een plug-in die door NewCompliance al een tijdje wordt gebruikt. Ik kon mij snel inwerken in Highcharts door de vele demo's die beschikbaar waren op hun website.

Doordat we aan het begin van deze sprint de requirements vast hebben gezet kon ik snel beginnen aan de volgende fases. Dit heeft er voor gezorgd dat de taken van deze sprint ruim op tijd af waren. Ik heb toen een performance test uitgevoerd. Door het uitvoeren van deze test kwam het probleem naar voren dat de applicatie te traag is geworden volgens de NewCompliance standaard. Tot nu toe had ik nog niet nagedacht over de response tijd van het programma. In de volgende sprint zal ik wat moeten gaan doen om de performance te verbeteren.

4.6 Sprint 5 gevaarlijke situatie en klachten formulier



4.6.1 Taken

Aan het begin van sprint 5 heb ik aan de opdrachtgever en het Erasmus MC een Dashboard getoond met de modules van sprint 3 en 4. Uit dit gesprek bleek dat het ziekenhuis tevreden was met het resultaat en we door konden gaan met de volgende requirements. Besloten werd om de volgende twee use case slices uit te voeren, namelijk: Invoer klachten formulier en Invoer gevaarlijke situatie. We hebben toen deze requirements besproken en wat er precies ingevuld moet kunnen worden. Hieruit kwamen de volgende twee specifiekere requirements:

Requirement ID	Requirement Uitleg	Source
R1	Een CSA medewerker moet een gevaarlijke situatie formulier kunnen invullen in het systeem. Op het formulier moet automatische de datum ingevuld worden en de medewerker moet net nummer invullen of inscannen. Veder moet de afdeling, het voorwerp(mesje, naald en overig) en de medewerker zijn naam ingevuld worden.	Case
R2	Een OK medewerker moet klachtformulier kunnen invullen in het systeem. Hier moet de categorie(beschadiging papier, vies, vermissing instrument, verkeerd instrument, verkeerd gemonteerd en overig), net nummer en afdeling ingevuld worden.	Case

Nadat deze requirements aangepast waren heb ik deze aanpassingen doorgevoerd naar de use case slices. Hieuit kwam het volgende:

Invoer gevaarlijke situatie

Nummer	Use-case slice
1	Invoer gevaarlijke situatie
2	Invoer gevaarlijke situatie - connectie probleem

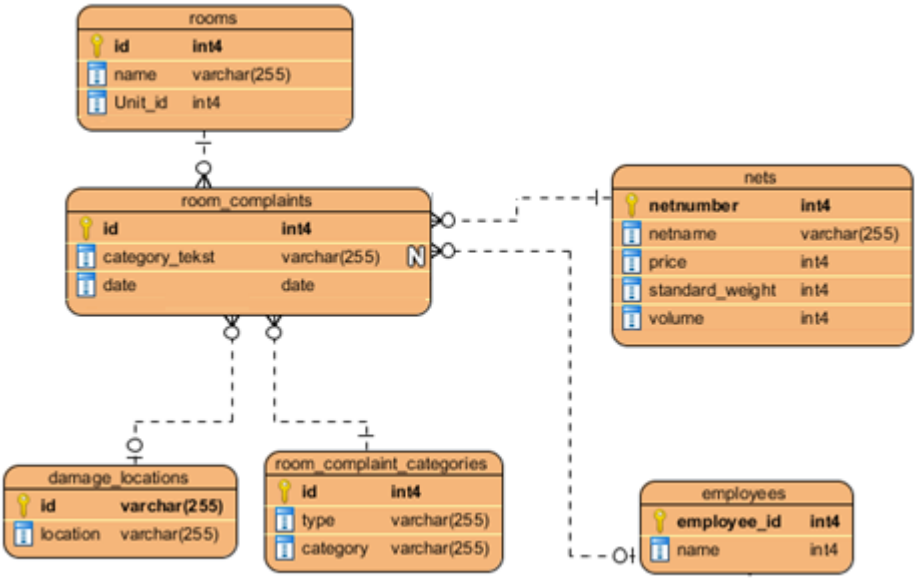
Invoer klachten OK

Nummer	Use-case slice
3	Invoer klachten OK 1
4	Invoer klachten OK 2
5	Invoer klachten OK - connectie probleem

Toen de use case slices gereed waren, had ik een goede indicatie van wat er gemaakt moest worden en kon ik hieraan beginnen. Ook heb ik met de opdrachtgever afgesproken dat ik in deze sprint de response tijd van de applicatie zou verbeteren.

4.6.2 Ontwerpen

Om de data van de twee invoer formulieren op te kunnen slaan heb ik database aanpassingen uitgevoerd. Van deze wijziging heb ik het onderstaande ERD gemaakt.

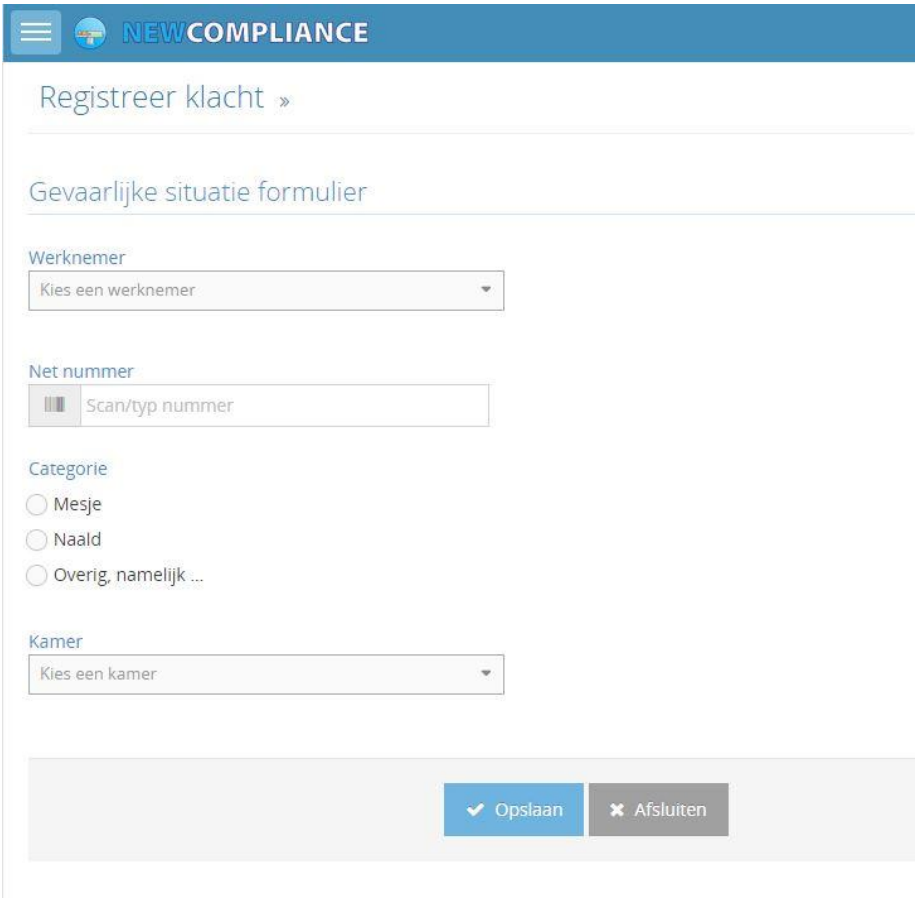


In eerste instantie was het idee om een specialisatie te maken van een invulformulier. Namelijk gevaarlijke situatie invulformulier en klachten invulformulier. Maar ik heb besloten dit niet te doen, omdat beide formulieren dezelfde data opslaan. Daarom heb ik er voor gekozen om beide formulieren in één table op te slaan genaamd room_complaint. Deze heeft dan een link met room_complaint_categories om het type klacht op te slaan. Zo heb ik voorkomen dat er meerdere keren dezelfde data in de database wordt opgeslagen.

In deze ontwerpfase heb ik ook nagedacht hoe ik de response tijd kan verlagen. De keuzes ik had waren meer hardware gebruiken wat een dure oplossing is of een wijziging in de software. Ik heb er toen voor gekozen om eerst naar de software te kijken of daarmee het probleem opgelost kan worden. Al snel kwam ik er achter dat tijdens het laden van het dashboard alle modules opgehaald worden. Vervolgens kijkt het systeem pas naar welke modules er op het scherm getoond moeten worden. Dit zorgt ervoor dat voor elke module die gemaakt wordt de laadtijd van de applicatie wordt vergroot. De oplossing die ik hiervoor bedacht heb, is kijken naar de type van de modules in combinatie met de layout type. Dit is de wijziging die ik in sprint 1 heb gemaakt en die nu weer van pas komt. Door alleen de modules te laden met het zelfde type als de layout zal de laadtijd dalen.

4.6.3 Bouwen

In de bouwfase ben ik begonnen met het maken van de invulformulieren. Een wens van het ziekenhuis was dat de velden gecontroleerd moeten worden op input. Bijvoorbeeld er moest gecontroleerd worden dat bij het invullen van het netnummer waar de klacht overgaat of deze in de database bestaat zodat er geen foutieve invoer kan worden gemaakt. Om dit gemakkelijk te kunnen uitvoeren heb ik gekozen om gebruikt maken van de plugin Formvalidation (Formvalidation, n.d.). Met deze plug-in kan in HTML gemakkelijk aangegeven worden dat een veld gecontroleerd moet worden. Dit kan zijn dat het veld niet leeg mag zijn of er kan in Javascript een check geschreven worden voor het veld. Een voorbeeld van het invoerveld is hieronder weergeven.



Vervolgens heb ik de bedachte oplossing voor het laadprobleem uitgevoerd.

4.6.4 Testen

In de test fase heb ik opnieuw de performance test uitgevoerd. Hieruit bleek dat de laadtijd naar 4 seconden is gedaald. Omdat de server op mijn eigen laptop draait is dit een accepteerbare laadtijd. Als de database straks op een server staat zal de laadtijd verbeteren, omdat een server geoptimaliseerd is om meer data te kunnen versturen. Voor het testen van de invoervelden heb ik de logische en fysieke testen uitgevoerd die ik bij het maken van de use case slices heb opgesteld.

4.6.5 Evaluatie

Aan het begin van deze sprint ben ik begonnen met het presenteren van de tot nu gemaakte software. Deze presentatie was voor een aantal nieuwe medewerkers van het Erasmus Ziekenhuis. De presentatie verliep niet goed omdat ik niet had verteld wat ik ging vertellen en wanneer er vragen gesteld mochten worden. Daardoor verliep deze presentatie chaotische en zaten er een aantal medewerkers met nog meer vragen dan voor de presentatie. Om dit volgende keer niet weer te laten gebeuren heb ik afgesproken met mijn stagebegeleider om deze presentaties voortaan eerst te oefenen. Hiermee bedoel ik dat we intern bij NewCompliance een test presentatie doen. Verder liep de sprint voorspoedig en waren de taken aan het eind van de sprint uitgevoerd.

4.7 Sprint 6 Checklists

4.7.1 Taken

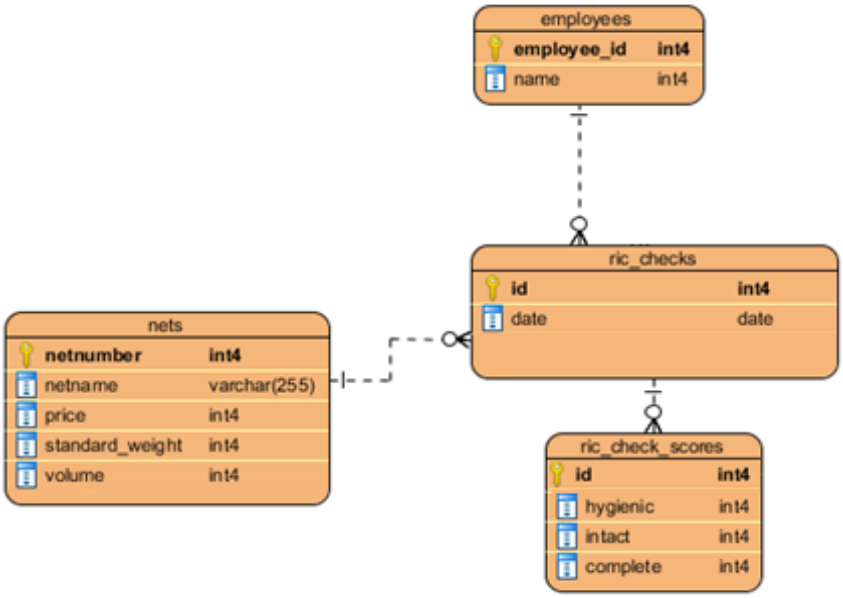
In sprint 6 heeft de opdrachtgever en het ziekenhuis de gemaakte invoervelden goedgekeurd. We hebben toen gekeken naar welke use case slices er nog uitgevoerd moesten worden. Dit waren de volgende use case slices:

Nummer	Use-case slice
6	Invoer daglijsten
7	Invoer RIC controle
8	Invoer RIC controle – Opnieuw
9	Invoer RIC controle – Reparatie
10	Daglijst wijzigen
12	Checklist apparatuur

Er is voor gekozen om nummer 7, 8, 9, 10 en 12 uit te voeren. Dit zijn de checklists die door de Dagoudste van de CSA afdeling ingevuld moeten kunnen worden. We hebben toen besproken wat er in die checklist gevraagd moet worden aan de gebruiker. Nadat ik alle informatie had ontvangen ben ik begonnen aan de ontwerp fase.

4.7.2 Ontwerpen

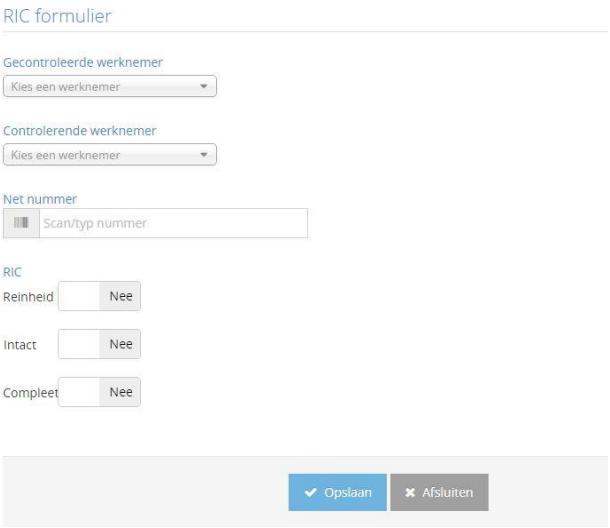
Voor het opslaan van de invoer van de RIC (Reinheid, Intact en Compleet) controle moest ik wijzigingen uitvoeren aan de database. Deze zijn in het onderstaande ERD te zien:



Voor de checklists van de apparatuur hoeft geen database wijziging gemaakt te worden, omdat er al een module bestaat voor het maken van een checklist voor apparatuur vragen.

4.7.3 Bouwen

In de bouwfase heb ik het onderstaande invulformulier gemaakt om de RIC check te kunnen uitvoeren



Voor het maken van de apparatuur checklist was het alleen nodig om de opgestelde vragen in de database te voegen. Hier is verder niks voor gebouwd.

4.7.4 Testen

Voor het testen van de invoer velden heb ik de logische en fysieke testen uitgevoerd die ik bij het maken van de use case slices heb opgesteld.

4.7.5 Evaluatie

In deze sprint heb ik veel contact gehad met het ziekenhuis, omdat zij geen keuze konden maken. Omdat er deze sprint niet veel aan ontwerpen, bouwen en testen hoefde te gebeuren heb ik meer tijd besteed om met de klant de mogelijke opties te bespreken.

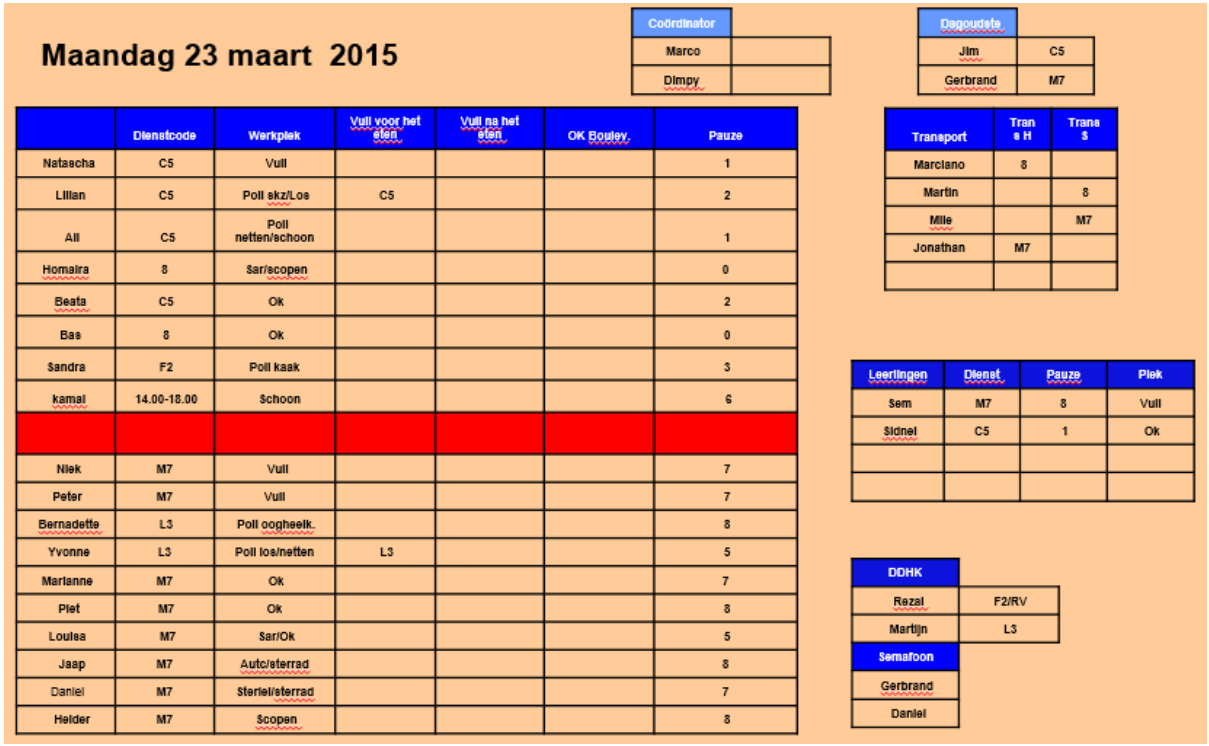
4.8 Sprint 7 Daglijsten

4.8.1 Taken

Gedurende deze sprint zullen de laatste use case slices uitgevoerd worden. Dit zijn:

Nummer	Use-case slice
6	Invoer daglijsten
10	Daglijst wijzigen

Op dit moment worden de daglijsten binnen de CSA afdeling gemaakt door middel van de onderstaande PowerPoint sheet.



Maandag 23 maart 2015

Coördinator	
Marco	
Dimpy	

Dagoudste	
Jim	C5
Gerbrand	M7

	Dienstcode	Werkplek	Vull voor het eten	Vull na het eten	OK Bouley.	Pauze
Natascha	C5	Vull				1
Lillian	C5	Poli skz/Los	C5			2
Ali	C5	Poli netten/schoon				1
Homaira	8	Sar/scopen				0
Beata	C5	Ok				2
Bas	8	Ok				0
Sandra	F2	Poli kaak				3
kamal	14.00-18.00	Schoon				6
Niek	M7	Vull				7
Peter	M7	Vull				7
Bernadette	L3	Poli oogheelk.				8
Yvonne	L3	Poli los/netten	L3			5
Marianne	M7	Ok				7
Piet	M7	Ok				8
Loulsa	M7	Sar/Ok				5
Jaap	M7	Autc/sterrad				8
Daniel	M7	Sterliel/sterrad				7
Helder	M7	Scopen				8

Transport	Tran s H	Trans $
Marciano	8	
Martin		8
Mlle		M7
Jonathan	M7	

Leerlingen	Dienst	Pauze	Plek
Sem	M7	8	Vull
Sidnel	C5	1	Ok

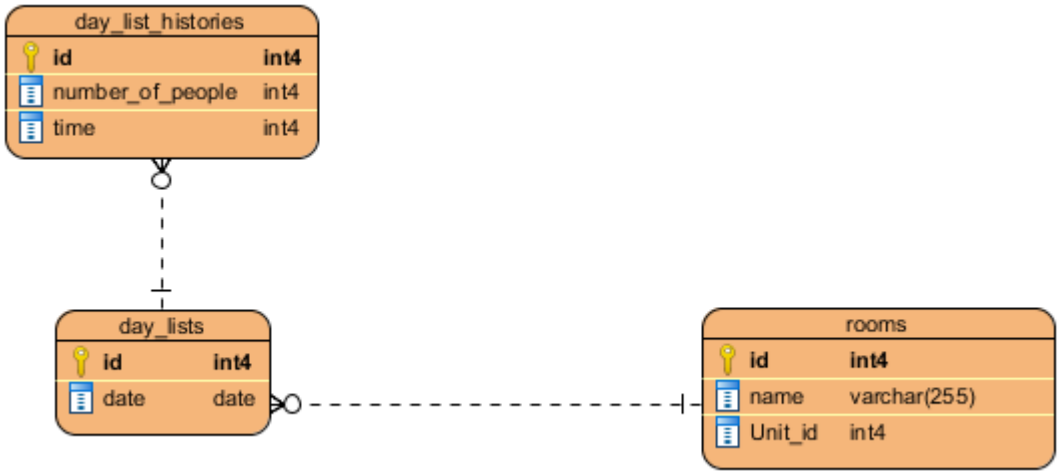
DDHK	
Rezai	F2/RV
Martijn	L3
Semafoon	
Gerbrand	
Daniel	

Door middel van de daglijsten wordt de werkplekverdeling gemaakt, hiervoor is dus een handmatige invoer nodig om dagelijks een verdeling te maken voor 28 man personeel over 8 verschillende locaties.

De informatie uit deze daglijsten wordt weer verder gebruikt voor de berekening van de bezetting op de verschillende locaties binnen de CSA afdeling. Omdat de werklast binnen deze afdelingen kan verschillen van uur tot uur is het belangrijk dat er gemakkelijk wijzigingen aangebracht kunnen worden aan deze invoer. De wens is om in een invoer scherm per kamer het aantal mensen aan te geven die daar op dat moment werken.

4.8.2 Ontwerpen

Bij het opslaan van een daglijst is het belangrijk dat de datum van die dag en hoeveel mensen er die dag op een bepaalde tijd in een kamer aan het werk waren wordt opgeslagen. Dit heb ik gedaan door een table te maken die de dag opslaat in combinatie met de kamer. Vervolgens staat in een andere table de history van die dag zodat er niet van een dag meerdere daglijsten worden opgeslagen. In de history table wordt elke wijziging van het aantal mensen op een kamer opgeslagen samen met de tijd van deze wijziging. Later kan er altijd worden teruggekeken hoeveel mensen er op een bepaald moment in een kamer aan het werk waren. Het onderstaande ERD toont deze uitbreiding:



4.8.3 Bouwen

In deze fase heb ik de bovenstaande uitbreiding gemaakt en het daglijst invulformulier. Vervolgens heb ik ook enkele wijzigingen gemaakt aan de CSS van de Applicatie. De CSS bepaald het letter type en vormgeving van de tekst.

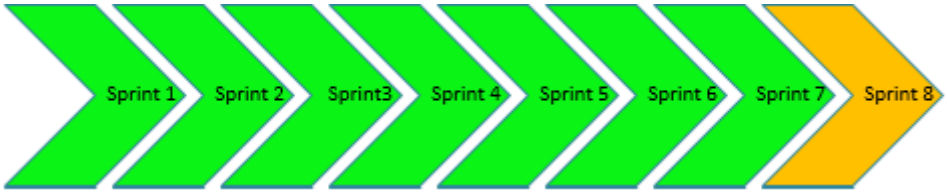
4.8.4 Testen

Voor het testen van de invoervelden heb ik de logische en fysieke testen uitgevoerd die ik bij het maken van de use case slices heb opgesteld.

4.8.5 Evaluatie

In deze sprint heb ik de laatste functionaliteit gemaakt die er voor dit project nodig was. Samen met Sylvester Oosterbos heb ik een volledige code review gedaan. Er is gekeken naar de kwaliteit van het totale werk en een laatste check gedaan om te controleren dat de gewenste functionaliteiten aanwezig zijn. Vervolgens hebben we alle unittesten uitgevoerd om te kijken of alle functionaliteiten ook werkten. Tijdens de test bleek inderdaad dat alle functionaliteiten werkten.

4.9 Sprint 8 afronding



Alle taken van de afstudeeropdracht zijn afgerond. Daarom ben ik het project gaan afronden door mijn werk op te leveren aan NewCompliance. NewCompliance vroeg mij of ik een eindpresentatie wilde geven aan het ziekenhuis. Dit heb ik gedaan. In deze presentatie zijn we door het project heen gelopen en heb ik alle functionaliteiten van de software getoond aan een aantal managers van het ziekenhuis. Omdat de vorige presentatie niet goed verliep heb ik een betere indeling van mijn presentatie gemaakt. De indeling ziet er als volgt uit:

- Wie ben ik
- Wat kom ik presenteren
- Project verloop
- applicatie
- afronding waar er vragen gesteld mochten worden.

Ik had van te voren aangeven dat de vragen aan het eind van de presentatie gesteld mochten worden. Zo kon ik eerst mijn verhaal doen zonder dat ik onderbroken werd. De presentatie heb ik eerst binnen NewCompliance gegeven om te kunnen oefenen en om tips te vragen.

Na het uitvoeren van deze presentatie en het opleveren van de documenten aan NewCompliance was de afstudeerstage bij NewCompliance afgerond.

Na het ontvangen van de documenten en software zal NewCompliance een gebruikers acceptatie test gaan uitvoeren. De test bestaat uit het gebruiken van de MED Cockpit door de medewerkers van de CSA gedurende een periode. Uit deze test zal blijken of de CSA werkzaamheden efficiënter kunnen worden uitgevoerd.

5 Evaluatie

In dit hoofdstuk evalueer ik het afstudeerproject. Dit doe ik door eerst het proces te evalueren en de leermomenten daarin aan te stippen. Hierna evalueer ik de geleverde producten en de leerprocessen die ik daarbij heb gehad. Daarna evalueer ik de opgeleverde producten.

5.1 Proces evaluatie

Het opzetten van dit project is goed verlopen. Door het hebben van meerdere gesprekken met de stakeholders kon ik een duidelijk plan van aanpak opstellen. De keuze van ontwikkelmethoden was Scrum. Dit bleek een goede keuze te zijn geweest omdat de requirements tot de laatste sprint nog zijn veranderd. De eerste fase van het project, waarbij aanpassing van de software van OK Cockpit naar MED Cockpit is gedaan, had ook uitgevoerd kunnen worden met RUP ontwikkelmethoden, omdat hier wel duidelijk vastgesteld kon worden wat er gedaan moest worden.

De requirements zijn tijdens het project constant veranderd. Door gebruik maken van een trello bord, waar alle eisen aan het systeem opstonden, konden er gemakkelijk aanpassingen gemaakt worden. Tijdens de gesprekken geeft het trello bord een goede houvast om naar te verwijzen. De aanpassingen aan het trello bord worden opgeslagen. Dus naderhand kan er gezien worden wat er veranderd is en dit kan dan doorgevoerd worden in de overige documenten. Voor volgende projecten zal ik zeker overwegen om zo'n trello bord weer toe te passen.

In de requirements fase heb ik de keuze gemaakt om de use cases op te delen in use case storys en vervolgens in use case slices. Deze keuze had ik gemaakt om van de use cases kleinere taken te maken, zodat deze gemakkelijker in sprints verdeeld konden worden. Dit bleek naderhand niet nodig te zijn omdat na verfijning van de requirements deze al goed in sprints ingedeeld konden worden. Deze techniek zal ik een volgend project dan ook niet meer gaan gebruiken.

Het ontwerpen en bouwen van de applicatie is op een gecontroleerde manier door mij uitgevoerd. Het ontwerpen heb ik gedaan aan de hand van de in de sprint opgestelde eisen. Ook het bouwen van de applicatie is soepel verlopen. Ook heeft code reviewing ervoor gezorgd dat de code van goede kwaliteit is. Voor een volgend project zou ik zeker weer gebruik maken van code reviewing om de kwaliteit van de code te waarborgen.

Voor het testen heb ik gebruik gemaakt van verzonnen data. Echter zou in een vervolgproject een beter alternatief zijn om echte werknemersgegevens te gebruiken. Door de gegevens van een databron van het ziekenhuis te gebruiken kan het mogelijk zijn dat de data fouten bevat, door bijvoorbeeld menselijke fouten. Op deze manier kan je dan beter testen of de applicatie blijft werken, zelfs onder onverwachte omstandigheden. Hierdoor wordt de applicatie robuuster.

Voor de opdracht stond 17 weken. Ik heb aan het begin van de stage ervoor gekozen om 20 weken te gebruiken. Dit zorgde ervoor dat alle wensen en eisen zijn uitgevoerd. Wel waren er aan het einde van de afstudeer stage niet genoeg taken om de volledige sprint te vullen. In het vervolg kan ik beter de extra tijd niet inplannen en pas gebruiken als deze nodig blijkt te zijn.

De leerpunten van deze van het afstudeer proces zijn als volgt:

- Het beter voorbereiden van presentaties en gesprekken.
- Extra tijd niet in te plannen en deze pas gebruiken indien deze nodig is.

5.2 Product evaluatie

Plan van aanpak

Het plan van aanpak is in de eerste weken van het project opgesteld. Ik ben zelf tevreden over het plan van aanpak. De gemaakte afspraken gaven een duidelijk beeld van het project. Het plan van aanpak is tijdens het project nog enkele keren aangepast. Verder heeft het plan van aanpak geholpen om de organisatie omtrent het project vanaf de start duidelijk te hebben. Het plan van aanpak was dus een goede overeenkomst tussen mij en de opdrachtgever.

Requirements rapport

Het requirementsrapport is een document waar ik tevreden over ben. De requirements zijn verkregen uit meerdere gesprekken en de opdracht omschrijving. Hierna zijn de requirements geprioriteerd aan de hand van de MoSCoW methode. Door voor elke sprint de requirements te bepalen ontstond er duidelijke afspraken over de te maken software. . Verder hebben de use case beschrijvingen mij ondersteund in het uitwerken van de requirements. Bovendien hebben de beschrijvingen als goede basis voor de logische en fysieke testcases kunnen bieden.

Functioneel ontwerp

Over het functioneel ontwerp ben ik tevreden. De schermontwerpen zijn gebaseerd op de wens van de klant in de huisstijl van NewCompliance. Met de schermontwerpen heb ik het ziekenhuis laten zien hoe de applicatie eruit zal komen te zien.

Techniche ontwerp

Ik ben ook tevreden over het technisch ontwerp. De gemaakte diagrammen hebben mij geholpen tijdens het bouwen van de applicatie. Hierdoor had ik een duidelijke structuur tijdens mijn project.

MED Cockpit

Ik ben heel tevreden over de MED Cockpit. De kwaliteit van de code is door herhaaldelijke code reviews van goede kwaliteit. Door het maken van goede ontwerpen heb ik een object georiënteerde applicatie kunnen programmeren. Hierdoor is de code duidelijk leesbaar en onderhoudbaar geworden. Alle functionaliteiten zijn in het programma uitgewerkt en alles was naar wens van de stakeholders.

Testrapport

Over het testrapport ben ik redelijk tevreden. Door het uitvoeren van unittesten is de kwaliteit van de code gewaarborgd. Verder heb ik met de use cases de requirements kunnen testen door hier logische en fysiek testen van te maken. Het uitvoeren van een gebruikersacceptatietest zou goed in dit rapport passen. Maar deze acceptatie test zal pas na mijn afstuderen uitgevoerd worden door NewCompliance.

6 Beroepstaken

6.1 Competentie 1.4: Uitvoeren analyse door definitie van requirements

Tijdens dit project heb ik requirements opgesteld. Er zijn tijdens het project op verschillende moment requirements naar voren gekomen. Uit het bestaande Trello bord kon ik een eerste versie van de requirements verkrijgen. Toen heb ik samen met de opdrachtgever en het ziekenhuis deze requirements besproken om een beter en dieper beeld van de requirements te krijgen.

Verder heb ik een aantal functionele requirements opgesteld. Hierbij heb ik scheiding gemaakt in de requirements op functionele- en niet functionele requirements. Op basis van de requirements heb ik een use case diagram opgesteld.

Aan het begin van alle sprints heb ik samen met de stakeholders de requirements geprioriteerd. Dit heb ik gedaan door de MoSCoW methode toe te passen. Tevens heb ik van de belangrijkste requirements een use case beschrijving gemaakt.

Tijdens elke sprint heb ik de wijzigingen die werden gedaan aan de requirements doorgevoerd in het requirementsverslag

6.2 Competentie 3.2: Ontwerpen systeemdeel

Tijdens dit project heb ik de OK Cockpit omgebouwd naar een nieuwe Cockpit genaamd MED Cockpit. Tijdens het ontwerpen van de applicatie ben ik begonnen met het maken van mock-ups van de schermen. Dit om te kijken of het aan de wens van de klant overeen kwam.

Door middel van use case beschrijvingen heb ik de interactie tussen actor en systeem ontworpen. Hierbij worden de functionaliteiten van de applicatie schematisch in beeld gebracht.

Verder heb ik tijdens het ontwerpen een duidelijke scheiding tussen de componenten aangehouden. Zo heb ik de view losgekoppeld gehouden van de model door gebruik van controllers. Doordat ik de applicatie object georiënteerd ontworpen heb is de applicatie gemakkelijk aanpasbaar gemaakt.

6.3 Competentie 3.3: Bouwen applicatie

Tijdens elke sprint heb ik de gemaakte ontwerpen verwerkt in de software. Door object georiënteerd ontwerpen kon ik ook gemakkelijk programmeren. Ook zorgde dit ervoor dat in de toekomst er gemakkelijk extra modules aan de software toegevoegd kunnen worden. Door dit vanaf de eerste sprint toe te passen kon ik elke sprint zonder veel aanpassingen doen.

Ik heb de applicatie in PHP gemaakt met gebruik van java script, CCS en HTML. Ook heb ik gebruik gemaakt van frameworks en plug-ins. De belangrijkste hiervan waren Laravel en PHP unit.

De source code heb ik in GitHub gezet, zodat ik gebruik kon maken van revisiebeheer. Ook kon Sylvester Oosterbos op deze manier de code reviewen.

6.4 Competentie 3.5: Uitvoeren en rapporteren over het testproces

De functionaliteit van de Cockpit moest worden getest. Er wordt hierbij een logische testontwerp en een fysiek testontwerp gemaakt.

Tijdens het bouwen van de applicatie heb ik gebruik gemaakt van PHP Unit om unit tests te maken. Met deze tests heb ik de verwachte in en uitvoer van specifieke methoden in de applicatie getest. Op deze manier heb ik de kwaliteit van de code verantwoord. Daarnaast heeft de code review er ook voor gezorgd dat de kwaliteit van de code wordt gewaarborgd.

Verder heb ik een performance test gedaan. Hierbij heb ik de laadtijden getest van het dashboard.

7 Literatuurlijst

Kassenaar, P. (2004). *Basiscursus PHP 5.* Den Haag: Sdu Uitgever bv.

NewCompliance. (sd). Opgehaald van www.newcompliance.nl.

php-fig. (sd). *http://www.php-fig.org/*. Opgehaald van PHP Framework Interop Group.

Schaaf, R. (2012, augustus). *use case slice based product backlog*. Opgehaald van Xebia: http://blog.xebia.com/2012/08/13/use-case-slice-based-product-backlog-an-example/

scrumalliance. (sd). Opgehaald van https://www.scrumalliance.org/why-scrum

Temnenco, V. (2006, November 15). *RUP*. Opgehaald van http://www.ibm.com/: http://www.ibm.com/developerworks/rational/library/nov06/temnenco/

Waterfall Model. (2015). Opgehaald van http://www.waterfall-model.com/

Westen, W. v. (2009). *Goed Geschreven.* Bussum: Uitgeverij coutinho.

8 Bijlagen

8.1 Woordenlijst

Woord	Verklaring
PHP	Is een script taal
HTML	Is een opmaaktaal
CSS	Zorgt voor de vormgeving van elementen van een webpagina
Javascript	Is een script taal
Mappings	Elementen van verschillende databronnen, die aan elkaar zijn gekoppeld
Timeboxing	Is het vooraf beperken van de hoeveelheid tijd die men aan een bepaalde activiteit wel en mag besteden.

Afkorting	Betekenis
CSA	Central Sterilisatie Afdeling
CRUD	Create, Read, Update, Delete
MoSCoW	Must have, Should have, Could have, Won't have
RUP	Rational Unified Process
MVC	Model View Controller
ORM	Object-relational mapping

8.2 Bijlagenlijst

Bijlage	Naam
A	Afstudeerplan
B	Plan van aanpak
C	Requirementsrapport
D	Functioneel ontwerp
E	Technisch ontwerp
F	Testrapport
G	Trello bord

Bijlage A: afstudeerplan

Afstudeerplan

Informatie afstudeerder en gastbedrijf (*structuur niet wijzigen*)

Afstudeerblok: 2015-1.2 (start uiterlijk 11 mei 2015)
Startdatum uitvoering afstudeeropdracht: (3 mei) nog niet zeker
Inleverdatum afstudeerdossier volgens jaarrooster: 5 oktober 2015

Studentnummer: 10015493
Achternaam: dhr (*) *weghalen niet van toepassing*
Voorletters: S P
Roepnaam: Shawn
Adres: Speenkruid 52
Postcode: 2643 KJ
Woonplaats: Pijnacker
Telefoonnummer:
Mobiel nummer: 0636114901
Privé emailadres: shawnsteinz@hotmail.com

Opleiding: Informatica
Locatie: Zoetermeer (*) *weghalen niet van toepassing*
Variant: voltijd

Naam studieloopbaanbegeleider: Arno Nederend
Naam begeleidend examinator: Nanny Jacobs
Naam tweede examinator: Arno Nederend

Naam bedrijf: NewCompliance
Afdeling bedrijf:
Bezoekadres bedrijf: Cobaltstraat 26
Postcode bezoekadres: 2718 RM
Postbusnummer:
Postcode postbusnummer:
Plaats: Zoetermeer
Telefoon bedrijf: 0797370000
Telefax bedrijf:
Internetsite bedrijf: http://newcompliance.nl/

Achternaam opdrachtgever: dhr Wiesman (*) *weghalen niet van toepassing*
Voorletters opdrachtgever: B
Titulatuur opdrachtgever:
Functie opdrachtgever: Ceo
Doorkiesnummer opdrachtgever:
Email opdrachtgever: b.wiesman@newcompliance.nl

Achternaam bedrijfsmentor: dhr Van den Dries (*) *weghalen niet van toepassing*
Voorletters bedrijfsmentor: H.
Titulatuur bedrijfsmentor:
Functie bedrijfsmentor: Projectleider
Doorkiesnummer bedrijfsmentor:
Email bedrijfsmentor: h.vandendries@newcompliance.nl
NB: bedrijfsmentor mag dezelfde zijn als de opdrachtgever

Doorkiesnummer afstudeerder:
Functie afstudeerder (deeltijd/duaal):

Titel afstudeeropdracht: Herontwikkeling van de OK Cockpit voor het sterilisatie proces van gereedschap en apparatuur van ziekenhuizen bij NewCompliance.

Opdrachtomschrijving

1. Bedrijf

NewCompliance is een jong bedrijf opgericht in 2006 dat innovatieve producten, gericht op het verbeteren van patiëntveiligheid en productiviteit, ontwikkelt en vermarkt. NewCompliance heeft 14 fulltime medewerkers in dienst en er wordt hard gewerkt aan de ontwikkeling en vermarkting van verschillende nieuwe technologische producten.

NewCompliance richt zich volledig op het verbeteren van de patiëntveiligheid en productiviteit met behulp van innovatieve producten die gebruiksvriendelijk en toekomstgericht zijn. Hierbij ligt de nadruk op technologische oplossingen die gerealiseerd worden door een combinatie van hardware en software.

Het meest verkochte product van NewCompliance is het OK Cockpit. Hierop kunnen artsen en doctoren de variabelen (Bijv. luchtdruk, aantal openingen van de deur, temperatuur van de patiënt) zien in een oogopslag. Hiermee kan het slagingspercentage van operaties worden verhoogd en op een simpele wijze binnen de afgesproken normen worden gehouden. De gegevens die vergaard zijn via de OK Cockpit kunnen worden bekeken in een analyse tool. Hierin kunnen een groot aantal filters worden toegepast op de gegevens.

2. Probleemstelling

Het huidige product OK Cockpit is ontwikkelt voor ziekhuizen voor in het OK om alle medische data bij de hand te hebben. Nu willen de ziekhuizen ook een soort gelijk systeem voor de sterilisatie afdeling, zodat er gemakkelijk gezien kan worden wat er gedaan moet worden, wat er gedaan is en of een bepaalde opdracht spoed heeft. De ziekhuizen willen het systeem over een half jaar in gebruik nemen. De huidige OK Cockpit is niet aanpasbaar genoeg om hergebruikt te worden voor de nieuwe Cockpit. Er kunnen delen van de Cockpit hergebruikt worden maar een groot deel moet herschreven worden om toepasbaar te zijn voor de nieuwe Cockpit. Voor de nieuwe Cockpit moeten ook een paar belangrijke algoritmes bedacht worden om de grote hoeveelheid data juist weer te geven in Cockpit met gebruik van tabellen.

3. Doelstelling van de afstudeeropdracht

De doelstelling van de opdracht is om aan het einde van het afstuderen een werkende Cockpit voor de afdeling sterilisatie op te leveren die getest is en in gebruik genomen kan worden. Ook moet deze nieuwe Cockpit aanpasbaar zijn voor nieuwe projecten of een uitbreiding.

4. Resultaat

Het resultaat van de opdracht zal zijn dat het ziekhuis de sterilisatie van spullen op een betere en snellere manier kan uitvoeren, wat tijd en geld bespaart. Ook zal dit ten goede komen van de veiligheid van de patiënten omdat de software beter kan bijhouden wat er gedaan is en hoe.

5. Uit te voeren werkzaamheden, inclusief een globale fasering, mijlpalen en bijbehorende activiteiten

Activiteiten:
1. Introductie tot de OK Cockpit en de scrum methode bij NewCompliance
2. Opstellen van een plan van aanpak volgens PRINCE2
3. Opstellen van requirements en uitbreidingen voor het systeem door middel van Interviews met CEO en projectleider van NewCompliance
4. Het prioriteren van de requirements met de stakeholders met behulp van MoSCoW. Dit zal aan het begin van elk sprint gedaan worden om de “Must have” requirements te prioriteren voor de sprint.
5. Opstellen van een functioneel ontwerp voor een nieuwe Cockpit op basis van de vergaarde requirements
6. Opstellen van een technisch ontwerp voor de nieuwe Cockpit
7. Opstellen testrapportage voor de gemaakte Cockpit. Dit zijn de uit te voeren tests als de software is geprogrammeerd.
8. Programmeren van de nieuw ontworpen Cockpit in PHP
9. Testen zoals beschreven in het testrapport
10. Afstudeerverslag maken

Planning:

Sprint	Start	Einde	Beschrijving*
1	04/05/2015	17/05/2015(2 weken)	Het opzetten van de afstudeerstage inwerken in de code, scrum en opstellen van plan van aanpak.
2	18/05/2015	31/05/2015(2 weken)	Het bouwen van de basis van de Cockpit
3	01/06/2015	14/06/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
4	15/06/2015	28/06/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
5	29/06/2015	12/07/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
6	13/07/2015	26/07/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
7	27/07/2015	09/08/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
8	10/08/2015	30/08/2015(3 weken)	Het afronden van het afstudeerproject

***In de sprints wordt het stuk van het systeem uitgewerkt met de klant, ontworpen, gebouwd en getest.**

Extra uitgangpunten:
- De interviews, die worden gehouden met de CEO, om de requirements vast te stellen zullen worden gedocumenteerd
- Er zullen wekelijkse overlegmomenten zijn met de CEO en stagebegeleider van NewCompliance om te kunnen bepalen of de Cockpit voldoet aan de requirements
- De stagiair wordt begeleid door een programmeur van de OK Cockpit. Aan deze persoon kunnen vragen over code of documentatie worden gesteld
- Er is wekelijks een programmeursoverleg, waarin alle werkzaamheden van de week worden doorgenomen met alle programmeurs
- Het gekozen ontwikkelproces voor deze opdracht is SCRUM
- De afstudeerder krijgt wekelijks tijd om te werken aan zijn afstudeerverslag
- De te hanteren testmethodiek is TestGoal
- De ontwerptechniek die gebruikt zal worden is UML

6. Op te leveren (tussen)producten

Het bedrijf zal de gemaakte requirements en UML diagrammen aangeleverd willen hebben aan het eind van de stage samen met de gemaakte code. In de overige documentatie is het bedrijf niet geïnteresseerd.

7. Te demonstreren competenties en wijze waarop

Competentie 1.4: Uitvoeren analyse door definitie van requirements
Het te ontwerpen informatiesysteem heeft nog geen duidelijke eisen en eigenschappen op het moment dat de afstudeerstage begint. Door middel van interviews zal er achterhaalt worden wat de requirements zijn. Om vast te stellen of de door mij genoteerde requirements kloppen zal er feedback gevraagd worden aan de projectleider van NewCompliance. Het in kaart brengen van de requirements zal dus volledig en dekkend moeten zijn. Tevens zullen de requirements met behulp van de MoSCoW methoden geprioriteerd worden.

Competentie 3.2: Ontwerpen systeemdeel
De nieuwe Cockpit wordt gebouwd uit een deel van een al bestaand systeem, dit systeem is de OK Cockpit. Ook zal het gebruikmaken van de analyse tool en een database. Er zal een ontwerp gemaakt worden wat past bij het bestaande systeem. Hiervoor wordt de architectuur van de huidige applicatie en database meegenomen in het ontwerp van de nieuwe Cockpit. Bovendien moet de Cockpit blijven werken met de analyse tool.

Competentie 3.3: Bouwen applicatie
Uiteindelijk resultaat is een werkende Cockpit. De uiteindelijk applicatie moet samen kunnen werken met de bestaande software van OK Cockpit en analyse tool. Verder moet de Cockpit zo geschreven zijn dat toekomstige veranderingen mogelijk blijven. De uiteindelijke Cockpit moet ook getest kunnen worden, op basis van het daarvoor gemaakte testplan.

Competentie 3.5: Uitvoeren en rapporteren over het testproces
De functionaliteit van de Cockpit moet worden getest. Er wordt hierbij een logische testontwerp en een fysiek testontwerp gemaakt. In alle waarschijnlijkheid zal er ook gebruik gemaakt worden van testscripts. Deze tests zullen worden uitgevoerd na het maken van de Cockpit. Hiervan zal er rapportage van worden gemaakt, die zal vaststellen of de Cockpit klaar is om in gebruik genomen kan worden.

Bijlage B: Plan van aanpak

Plan van aanpak

Project aanpak voor de ontwikkeling van de CSA cockpit voor NewCompliance.

Shawn Steinz

10-1-2015

Versie 1.1

Inhoudsopgave

1. Inleiding 1
2. Contact 2
3. Opdrachtomschrijving 3
3.1 Bedrijf 3
3.2 Probleembeschrijving 3
3.3 Doelstelling van de opdracht 4
3.3.1 Activiteiten: 4
4. Systeem scope 4
5. Randvoorwaarden 4
6. Methoden en technieken 5
7. Risicofactoren 5
8. Projectorganisatie 5
9. Wijze van rapporteren 6
10. Stakeholders 6
11. Benodigde middelen 6
12. Planning 7
12.1 Planning Stage Activiteiten 7
12.2 Planning Stage gerelateerde school dagen 7
12.3 Bedrijfsplanning 8
12.4 Bedrijfsplanning afstudeerdeel 9
13. Beschrijving mijlpaalproducten 10

1. Inleiding

Er is vraag naar ondersteunende software voor de CSA(Centrale Sterialisatie Afdeling) van het Erasmus MC, deze heeft als basis de dashboards van de OK Cockpit. Deze software geeft gemakkelijk belangrijke data weer. Op basis van de data kunnen medewerkers gepaste handelingen uitvoeren. Dit zorgt voor een meer productievere en veiligere omgeving

Deze software wordt ontwikkeld bij NewCompliance. NewCompliance is een klein ICT-bedrijf in Zoetermeer die zich specialiseert op innovatieve producten, gericht op het verbeteren van patiëntveiligheid en productiviteit

Tijdens het project wordt er gebruik gemaakt van SCRUM. SCRUM zorgt voor meer flexibiliteit, communicatie en samenwerking. Dit omdat er regelmatig standup meetings gehouden worden om al het werk te bespreken, daarna wordt er een werkverdeling gemaakt.

2. Contact

In dit hoofdstuk staan de contact gegevens van de verantwoordelijke mensen v.d. opdracht.

Naam: Dhr. B. Wiesman
Mail: b.wiesman@newcompliance.nl
Telefoon 0641276112

Naam: Dhr. H. Van den Dries
Mail: h.vandendries@newcompliance.nl
Telefoon 06-53813698

Naam: Dhr. B. van Oost

Naam: Dhr. S.P. Steinz
Mail: s.steinz@newcompliance.nl
Telefoon 06-36114901

3. Opdrachtomschrijving

3.1 Bedrijf

Het bedrijf, waar het project wordt uitgevoerd, is NewCompliance.

NewCompliance

NewCompliance is een jong bedrijf opgericht in 2006 dat innovatieve producten, gericht op het verbeteren van patiëntveiligheid en productiviteit, ontwikkelt en vermarkt. NewCompliance heeft 14 fulltime medewerkers in dienst en er wordt hard gewerkt aan de ontwikkeling en vermarkting van verschillende nieuwe technologische producten.

NewCompliance richt zich volledig op het verbeteren van de patiëntveiligheid en productiviteit met behulp van innovatieve producten die gebruiksvriendelijk en toekomstgericht zijn. Hierbij ligt de nadruk op technologische oplossingen die gerealiseerd worden door een combinatie van hardware en software.

Het meest verkochte product van NewCompliance is het OK Cockpit. Hierop kunnen artsen en doctoren de variabelen (Bijv. luchtdruk, aantal openingen van de deur, temperatuur van de patiënt) zien in een oogopslag. Hiermee kan het slagingspercentage van operaties worden verhoogd en op een simpele wijze binnen de afgesproken normen worden gehouden. De gegevens die vergaard zijn via de OK Cockpit kunnen worden bekeken in een analyse tool. Hierin kunnen een groot aantal filters worden toegepast op de gegevens.

3.2 Probleembeschrijving

Het huidige product OK Cockpit is ontwikkelt voor ziekhuizen voor in het OK om alle medische data bij de hand te hebben. Nu wil het Erasmus MC ook een soort gelijk dashboardsysteem voor de CSA(Centrale Sterilisatie Afdeling). Dit zodat er gemakkelijk gezien kan worden wat er gedaan moet worden, wat er gedaan is en of een bepaalde opdracht spoed heeft. Het ziekgenhuis wil het systeem over een half jaar in gebruik nemen. Delen van de huidige OK Cockpit kunnen hergebruikt worden voor de nieuwe Cockpit. Er kunnen delen van de Cockpit hergebruikt worden, maar een groot deel moet wel herschreven worden om toepasbaar te zijn voor de nieuwe Cockpit. Voor de nieuwe Cockpit moeten ook een paar belangrijke algoritmes bedacht worden om de grote hoeveelheid data juist weer te geven in Cockpit met gebruik van tabellen.

3.3 Doelstelling van de opdracht

De doelstelling van de opdracht is om aan het einde van het project een werkende Cockpit voor de afdeling sterilisatie op te leveren, die getest is en in gebruik genomen kan worden. Ook moet deze nieuwe Cockpit aanpasbaar zijn voor nieuwe projecten of een uitbreiding. Voor het realiseren zullen enkele werkzaamheden gedaan moeten worden namelijk:

3.3.1 Activiteiten:

1. Introductie tot de OK Cockpit en NewCompliance
2. Opstellen van een plan van aanpak
3. Opstellen van requirements en uitbreidingen voor het systeem door middel van Interviews met CEO en projectleider van NewCompliance en het Erasmus MC
4. Het prioriteren van de requirements met de stakeholders met behulp van MoSCoW. Dit zal aan het begin van elk sprint gedaan worden om de “Must have” requirements te prioriteren voor de sprint.
5. Opstellen van een functioneel ontwerp voor een nieuwe Cockpit op basis van de vergaarde requirements
6. Opstellen van een technisch ontwerp voor de nieuwe Cockpit
7. Opstellen testrapportage voor de gemaakte Cockpit. Dit zijn de uit te voeren tests als de software is geprogrammeerd.
8. Programmeren van de nieuw ontworpen Cockpit in PHP
9. Testen zoals beschreven in het testrapport
10. Afstudeerverslag maken

4. Systeem scope

Het huidige product OK Cockpit is ontwikkelt voor ziekenhuizen voor in het OK om alle medische data bij de hand te hebben. Nu wil het EMC ook een soort gelijk systeem voor de CSA(Centrale Sterilisatie Afdeling), zodat er gemakkelijk gezien kan worden wat er gedaan moet worden, wat er gedaan is en of een bepaalde opdracht spoed heeft. Het EMC wil het systeem over een half jaar in gebruik nemen. De huidige OK Cockpit is niet aanpasbaar genoeg om uitgebreid te worden voor de nieuwe Cockpit. Er kunnen delen van de Cockpit hergebruikt worden maar een groot deel moet herschreven worden om toepasbaar te zijn voor de nieuwe Cockpit. Voor de nieuwe Cockpit moeten ook een paar belangrijke algoritmes bedacht worden om de grote hoeveelheid data juist weer te geven in Cockpit met gebruik van tabellen en grafieken.

5. Randvoorwaarden

- Om het project succesvol te kunnen voltooien zullen er duidelijke afspraken gemaakt moeten worden met de opdrachtgever.
- De CSA Cockpit moet gebruik maken van het laravel framework om onderhoudbaarheid voor NewCompliance te verhogen
- NewCompliance zal de opdrachtnemer voorzien van benodigde software

6. Methoden en technieken

Voor de softwareontwikkeling wordt er gebruik gemaakt van Scrum. Scrum is een Agile softwareontwikkel methode waar gewerkt wordt in korte sprints waarbij aan het einde van elke sprint producten worden opgeleverd.

Er is gekozen voor Scrum voor de volgende redenen:

- (Flexibiliteit) Omdat er gewerkt word in korte sprint kan er ingespeeld kan worden op verandering zonder dat er veel werk verloren gaat.
- (Communicatie) Omdat er veel producten opleveren worden kunnen er vaak producten getoond worden aan de stakeholders waardoor een dialoog ontstaat tussen stakeholder en ontwikkelaar.
- (Samenwerking) Één keer per week is er een programmeurs overleg waarin de werkverdeling wordt gemaakt en stand van zaken wordt besproken hierdoor is het gemakkelijk om samen te werken

Als back log en scrum bord wordt er gebruik gemaakt van Trello. Omdat het project een looptijd heeft van meerdere maanden is er gekozen om de sprints twee weken te laten duren.

Er is voor gekozen om alle modellen in UML uit te voeren zodat deze leesbaar en herbruikbaar zijn. UML 2.0 bevat een groot aantal modeleer technieken zodat voor elke fase van het project de juiste techniek gebruikt kan worden.

7. Risicofactoren

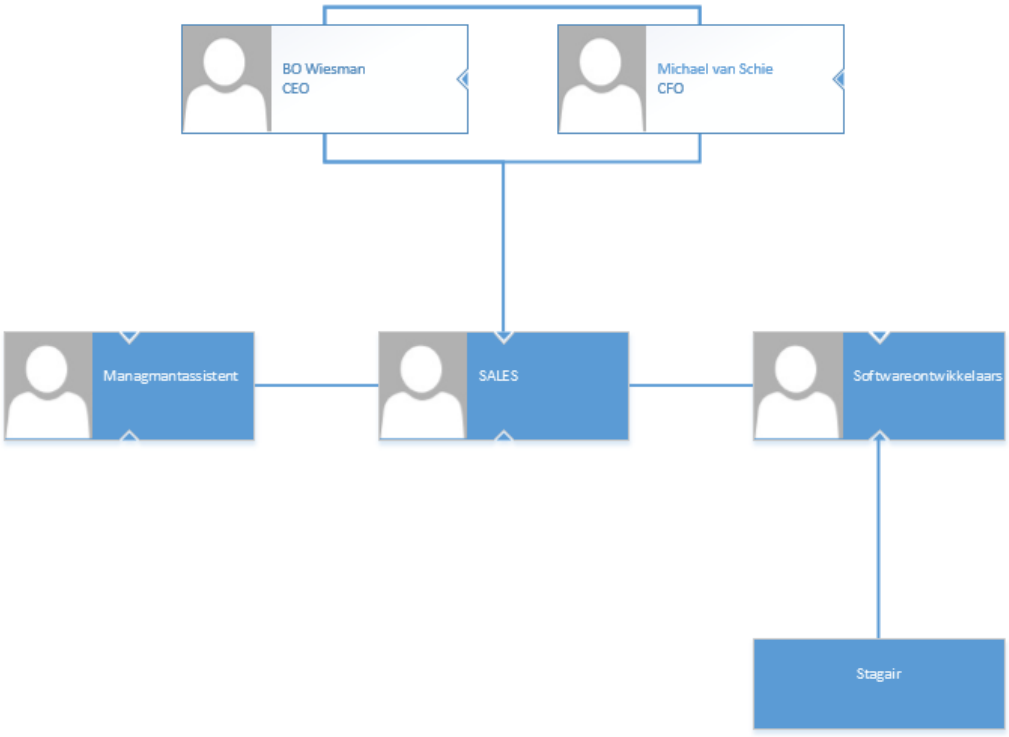
Alle wensen van de opdrachtgevers moeten worden vervuld. Deze requirement brengt het bekende scope creep met zich mee. Dat wil zeggen dat met het wijzigen van de requirements de scope van het systeem ook groter wordt. Hierdoor kan de planning uitlopen. Dit risico kan beheerst worden door gebruik te maken van requirement management.

Tijdens het project zal er met de ICT afdeling van het ziekhuis en leveranciers van het ziekhuis gecommuniceerd moeten worden. Het werken met derden bedrijven kan zorgen voor vertraging omdat er service van andere nodig is.

Er moet ondersteunding zijn voor meerdere soorten webbrowsers. Dit zorgt ervoor dat er beter gekeken moet worden naar de plug-ins die gebruikt gaan worden.

8. Projectorganisatie

Tijdens de afstudeerstage zal ik in een bestaand project ingewerkt worden. Deze projectgroep heeft elke week een overleg.



9. Wijze van rapporteren

Elke twee weken zal er een voortgangsbespreking zijn met de Projectleider. In deze bespreking zal de planning van de komende twee week doorgenomen worden en worden de producten die af zijn besproken. Aan de detail planning kan worden gezien wat er komende week opgeleverd zal worden. Aan de hand van de mijlpaalproducten kan hij inzien of de planning gehaald kan worden.

10. Stakeholders

De volgende relaties zijn gemaakt om dit project op te zetten:

- Dhr. H. van den Dries Opdrachtgever/bedrijfsmentor
- Dhr. B. Wiesman Opdrachtgever
- Dhr. B. van Oost Unithoofd CSA Erasmus MC

11. Benodigde middelen

De volgende software zal nodig zijn in dit project:

- Microsoft Office(Word, Excel, PowerPoint en Outlook) & Microsoft Project
- Visual Paradigm
- PHP
- Apache
- phpstorm
- Sql server
- MongoDB
- Laravel

12. Planning

12.1 Planning Stage Activiteiten

Sprint	**Start**	**Einde**	**Beschrijving***
0 project opzet	04/05/2015	24/05/2015(3 weken)	Het opzetten van de afstudeerstage inwerken in de code, scrum en opstellen van plan van aanpak.
1	25/05/2015	14/06/2015(3 weken)	Het bouwen van de basis van de Cockpit
2	15/06/2015	28/06/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
3	29/06/2015	12/07/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
4	13/07/2015	26/07/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
5	27/07/2015	09/08/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint
6	10/08/2015	23/08/2015(2 weken)	Het toevoegen van functies die het hoogst geprioriteerd zijn voor deze sprint

*In de sprints wordt het stuk van het systeem uitgewerkt met de klant, ontworpen, gebouwd en getest.

12.2 Planning Stage gerelateerde school dagen

Activiteit	Stage Week	Datum
Bedrijfsbezoek	4	01/06/2015 – 08/06/2015
Voortgangsverslag	8	29/06/2015 – 05/07/2015
Bespreking afstudeerdossier	12	27/07/2015 – 02/08/2015
Tussentijds assessment	-	30/08/2015 – 14/09/2015
Inleveren afstudeerdossier	-	Inlever datum 05/10/2015

12.3 Bedrijfsplanning

ID	Task Name	Duration	Start	Finish	Predecessors
0	**2015-06-03 Implementatie CSA Cockpit**	**279 days**	**Wed 11/03/15**	**Thu 31/03/16**	
1	**1 Aftrap project**	**1 day**	**Wed 11/03/15**	**Wed 11/03/15**	
2	1.1 Kick-off CSA Cockpit	1 day	Wed 11/03/15	Wed 11/03/15	
3	**2 ICT acties algemeen**	**241 days**	**Thu 12/03/15**	**Tue 09/02/16**	**1**
4	2.1 Goedkeuring projectmanagement ICT	50 days	Thu 12/03/15	Wed 20/05/15	
5	**2.2 Acties t.b.v. servers**	**158 days**	**Thu 21/05/15**	**Thu 24/12/15**	**4**
6	**2.2.1 Virtuele testserver inrichten met VPN toegang**	**30 days**	**Thu 21/05/15**	**Wed 01/07/15**	
7	2.2.1.1 Aanvraag virtuele test server	5 days	Thu 21/05/15	Wed 27/05/15	
8	2.2.1.2 Oplevering test server met VPN toegang	20 days	Thu 28/05/15	Wed 24/06/15	7
9	2.2.1.3 Configuratie test virtuele server	5 days	Thu 25/06/15	Wed 01/07/15	8
10	**2.2.2 Acceptatie server inrichten met VPN toegang**	**30 days**	**Fri 02/10/15**	**Thu 12/11/15**	**52,18**
11	2.2.2.1 Aanvraag acceptatie server	5 days	Fri 02/10/15	Thu 08/10/15	
12	2.2.2.2 Oplevering acceptatie server met VPN toegang	20 days	Fri 09/10/15	Thu 05/11/15	11
13	2.2.2.3 Configuratie acceptatie server	5 days	Fri 06/11/15	Thu 12/11/15	12
14	**2.2.3 Productie server inrichten met VPN toegang**	**30 days**	**Fri 13/11/15**	**Thu 24/12/15**	**10**
15	2.2.3.1 Aanvraag productie server	5 days	Fri 13/11/15	Thu 19/11/15	
16	2.2.3.2 Oplevering productie server met VPN toegang	20 days	Fri 20/11/15	Thu 17/12/15	15
17	2.2.3.3 Configuratie productie server	5 days	Fri 18/12/15	Thu 24/12/15	16
18	**2.3 Documentatie en goedkeuringen ICT**	**88 days**	**Thu 04/06/15**	**Thu 01/10/15**	**29**
19	2.3.1 Functioneel ontwerp	5 days	Thu 04/06/15	Wed 10/06/15	
20	2.3.2 Overige acties ICT (aan te vullen met info Marjan)	83 days	Thu 11/06/15	Thu 01/10/15	19
21	2.4 Toegang database acceptatie omgeving T-DOC	1 day	Thu 25/06/15	Thu 25/06/15	8
22	2.5 Bestaande displays voorzien van recente browser	9 days	Fri 15/01/16	Wed 27/01/16	
23	2.6 Instellen adres en opstarten t.b.v. clients	9 days	Thu 28/01/16	Tue 09/02/16	22
24	2.7 Autorisatie gebruikers o.b.v. LDAP	9 days	Thu 28/01/16	Tue 09/02/16	
25	**3 Koppelingen bronsystemen**	**165 days**	**Thu 12/03/15**	**Mon 26/10/15**	**1**
26	**3.1 Parameter- en procesanalyse informatie systemen**	**60 days**	**Thu 12/03/15**	**Wed 03/06/15**	
27	3.1.1 Globale parameter analyse bronsystemen	28 days	Thu 12/03/15	Mon 20/04/15	
28	3.1.2 Afzonderlijk afstemmingsoverleg betrokken leveranciers / ICT verantwoordelijken (T-doc, Harmony en 3M)	27 days	Tue 21/04/15	Wed 27/05/15	27
29	3.1.3 Vastleggen parameter- en procesanalyse	2 days	Tue 02/06/15	Wed 03/06/15	28
30	**3.2 Technische realisatie koppeling T-doc**	**85 days**	**Thu 02/07/15**	**Mon 26/10/15**	**26,6**
31	3.2.1 Analyse database T-DOC voor koppeling	10 days	Thu 02/07/15	Wed 15/07/15	
32	3.2.2 Realisatie connectie T-doc > CSA Cockpit	10 days	Thu 16/07/15	Wed 29/07/15	31
33	3.2.3 Programmeren ODBC koppeling CSA Cockpit	60 days	Thu 30/07/15	Mon 19/10/15	32
34	3.2.4 Testen en valideren koppeling T-doc	5 days	Tue 20/10/15	Mon 26/10/15	33
35	**3.3 Technische realisatie koppeling Harmony**	**55 days**	**Thu 02/07/15**	**Mon 14/09/15**	**6,26**
36	3.3.1 Afstemmingsoverleg Harmony	15 days	Thu 02/07/15	Wed 22/07/15	
37	3.3.2 Realisatie connectie Harmony > CSA Cockpit	10 days	Thu 23/07/15	Wed 05/08/15	36
38	3.3.3 Inrichten koppeling Harmony	10 days	Thu 06/08/15	Tue 18/08/15	37
39	3.3.4 Programmeren interpretatie koppeling CSA Cockpit	15 days	Wed 19/08/15	Mon 07/09/15	38
40	3.3.5 Testen en valideren koppeling Harmony	5 days	Tue 08/09/15	Mon 14/09/15	39
41	**3.4 Technische realisatie koppeling 3M**	**25 days**	**Thu 02/07/15**	**Wed 05/08/15**	**6,26**
42	3.4.1 Realisatie connectie 3M > CSA Cockpit	10 days	Thu 02/07/15	Wed 15/07/15	
43	3.4.2 Inrichten koppeling 3M	5 days	Thu 16/07/15	Wed 22/07/15	42
44	3.4.3 Programmeren interpretatie koppeling CSA Cockpit	5 days	Thu 23/07/15	Wed 29/07/15	43
45	3.4.4 Testen en valideren koppeling 3M	5 days	Thu 30/07/15	Wed 05/08/15	44
46	**4 Ontwikkeling en implementatie software**	**131 days**	**Thu 28/05/15**	**Tue 24/11/15**	**28**
47	**4.1 Dashboards**	**52 days**	**Thu 28/05/15**	**Fri 07/08/15**	
48	4.1.1 Aanpassingen weergaves voor track and trace	10 days	Thu 28/05/15	Wed 10/06/15	
49	4.1.2 Modules bouwen	42 days	Thu 11/06/15	Fri 07/08/15	48
50	4.2 Dashboards opbouwen	2 days	Mon 10/08/15	Tue 11/08/15	49
51	4.3 Checklists aanmaken	10 days	Mon 10/08/15	Thu 20/08/15	47
52	**4.4 Nieuwe invulformulier(en) inclusief links & touch optimalisatie**	**30 days**	**Fri 21/08/15**	**Wed 30/09/15**	**51**
53	4.4.1 Invoer bijna prikincidenten	5 days	Fri 21/08/15	Thu 27/08/15	
54	4.4.2 Invoer klachten OK	5 days	Fri 28/08/15	Wed 02/09/15	53
55	4.4.3 Invoer RIC controle	5 days	Thu 03/09/15	Wed 09/09/15	54
56	4.4.4 Invoer daglijsten	15 days	Thu 10/09/15	Wed 30/09/15	55
57	**4.5 Analyse software bouwen (huidige selectie grafieken)**	**35 days**	**Thu 01/10/15**	**Wed 18/11/15**	**47,51,52**
58	4.5.1 Analyse software toesplitsen op nieuwe eenheid	15 days	Thu 01/10/15	Wed 21/10/15	
59	4.5.2 Filters bouwen	20 days	Thu 22/10/15	Wed 18/11/15	58
60	4.6 Inrichting configuratie, dashboards, analysefacetten & rechten	4 days	Thu 19/11/15	Tue 24/11/15	47,51,52,57
61	**5 Acties t.b.v. hardware**	**133 days**	**Thu 04/06/15**	**Thu 03/12/15**	**26**
62	5.1 Afstemming display locaties	1 day	Thu 04/06/15	Thu 04/06/15	
63	5.2 Afstemming te leveren displays	20 days	Fri 05/06/15	Thu 02/07/15	62
64	5.3 Bestellen benodigde displays	1 day	Thu 01/10/15	Thu 01/10/15	63,52
65	5.4 Levering displays	30 days	Fri 02/10/15	Thu 12/11/15	64
66	5.5 Configureren displays	5 days	Fri 13/11/15	Thu 19/11/15	65
67	5.6 Ophanging displays	10 days	Fri 20/11/15	Thu 03/12/15	66
68	**6 Acceptatietest en oplevering**	**56 days**	**Wed 25/11/15**	**Wed 10/02/16**	**46,25,10**
69	6.1 Migratie test naar acceptatie	5 days	Wed 25/11/15	Tue 01/12/15	
70	**6.2 Acceptatietest**	**40 days**	**Wed 02/12/15**	**Tue 26/01/16**	**69**
71	6.2.1 Dashboard	20 days	Wed 02/12/15	Tue 29/12/15	
72	6.2.2 Invulformulieren	20 days	Wed 02/12/15	Tue 29/12/15	
73	6.2.3 Checklists	20 days	Wed 02/12/15	Tue 29/12/15	
74	6.2.4 Analysetool	20 days	Wed 30/12/15	Tue 26/01/16	71,72,73
75	6.3 Migratie naar productie-omgeving	10 days	Wed 27/01/16	Tue 09/02/16	70,61,14
76	6.4 Oplevering	1 day	Wed 10/02/16	Wed 10/02/16	75,3
77	**7 Scholing en nazorg**	**36 days**	**Thu 11/02/16**	**Thu 31/03/16**	**68**
78	7.1 Scholing dashboards	5 days	Thu 11/02/16	Wed 17/02/16	
79	7.2 Scholing analysetool	2 days	Thu 11/02/16	Fri 12/02/16	
80	7.3 Betatest periode	30 days	Thu 18/02/16	Wed 30/03/16	78,79
81	7.4 Evaluatie	1 day	Thu 31/03/16	Thu 31/03/16	80

12.4 Bedrijfsplanning afstudeerdeel

	Task Mode	Task Name	Work	Duration	Start
0		**Planning Programmeren afstude**	**962 hrs**	**120.25 days**	**Thu 5/28/15**
1		**Ontwikkeling en implementatie software**	**962 hrs**	**120.25 days**	**Thu 5/28/15**
2		**Dashboards**	**330 hrs**	**71.25 days**	**Thu 5/28/15**
3		Aanpassingen weergaves voor track and trace	80 hrs	10 days	Thu 5/28/15
4		**Modules bouwen**	**236 hrs**	**59.5 days**	**Thu 6/11/15**
5		Flex scopen	8 hrs	1 day	Thu 6/11/15
6		Capaciteit apparatuur t.o.v. volume in aantocht	10 hrs	10 hrs	Fri 6/12/15
7		Spoed netten	12 hrs	12 hrs	Mon 6/15/15
8		Spoed plan	12 hrs	12 hrs	Tue 6/16/15
9		Afwijkend gewicht	10 hrs	10 hrs	Thu 6/18/15
10		Netten in aantocht	8 hrs	8 hrs	Fri 6/19/15
11		Werklast	12 hrs	12 hrs	Mon 6/22/15
12		Verlooptijd flex scopen	8 hrs	8 hrs	Wed 6/24/15
13		OK klachten	8 hrs	8 hrs	Thu 6/25/15
14		Productie % tegen over k	10 hrs	10 hrs	Fri 6/26/15
15		RIC controle	12 hrs	12 hrs	Mon 7/6/15
16		Doorlooptijd	10 hrs	10 hrs	Tue 7/7/15
17		Prestatie cijfer	10 hrs	10 hrs	Thu 7/9/15
18		Netten in reparatie	8 hrs	8 hrs	Fri 7/10/15
19		BD-waarde	40 hrs	40 hrs	Mon 7/13/15
20		Checklists controle	10 hrs	10 hrs	Mon 8/3/15
21		Bezetting	10 hrs	10 hrs	Mon 8/24/15
22		Netten voor uitgifte	10 hrs	10 hrs	Tue 8/25/15
23		Verlopen eenheden	10 hrs	10 hrs	Thu 8/27/15
24		Gevaarlijke situatie meld	8 hrs	8 hrs	Fri 8/28/15
25		Netten uitgeleend	10 hrs	10 hrs	Sat 8/29/15
26		**Beeldschermen**	**14 hrs**	**1.75 days**	**Mon 8/31/15**
27		Vuile ruimte	2 hrs	2 hrs	Mon 8/31/15
28		Verpakruimte 1	2 hrs	2 hrs	Mon 8/31/15
29		Verpakruimte 2	2 hrs	2 hrs	Tue 9/1/15
30		Uitgifste CSA	2 hrs	2 hrs	Tue 9/1/15
31		Dagoudste	2 hrs	4 hrs	Tue 9/1/15
32		Ok X 5	4 hrs	2 hrs	Wed 9/2/15
33		Checklists aanmaken	80 hrs	10 days	Mon 7/20/15
		NewCompliance (dev.)	*80 hrs*		*Mon 7/20/15*
34		**Nieuwe invulformulier(en) inclusief links & touch optimalisatie**	**240 hrs**	**59 days**	**Mon 6/29/15**
39		**Analyse software bouwen (huidige selectie grafieken)**	**280 hrs**	**35 days**	**Wed 9/16/15**
40		Analyse software toesplitsen op nieuwe eenheid	120 hrs	15 days	Wed 9/16/15
		NewCompliance (dev.)	*120 hrs*		*Wed 9/16/15*
41		Filters bouwen	160 hrs	20 days	Wed 10/7/15
		NewCompliance (dev.)	*160 hrs*		*Wed 10/7/15*
42		Inrichting configuratie, dashboards, analysefacetten & rechten	32 hrs	4 days	Wed 11/4/15
		NewCompliance (dev.)	*32 hrs*		*Wed 11/4/15*

13. Beschrijving mijlpaalproducten

Requirements rapport
in dit rapport worden de requirements van het project opgesteld.

Technisch ontwerp
in dit rapport wordt het technisch ontwerp weergeven.

Functioneel ontwerp
in dit rapport wordt het functioneel ontwerp weergeven.

Applicatie
Hier wordt de MedCockpit opgeleverd

Testrapport
in dit rapport worden de testrapportage weergeven.

Bijlage C: Requirementsrapport

Requirements Rapport

Project aanpak voor de ontwikkeling van de CSA cockpit voor NewCompliance.

Shawn Steinz

6-8-2015

Versie 1.0

Inhoudsopgave

1. Inleiding 1
2. Systeem scope 2
3. Description of Requirements Based on Case Materials 3
3.1 Business Rules 3
4. Description of User Requirements Based on Case Materials and Interviews 4
5. Use Case Diagram 5
5.1 Actor List 5
5.2 Use Case Diagram 5
5.3 Use Case List (User Requirements) 6
6. Prioritized Requirements 7
6.1 Must Have 7
6.2 Should Have 7
6.3 Could Have 7
7. Functional and Non-Functional Software Requirements 8
8. Technical Constraints 8
9. Use Cases 8
9.1 Use Case Beschrijvingen 8
9.2 Use Case Stories 11
9.3 Use Case Slices 12
9.4 Uitvoering per sprint 13

1. Inleiding

De Cockpit wordt uitgebreid om aan de wensen van het EMC(Erasmus Medische Centrum) te voldoen. Het EMC wil een aantal extra functionaliteiten. In dit rapport worden de wensen van de klant en de opdrachtgever vastgelegd. OK Cockpit is niet toereikend genoeg voor de CSA dus wordt er een nieuwe CSA Cockpit gemaakt.

De requirements in dit document zijn vastgesteld samen met Henk van den Dries, de opdrachtgever van dit project. De verkregen requirements zijn ook met hem geprioriteerd.

De duur van het project is 17 weken. Daarom zullen de requirements geprioriteerd worden om de belangrijkste wensen te kunnen realiseren. Dit om er voor te zorgen dat de belangrijkste requirements uitgewerkt zijn als blijkt dat er niet genoeg tijd is om alle eisen te realiseren.

De opbouw van het document is als volgt. Als eerst zal de scope van het systeem beschreven worden. Wat is de functie van CSA Cockpit? Wat heeft het systeem nodig van de andere systemen?

Daarna volgt de beschrijving van de requirements. Die zijn gebaseerd op de bestaande documentatie. Er wordt hier onderscheid gemaakt tussen business- en gebruikersrequirements. Hier worden ook de business regels vastgelegd.

In het deel hierna worden de requirements die zijn voortgekomen uit interviews met Henk van den Dries beschreven. Aan elke requirement wordt een nummer toegekend om deze tracability toe te voegen.

Vervolgens worden de use cases uitgewerkt. Dit wordt gedaan met een use case diagram, een actor beschrijving en een use case list waar de koppeling met requirement staat aangegeven.

Hierna wordt een tabel getoond met de geprioriteerde requirements.

Dit wordt gevolgd door de verdeling van de requirements in functionele en niet functionele requirements.

De technische beperkingen worden hierna vastgelegd. Hier staan de frameworks vermeld.

Er wordt afgesloten met de use case uitgewerkt in beschrijvingen die vervolgens in use case stories zijn uitgewerkt en die worden opgesplitst in use case slices. Aan wordt hier de indeling van use case slice per sprint getoond die tijdens het project gemaakt is.

2. Systeem scope

Het huidige product OK Cockpit is ontwikkelt voor ziekenhuizen voor in het OK om alle medische data bij de hand te hebben. Nu wil het EMC ook een soort gelijk systeem voor de CSA(Centrale Sterilisatie Afdeling), zodat er gemakkelijk gezien kan worden wat er gedaan moet worden, wat er gedaan is en of een bepaalde opdracht spoed heeft. Het EMC wil het systeem over een half jaar in gebruik nemen. De huidige OK Cockpit is niet aanpasbaar genoeg om uitgebreid te worden voor de nieuwe Cockpit. Er kunnen delen van de Cockpit hergebruikt worden maar een groot deel moet herschreven worden om toepasbaar te zijn voor de nieuwe Cockpit. Voor de nieuwe Cockpit moeten ook een paar belangrijke algoritmes bedacht worden om de grote hoeveelheid data juist weer te geven in Cockpit met gebruik van tabellen en grafieken.

3. Description of Requirements Based on Case Materials

Via de bestaande software en een trello bord, zijn er een aantal requirements te verkrijgen. Deze requirements samen met de oude cockpit moet een nieuwe versie van de Cockpit worden.

Requirement ID	Requirement
NFSR1	Er moet een connectie met T-DOC en 3M gemaakt worden om data input te krijgen voor de CSA Cockpit.
NFSR2	De Cockpit moet real time data kunnen tonen
NFSR3	De Cockpit moet gebruik maken van een MSSQL database
FSR1	De werklast van de CSA moet berekent kunnen worden door het systeem. Dit geld voor alle afdelingen apart en ook gezamenlijk.
R1	Een CSA medewerker moet een gevaarlijke situatie formulier kunnen invullen in het systeem. Op het formulier moet automatische de datum ingevuld worden en de medewerker moet net nummer invullen of inscannen. Veder moet de afdeling, het voorwerp(mesje, naald en overig) en de medewerker zijn naam ingevuld worden.
R2	Een OK medewerker moet klachtformulier kunnen invullen in het systeem. Hier moet de categorie(beschadiging papier, vies, vermissing instrument, verkeerd instrument, verkeerd gemonteerd en overig), net nummer en afdeling ingevuld worden.
R3	De Dagoudste moet een checklist kunnen uitvoeren genaamd RIC controle. Hierbij worden 3 punten gevraagd net reinheid, instrumenten intact en of het net compleet is. Veder moet de naam van de medewerker ingevuld worden en de naam van de medewerker die gecontroleerd word.
R4	Een medewerker moet een daglijst kunnen invullen en wijzigen. Omdat de werklast binnen deze afdelingen kan verschillen van uur tot uur is het belangrijk dat binnen deze invoer wijzigingen aan te brengen zijn.
R5	Vuile ruimte dit scherm bevat informatie over Flex scopen, capaciteit apparatuur t.o.v. volume in aantocht, spoed netten, spoed plan en de werklast.
R6	Verpakruimte 1 dit scherm bevat informatie over verlooptijd Flex scopen, klachten van de ok, spoed net, spoed plan, productie % tegen over klachten, werklast en prestatie cijfer.
R7	Verpakruimte 2 dit scherm bevat informatie over netten in reparatie, capaciteit apparatuur t.o.v. volume in aantocht en bezetting
R8	Uitgifte CSA dit scherm bevat informatie over flex scopen, netten, spoed netten en spoed plan
R9	Dagoudste dit scherm bevat informatie over verlopen eenheden, bd-waarde, OK klachten, gevaarlijke situatie meldingen en de ingevulde checklists.
R10	OK's deze schermen tonen per OK de volgende informatie verlopen eenheden, netten uitgeleend, netten reparatie, spoed plan, spoed net en gevaarlijke situatie meldingen.
R11	In de bestaande analyse tool moet er een rapport gemaakt kunnen worden met betrekking op netten, tijdsduur, kwaliteit, ziekteverzuim, gebruiksintensiteit van netten, prestatie van medewerkers, kostenplaats en verschillen afwijkend gewicht tussen OK en CSA.

3.1 Business Rules

Business Rule ID	Requirement Uitleg
BR1	Gebruikersgegevens en patientengegevens mogen niet worden gepubliceerd of op een andere manier uitlekken

4. Description of User Requirements Based on Case Materials and Interviews

Uit interviews met Henk van den Dries en Bart van Oost zijn nog enkele requirements naar voren gekomen. Deze zijn samen met de al bestaande requirements verwerkt in de onderstaande tabel.

Requirement ID	Requirement	
R1	Een CSA medewerker moet een gevaarlijke situatie formulier kunnen invullen in het systeem. Op het formulier moet automatische de datum ingevuld worden en de medewerker moet net nummer invullen of inscannen. Veder moet de afdeling, het voorwerp(mesje, naald en overig) en de medewerker zijn naam ingevuld worden.	Case
R2	Een OK medewerker moet klachtformulier kunnen invullen in het systeem. Hier moet de categorie(beschadiging papier, vies, vermissing instrument, verkeerd instrument, verkeerd gemonteerd en overig), net nummer en afdeling ingevuld worden.	Case
R3	De Dagoudste moet een checklist kunnen uitvoeren genaamd RIC controle. Hierbij worden 3 punten gevraagd net reinheid, instrumenten intact en of het net compleet is. Veder moet de naam van de medewerker ingevuld worden en de naam van de medewerker die gecontroleerd word.	Case
R4	Een medewerker moet een daglijst kunnen invullen en wijzigen. Omdat de werklast binnen deze afdelingen kan verschillen van uur tot uur is het belangrijk dat binnen deze invoer wijzigingen aan te brengen zijn.	Case
R5	Vuile ruimte dit scherm bevat informatie over Flex scopen, capaciteit apparatuur t.o.v. volume in aantocht, spoed netten, spoed plan en de werklast.	Case
R6	Verpakruimte 1 dit scherm bevat informatie over verlooptijd Flex scopen, klachten van de ok, spoed net, spoed plan, productie % tegen over klachten, werklast en prestatie cijfer.	Case
R7	Verpakruimte 2 dit scherm bevat informatie over netten in reparatie, capaciteit apparatuur t.o.v. volume in aantocht en bezetting	Case
R8	Uitgifte CSA dit scherm bevat informatie over flex scopen, netten, spoed netten en spoed plan	Case
R9	Dagoudste dit scherm bevat informatie over verlopen eenheden, bd-waarde, OK klachten, gevaarlijke situatie meldingen en de ingevulde checklists.	Case
R10	OK's deze schermen tonen per OK de volgende informatie verlopen eenheden, netten uitgeleend, netten reparatie, spoed plan, spoed net en gevaarlijke situatie meldingen.	Case
R11	In de bestaande analyse tool moet er een rapport gemaakt kunnen worden met betrekking op netten, tijdsduur, kwaliteit, ziekteverzuim, gebruiksintensiteit van netten, prestatie van medewerkers, kostenplaats en verschillen afwijkend gewicht tussen OK en CSA.	Case
R12	De medewerkers van de CSA moeten een apparaat checklist kunnen invullen. Deze toont de status van de apparatuur en of er gepaste acties uitgevoerd moeten worden.	Interview

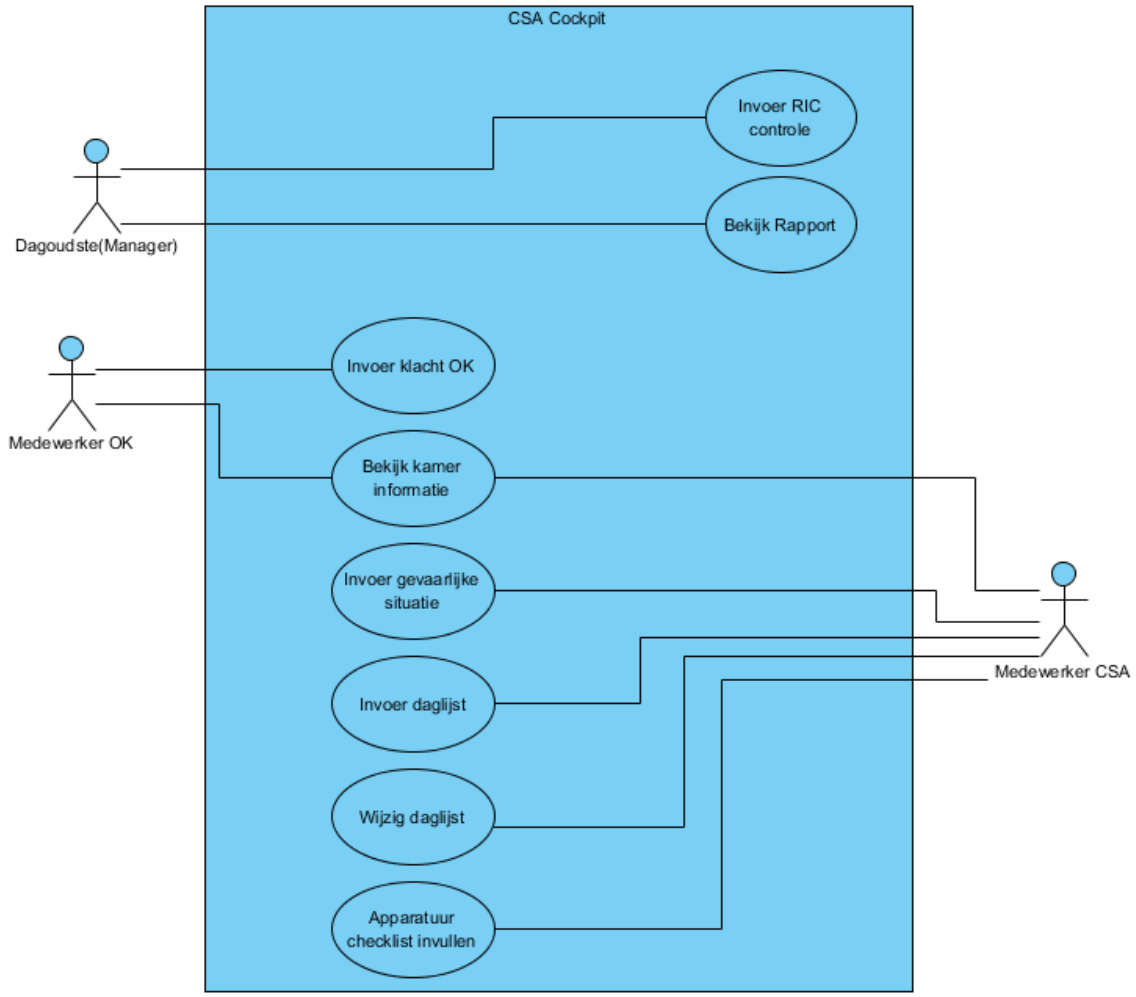
5. Use Case Diagram

Op basis van de vergaarde requirements kan er een Use Case Diagram worden gemaakt. Dit is een grafische weergave van de mogelijke gebruikers situaties. Er zal worden begonnen met een Actorenlijst gevolgd door een Use Case diagram. Er zal worden afgesloten met een use case lijst.

5.1 Actor List

Actor	Role	Description
CSA Medewerker	User	De actor kan daglijsten invoeren en wijzigen ook kan hij gevaarlijke situatie formulier invoeren en ook kan hij de Dashbord module bekijken.
OK medewerker	User	De actor kan een OK klacht invoeren en hij kan de Dashbord module bekijken.
Dagoudste(Manager)	Superuser	De actor kan RIC controle invoeren en data bekijken in de analyse tool.

5.2 Use Case Diagram



5.3 Use Case List (User Requirements)

Use Case ID	Use Case	Description	Requirement nr
U1	Invoer gevaarlijke situatie	De CSA medewerker vult het gevaarlijke situatie formulier in.	R1
U2	Invoer klachten OK	De OK medewerker vult het klachten formulier in.	R2
U3	Invoer RIC controle	De Dagoudste op de CSA vult het RIC formulier in.	R3
U4	Invoer daglijst	De CSA medewerker vult de daglijst in.	R4
U5	Wijzig daglijst	De CSA medewerker wijzigt de daglijst.	R4
U6	Bekijk kamer Module	De actor ziet een informatie scherm die is gebaseerd op de kamer waar hij zich bevind.	R5, R6, R7, R8, R9, R10
U7	Bekijk Rapport	De Dagoudste kan hier alle opgeslagen gegevens in een rapport weergeven.	R11
U8	Apparatuur checklist invullen	De CSA medewerker vult de apparatuur checklists in.	R12

6. Prioritized Requirements

De hiervoor beschreven use cases zullen in dit hoofdstuk worden geprioriteerd. Dit om te kijken waar de prioritiet licht binnen dit project. Per sprint zal gepriotireerd worden welke use casa slices gedaan zullen worden. Dit zal gebeuren met de MoSCoW methode. Deze methode verdeeld de requirements in 4 niveau's via prioriteit, namelijk:

- M - must haves: deze requirements moeten in het eindresultaat terugkomen, zonder deze requirements is het product niet bruikbaar;
- S - should haves: deze requirements zijn zeer gewenst, maar zonder is het product wel bruikbaar;
- C - could haves: deze requirements zullen alleen aan bod komen als er tijd genoeg is;
- W - won't haves: deze requirements zullen in dit project niet aan bod komen maar kunnen in de toekomst, bij een vervolgproject, interessant zijn.

6.1 Must Have

Requirements	**U1, U2, U3, U4, U5, U6, U8**

6.2 Should Have

Er is geen Requirement in deze klasse geplaatst.

6.3 Could Have

Requirement	**U7**

7. Functional and Non-Functional Software Requirements

Requirement ID	Requirement Uitleg	Source
NFSR1	Er moet een connectie met T-DOC en 3M gemaakt worden om data input te krijgen voor de CSA Cockpit.	Case
NFSR2	De Cockpit moet real time data kunnen tonen	Interview
NFSR3	De Cockpit moet gebruik maken van een MSSQL database	Case
FSR1	De werklast van de CSA moet berekent kunnen worden door het systeem. Dit geld voor alle afdelingen apart en ook gezamenlijk.	Case

8. Technical Constraints

De technische constraints van de CSA Cockpit zijn als volgt:

- De CSA Cockpit moet geschreven worden in PHP
- De interface van de CSA Cockpit mag niet volledig opnieuw worden ingericht.

9. Use Cases

In deze sprint zullen alle Must Have use cases beschreven worden door middel van main flow beschrijvingen. De beschrijvingen geven een beeld van de interactie tussen het systeem en de actoren. Vervolgens worden de use case beschrijving omgezet in use case stories. De use case stories worden vervolgens opgesplitst in Use case Slices. Deze slices zullen dan over de Sprints verdeeld worden.

9.1 Use Case Beschrijvingen

Invoer gevaarlijke situatie

name	Invoer gevaarlijke situatie
ID	U1
Korte beschrijving	De CSA medewerker vult het gevaarlijke situatie formulier in.
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: ”Meld gevaarlijke situatie”. 2. Het systeem toont gevaarlijke situatie formulier met de datum van die dag. 3. De actor scant het net nummer of vult deze handmatig in. 4. De actor vult de afdeling in. 5. De actor vult klacht in(mesje, naald of overig). 6. De actor vult zijn naam in. 7. De actor drukt op opslaan. 8. Systeem toont data opgeslagen en toont klachten scherm
Postconditie	Het gevaarlijke situatie formulier is ingevuld.
Alternatieve flows	2a. T-doc is niet beschikbaar.

Invoer klachten OK

name	Invoer klachten OK
ID	U2
Korte beschrijving	De OK medewerker vult het klachten formulier in.
Actor	OK medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: ”Meld klacht”. 2. Het systeem toont Ok klachten formulier. 3. De actor scant het net nummer of vult deze handmatig in. 4. De actor vult zijn naam in. 5. De actor klikt één categorie aan(beschadiging papier, vies, vermissing instrument, verkeerd instrument, verkeerd gemonteerd en overig). 6. De actor vult de afdeling in. 7. De actor drukt op opslaan. 8. Systeem toont data opgeslagen en toont klachten scherm
Postconditie	Het OK klachten formulier is ingevuld.
Alternatieve flows	2a. T-doc is niet beschikbaar. 5a. als de actor op beschadiging papier drukt wordt de keuze voorgelegd waar deze beschadiging is. 5b. als de actor op overig drukt krijgt hij een invoerveld waar hij de categorie kan invullen.

Invoer RIC controle

name	Invoer RIC controle
ID	U3
Korte beschrijving	De dag oudste op de CSA vult het RIC formulier in
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: “vul RIC controle in”. 2. Het systeem toont RIC formulier. 3. De actor selecteert “goed” bij net reinheid. 4. De actor selecteert “goed” bij instrumenten intact. 5. De actor selecteert “goed” bij net compleet. 6. De actor vult zijn naam in. 7. De actor vult naam van de gecontroleerde medewerker in. 8. De actor drukt op opslaan. 9. Systeem toont data opgeslagen en toont RIC scherm
Postconditie	De RIC controle formulier is ingevuld.
Alternatieve flows	3a. Bericht met dat net opnieuw schoongemaakt moet worden. 4a. Bericht dat net in reparatie moet. 5a. Bericht dat net in reparatie moet.

Invoer daglijst

name	Invoer daglijst
ID	U4
Korte beschrijving	De CSA medewerker vult de daglijst in.
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: “vul daglijst in”. 2. Het systeem toont de daglijsten. 3. De actor selecteert “nieuwe daglijst”. 4. Het systeem toont een lege daglijst. 5. De actor vult de daglijst in. 6. De actor drukt op opslaan. 7. Systeem toont data opgeslagen en toont daglijst scherm
Postconditie	De daglijst is ingevuld.
Alternatieve flows	-

Wijzig daglijst

name	Wijzig daglijst
ID	U5
Korte beschrijving	De CSA medewerker wijzigt de daglijst
Actor	CSA medewerker
Pre conditie	De daglijst moet bestaan
Scenario beschrijving	1. De actor selecteert: “Wijzig daglijst”. 2. Het systeem toont de daglijsten. 3. De actor selecteert één daglijst. 4. Het systeem toont de daglijst. 5. De actor wijzigt de daglijst. 6. De actor drukt op opslaan. 7. Systeem toont data opgeslagen en toont daglijst scherm
Postconditie	De daglijst is gewijzigd.
Alternatieve flows	-

Bekijk kamer Module

name	Bekijk kamer Module
ID	U6
Korte beschrijving	De actor ziet een informatie scherm die is gebaseerd op de kamer waar hij zich bevind
Actor	CSA medewerker, OK medewerker en Dagoudste
Pre conditie	None
Scenario beschrijving	1. Het systeem toont kamer module.
Postconditie	De actor heeft kamer informatie bekeken.
Alternatieve flows	-

Apparatuur checklist invullen

name	Apparatuur checklist invullen
ID	U8
Korte beschrijving	De CSA medewerker vult de apparatuur checklists in.
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: “apparaat checklist” 2. Het systeem toont een checklist. 3. De actor vult de checklist in. 4. De actor drukt op opslaan 5. Systeem toont data opgeslagen en toont checklist scherm
Postconditie	Checklist is ingevuld
Alternatieve flows	-

9.2 Use Case Stories

Invoer gevaarlijke situatie

Use-case Story	Use Case	Flow(s)
Invoer gevaarlijke situatie	Invoer gevaarlijke situatie	Basic flow
Invoer gevaarlijke situatie – niet beschikbaar	Invoer gevaarlijke situatie	Basic flow + 2a

Invoer klachten OK

Use-case Story	Use Case	Flow(s)
Invoer klachten OK	Invoer klachten OK	Basic flow
Invoer klachten OK – niet beschikbaar	Invoer klachten OK	Basic flow + 2a
Invoer klachten OK – papier	Invoer klachten OK	Basic flow + 5a
Invoer klachten OK – overig	Invoer klachten OK	Basic flow + 5b

Invoer daglijst

Use-case Story	Use Case	Flow(s)
Invoer daglijst	Invoer daglijst	Basic flow

Invoer RIC controle

Use-case Story	Use Case	Flow(s)
Invoer RIC controle	Invoer RIC controle	Basic flow
Invoer RIC controle – Opnieuw	Invoer RIC controle	Basic flow + 3a
Invoer RIC controle – Reparatie 1	Invoer RIC controle	Basic flow + 4a
Invoer RIC controle – Reparatie 2	Invoer RIC controle	Basic flow + 5a

Wijzig daglijst

Use-case Story	Use Case	Flow(s)
Wijzig daglijst	Wijzig daglijst	Basic flow

Bekijk kamer Module

Use-case Story	Use Case	Flow(s)
Bekijk kamer module	Bekijk kamer module	Basic flow

Apparatuur checklist invullen

Use-case Story	Use Case	Flow(s)
Apparatuur checklist invullen	Apparatuur checklist invullen	Basic flow

9.3 Use Case Slices

Invoer gevaarlijke situatie

Nummer	Use-case slice	Use-case Story	Test-case
1	Invoer gevaarlijke situatie	Invoer gevaarlijke situatie	Invoer één gevaarlijke situatie
2	Invoer gevaarlijke situatie - connectie probleem	Invoer gevaarlijke situatie – niet beschikbaar	Invoer één gevaarlijke situatie als data systeem niet beschikbaar is

Invoer klachten OK

Nummer	Use-case slice	Use-case Story	Test-case
3	Invoer klachten OK 1	Invoer klachten OK	Invoer één klachten formulier
4	Invoer klachten OK 2	Invoer klachten OK – papier Invoer klachten OK - overig	Invoer één klachten formulier met papier beschadiging categorie
			Invoer één klachten formulier met overig categorie
5	Invoer klachten OK - connectie probleem	Invoer klachten OK – niet beschikbaar	Invoer één klachten formulier als data systeem niet beschikbaar is

Invoer daglijst

Nummer	Use-case slice	Use-case Story	Test-case
6	Invoer daglijsten	Invoer daglijst	Invoer één daglijst

Invoer RIC controle

Nummer	Use-case slice	Use-case Story	Test-case
7	Invoer RIC controle	Invoer RIC controle	Invoer één RIC controle
8	Invoer RIC controle – Opnieuw	Invoer RIC controle – Opnieuw	Invoer één RIC controle met fout bij reinheid
9	Invoer RIC controle – Reparatie	Invoer RIC controle – Reparatie 1 Invoer RIC controle – Reparatie 2	Invoer één RIC controle met fout bij intact
			Invoer één RIC controle met fout bij compleet

Wijzig daglijst

Nummer	Use-case slice	Use-case Story	Test-case
10	Daglijst wijzigen	Wijzig daglijst	Wijzig één daglijst

Bekijk kamer Module

Nummer	Use-case slice	Use-case Story	Test-case
11	Kamer modules	Bekijk kamer module	Bekijk kamer module

Apparatuur checklist invullen

Nummer	Use-case slice	Use-case Story	Test-case
12	Checklist apparatuur	Apparatuur checklist invullen	Vul één checklist in

9.4 Uitvoering per sprint

Deze indeling is gemaakt per sprint welke use case slices er die sprint uitgevoerd zou worden.

Sprint	Beschrijving	Use case slice
0	Project opzet	-
1	Basis Cockpit	-
2	Database wijzigingen	-
3	Modules	11
4	Modules	11
5	Gevaarlijke situatie en klachten formulier	1, 2, 3, 4, 5
6	RIC checks	7, 8, 9
7	Daglijsten	6, 10

Bijlage D: Functioneel ontwerp

Functioneel ontwerp

Functioneel ontwerp voor de ontwikkeling van de CSA cockpit voor NewCompliance.

Shawn Steinz

6-8-2015

Versie 1.0

Inhoudsopgave

1. Inleiding 1
1.1 Doel 1
1.2 Opbouw 1
2. Algemene beschrijving 2
2.1 Use case 2
2.2 actoren 2
3. Beschrijving van functionaliteiten 3
3.1 Use case beschrijvingen 3
4. Schermontwerpen 6

1. Inleiding

1.1 Doel

Het functioneel ontwerp beschrijft de functies van de te ontwikkelen Cockpit. De functionaliteiten zijn opgesteld aan de hand van de requirements. Deze requirements zijn verkregen tijdens de gesprekken met de opdracht gever en het Erasmus MC. Deze requirements zijn terug te vinden in het requirements rapport. Het doel van dit document is het beschrijven van de gewenste functionaliteiten van de Cockpit.
Aan de hand van het functioneel- en technisch ontwerp zal de Cockpit worden ontwikkeld. Het functioneel ontwerp is onderdeel van een reeks documenten. Deze documenten vormen de basis van de Cockpit. De documenten in de Cockpit reeks zijn: requirements rapport, functioneel ontwerp, technisch ontwerp en testrapportage.

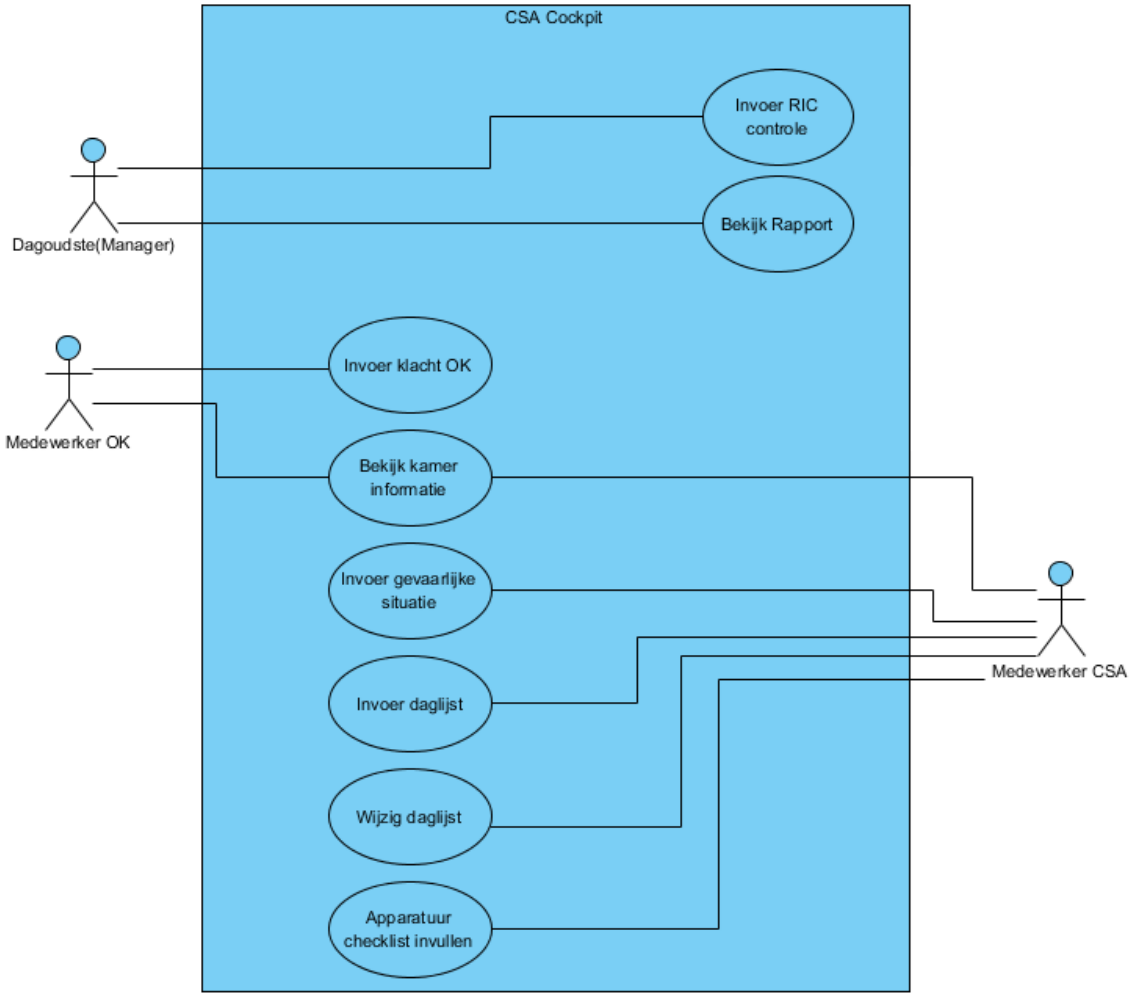
1.2 Opbouw

Het functioneel ontwerp is als volgt opgebouwd. Er wordt begonnen met alle opgestelde use cases. Deze zijn voortgekomen uit de opgestelde requirements. Hierna worden de betrokken actoren naar voren gebracht. In het volgende hoofdstuk worden de use case scenario's uitgewerkt. De uitgewerkte use cases zijn voortgekomen uit de geprioriteerde requirements. Het functioneel ontwerp sluit af met het tonen van de schermenontwerpen.

2. Algemene beschrijving

In dit hoofdstuk wordt een totaal overzicht geschetst van de functionaliteiten van CSA Cockpit. Dit wordt gedaan door alle use cases te tonen. Dit use case diagram is overeenstemmend met het use case diagram in het requirements rapport. Hierna worden de bijbehorende actoren met hun rechten binnen het systeem naar voren gebracht.

2.1 Use case



2.2 actoren

Actor	Role	Description
CSA Medewerker	User	De actor kan daglijsten invoeren en wijzigen ook kan hij gevaarlijke situatie formulier invoeren en ook kan hij de Dashbord module bekijken.
OK medewerker	User	De actor kan een OK klacht invoeren en hij kan de Dashbord module bekijken.

Dagoudste(Manager)	Superuser	De actor kan RIC controle invoeren en data bekijken in de analyse tool.

3. Beschrijving van functionaliteiten

In dit hoofdstuk zijn de use case beschrijvingen(scenario's) opgenomen. Enkel de beschrijvingen van de use cases die worden uitgewerkt zijn hier te vinden. Deze zijn bepaalt door de prioriteiten die de opdrachtgever heeft gesteld aan de requirements. Voor deze prioritering en compleet overzicht van de use cases zie het Requirements document.

3.1 Use case beschrijvingen

Invoer gevaarlijke situatie

name	Invoer gevaarlijke situatie
ID	U1
Korte beschrijving	De CSA medewerker vult het gevaarlijke situatie formulier in.
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: "Meld gevaarlijke situatie". 2. Het systeem toont gevaarlijke situatie formulier met de datum van die dag. 3. De actor scant het net nummer of vult deze handmatig in. 4. De actor vult de afdeling in. 5. De actor vult klacht in(mesje, naald of overig). 6. De actor vult zijn naam in. 7. De actor drukt op opslaan. 8. Systeem toont data opgeslagen en toont klachten scherm
Postconditie	Het gevaarlijke situatie formulier is ingevuld.
Alternatieve flows	2a. T-doc is niet beschikbaar.

Invoer klachten OK

name	Invoer klachten OK
ID	U2
Korte beschrijving	De OK medewerker vult het klachten formulier in.
Actor	OK medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: "Meld klacht". 2. Het systeem toont Ok klachten formulier. 3. De actor scant het net nummer of vult deze handmatig in. 4. De actor vult zijn naam in. 5. De actor klikt één categorie aan(beschadiging papier, vies, vermissing instrument, verkeerd instrument, verkeerd gemonteerd en overig). 6. De actor vult de afdeling in. 7. De actor drukt op opslaan. 8. Systeem toont data opgeslagen en toont klachten scherm
Postconditie	Het OK klachten formulier is ingevuld.
Alternatieve flows	2a. T-doc is niet beschikbaar. 5a. als de actor op beschadiging papier drukt wordt de keuze voorgelegd waar deze beschadiging is. 5b. als de actor op overig drukt krijgt hij een invoerveld waar hij de categorie kan invullen.

Invoer RIC controle

name	Invoer RIC controle
ID	U3
Korte beschrijving	De dag oudste op de CSA vult het RIC formulier in
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: “vul RIC controle in”. 2. Het systeem toont RIC formulier. 3. De actor selecteert “goed” bij net reinheid. 4. De actor selecteert “goed” bij instrumenten intact. 5. De actor selecteert “goed” bij net compleet. 6. De actor vult zijn naam in. 7. De actor vult naam van de gecontroleerde medewerker in. 8. De actor drukt op opslaan. 9. Systeem toont data opgeslagen en toont RIC scherm
Postconditie	De RIC controle formulier is ingevuld.
Alternatieve flows	3a. Bericht met dat net opnieuw schoongemaakt moet worden. 4a. Bericht dat net in reparatie moet. 5a. Bericht dat net in reparatie moet.

Invoer daglijst

name	Invoer daglijst
ID	U4
Korte beschrijving	De CSA medewerker vult de daglijst in.
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: “vul daglijst in”. 2. Het systeem toont de daglijsten. 3. De actor selecteert “nieuwe daglijst”. 4. Het systeem toont een lege daglijst. 5. De actor vult de daglijst in. 6. De actor drukt op opslaan. 7. Systeem toont data opgeslagen en toont daglijst scherm
Postconditie	De daglijst is ingevuld.
Alternatieve flows	-

Wijzig daglijst

name	Wijzig daglijst
ID	U5
Korte beschrijving	De CSA medewerker wijzigt de daglijst
Actor	CSA medewerker
Pre conditie	De daglijst moet bestaan
Scenario beschrijving	1. De actor selecteert: “Wijzig daglijst”. 2. Het systeem toont de daglijsten. 3. De actor selecteert één daglijst. 4. Het systeem toont de daglijst. 5. De actor wijzigt de daglijst. 6. De actor drukt op opslaan. 7. Systeem toont data opgeslagen en toont daglijst scherm
Postconditie	De daglijst is gewijzigd.
Alternatieve flows	-

Bekijk kamer Module

name	Bekijk kamer Module
ID	U6
Korte beschrijving	De actor ziet een informatie scherm die is gebaseerd op de kamer waar hij zich bevind
Actor	CSA medewerker, OK medewerker en Dagoudste
Pre conditie	None
Scenario beschrijving	1. Het systeem toont kamer module.
Postconditie	De actor heeft kamer informatie bekeken.
Alternatieve flows	-

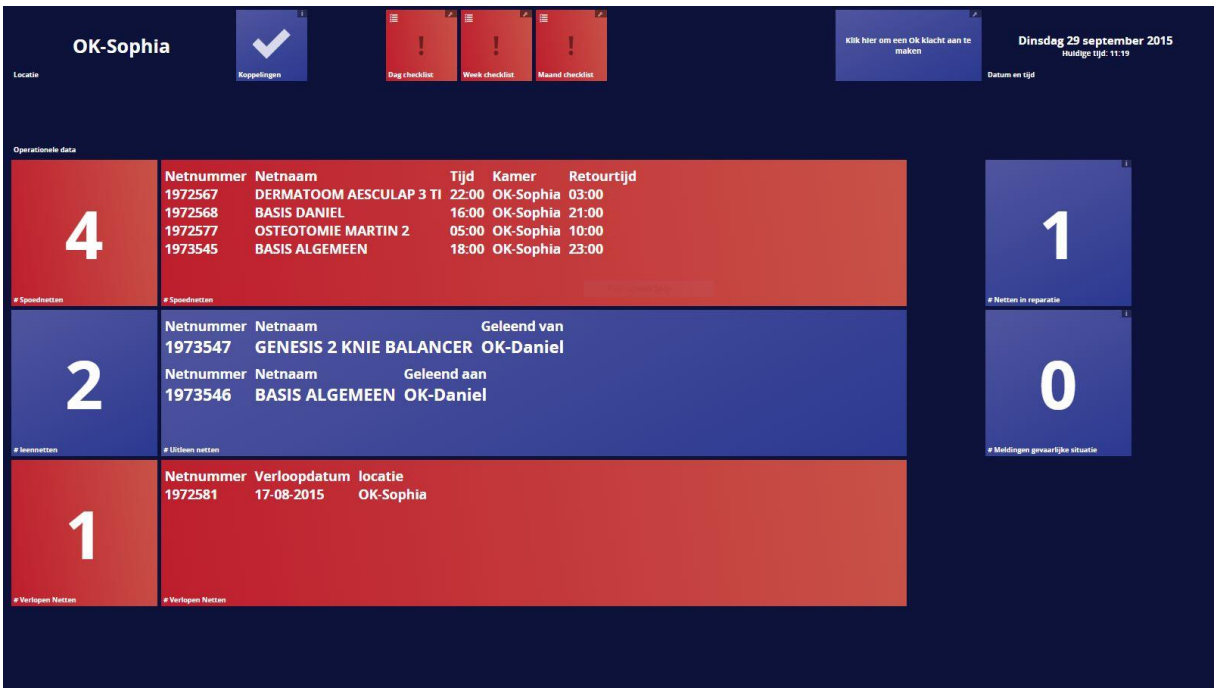
Apparatuur checklist invullen

name	Apparatuur checklist invullen
ID	U8
Korte beschrijving	De CSA medewerker vult de apparatuur checklists in.
Actor	CSA medewerker
Pre conditie	None
Scenario beschrijving	1. De actor selecteert: “apparaat checklist” 2. Het systeem toont een checklist. 3. De actor vult de checklist in. 4. De actor drukt op opslaan 5. Systeem toont data opgeslagen en toont checklist scherm
Postconditie	Checklist is ingevuld
Alternatieve flows	-

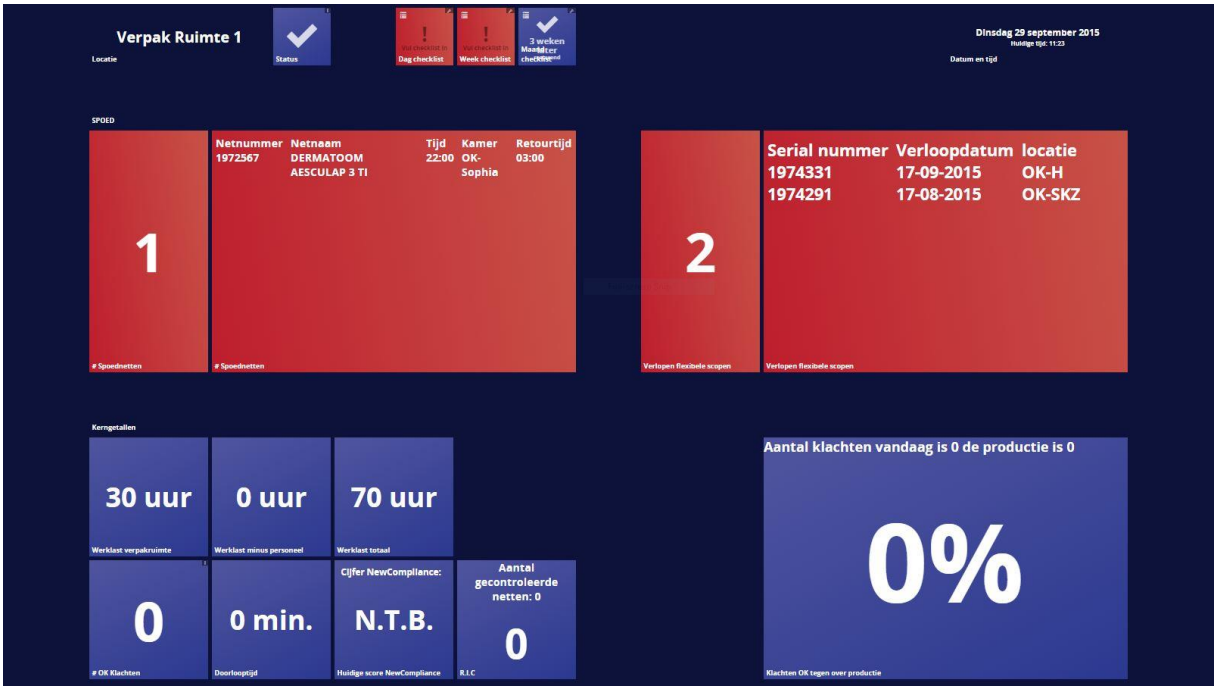
4. Schermontwerpen

Dagoudste scherm

OK distributie scherm



Verpak ruimte 1 scherm



Verpak ruimte 2 scherm

Vuile ruimte scherm

Bijlage E: Technisch ontwerp

Technisch ontwerp

Technisch ontwerp voor de ontwikkeling van de CSA cockpit voor NewCompliance.

Shawn Steinz

30-9-2015

Versie 1.0

Inhoudsopgave

1. Inleiding 1
1.1 Doel 1
1.2 Opbouw 1
2. Laravel 2
3. MED Cockpit 3
3.1 Klassendiagram 3
3.2 Package diagram 4
3.3 ERD 5

1. Inleiding

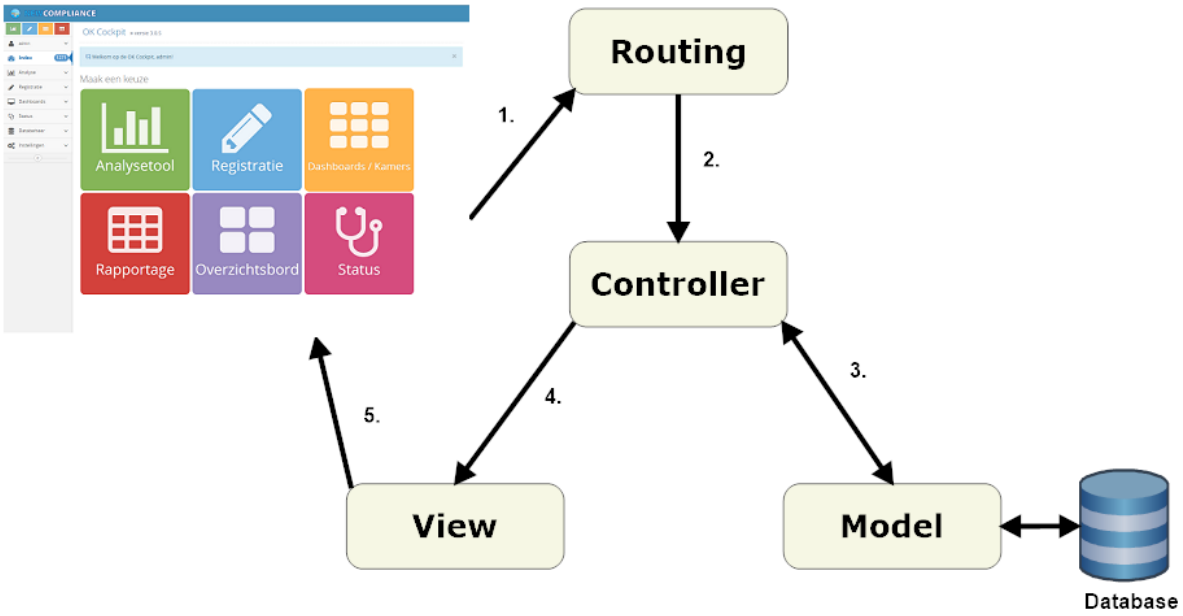
1.1 Doel

Het technisch ontwerp beschrijft de technische beslissingen van de te ontwikkelen Cockpit. Dit document beschrijft op technisch niveau de applicatie. Het doel van dit document is om een technische basis te leggen van de Cockpit. Het technisch ontwerp dient als basis tot het ontwikkelen van de Cockpit. Het technisch ontwerp is een onderdeel van een reeks documenten. Deze documenten reeks vormt de basis tot het ontwikkelen van Cockpit. Het requirements document en het functioneel ontwerp zijn de voorgaande documenten uit deze reeks.

1.2 Opbouw

Het technisch ontwerp is als volgt opgebouwd. Er wordt begonnen met het beschrijfen van het laravel framework. Vervolgens worden de gemaakte UML ontwerpen voor de CSA uitbreidingen van de Cockpit getoont. Het functioneel en technisch ontwerp zijn de basis documenten waarop de ontwikkeling van de tool zal plaatsvinden. Verder zullen de ontwerpen als basis dienen voor het testproces en rapport.

2. Laravel

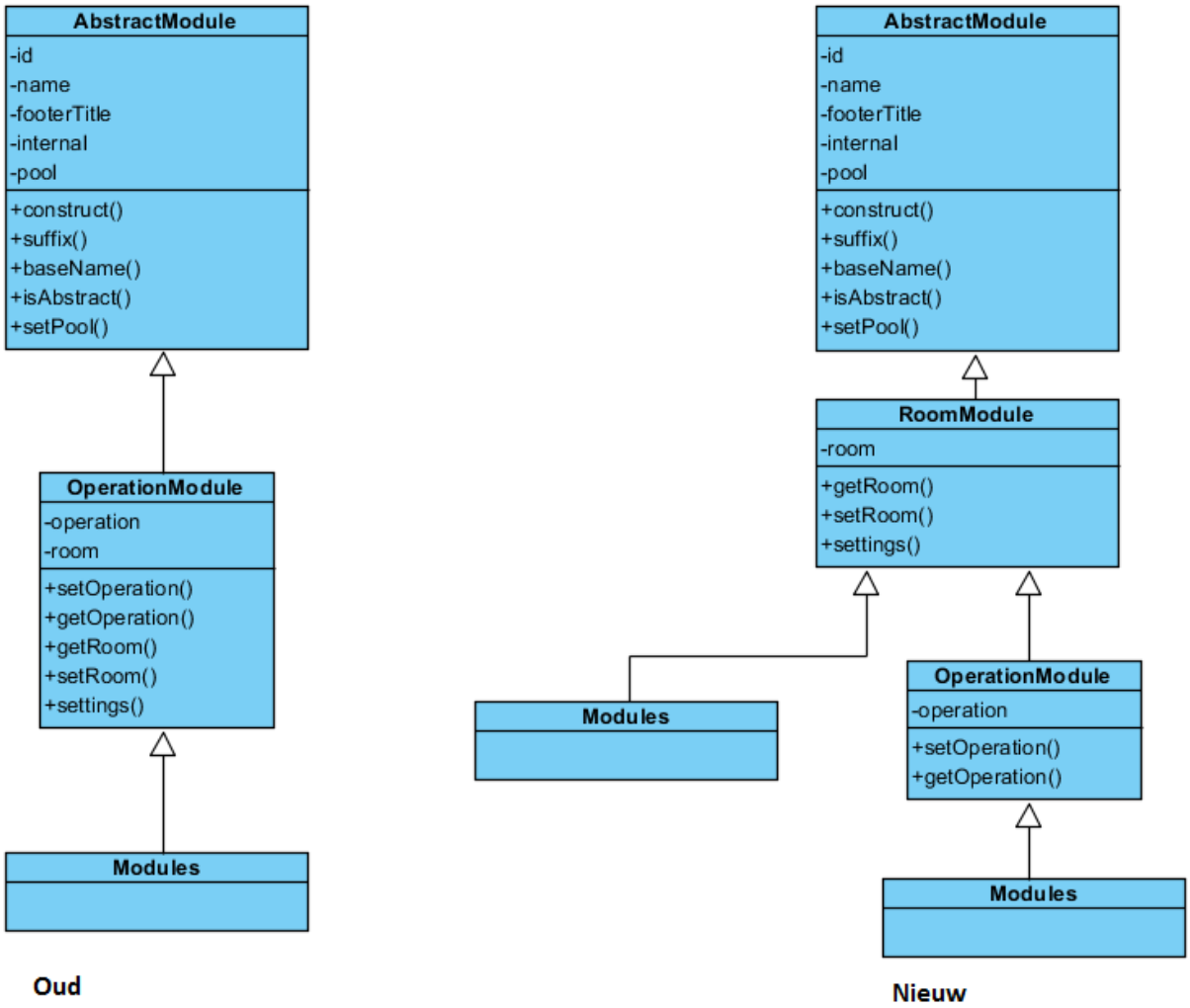


Stap 1: de user vraagt via de browser een pagina aan.
Stap 2: De routing roept de juiste controller aan. Laravel maakt gebruik van een Routing file om dit gemakkelijk te kunnen bewerkstelligen.
Stap 3: de controller heeft interactie met de model. Hier wordt met behulp van eloquent mapping de data uit de database gehaald. Vervolgens wordt in de aangeroepen model class de data getransformeerd om klaar gezet te worden voor de view.
Stap 4: de Controller roept de view op die bij de user aanvraag hoort en geeft de data mee.
Stap 5: de View wordt opgebouwd met de data en getoond in de browser.

3. MED Cockpit

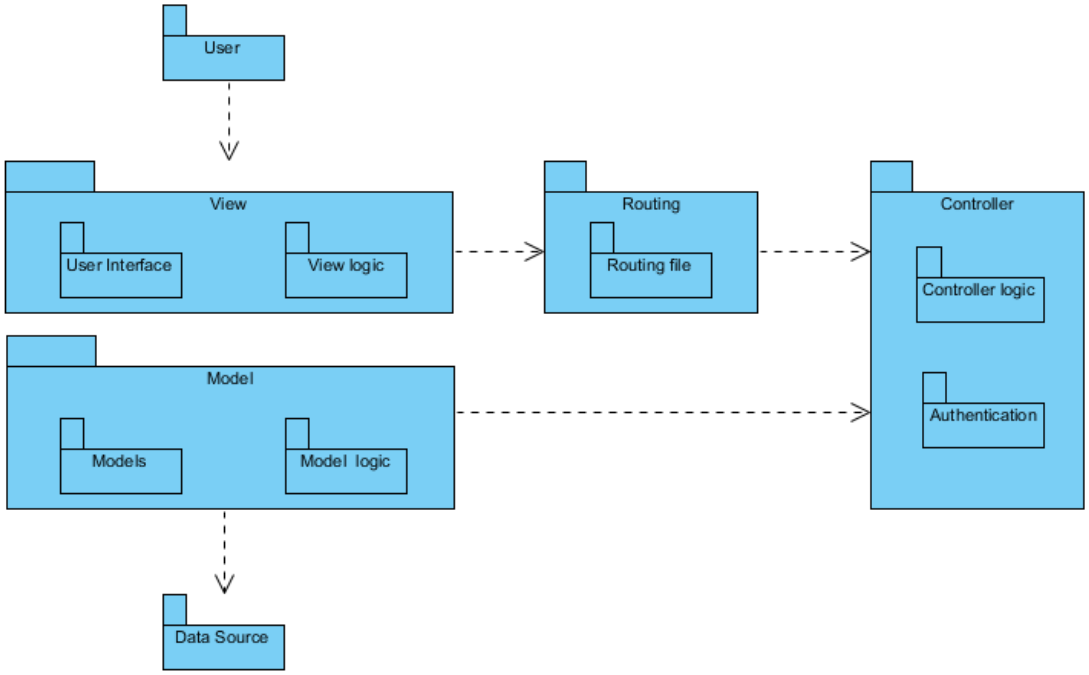
3.1 Klassendiagram

De software gebruikte voor elke module een operatieparameter. Voor de CSA Cockpit wordt er geen gebruik gemaakt van de operatieparameter. Daarom heb ik uit de Operation Module de kamerparameter gehaald en deze los gezet in Room Module. De Operation Module overerft nu de Room parameter vanuit de Room Module. Modules die geen gebruik maken van de operatieparameter kunnen nu Room Module overerven. Voor de modules die wel de operatieparameter nodig hebben kunnen Operation Module overerven. Met deze wijziging blijven de oude modules werken en kunnen er ook nieuwe modules gemaakt worden die geen operatieparameter bezitten. Dit is één van de plekken in de software waar deze wijziging door gevoerd moest worden. Ik heb onderstaande oude en nieuwe klassendiagram gemaakt om een duidelijk beeld te scheppen van deze wijziging:



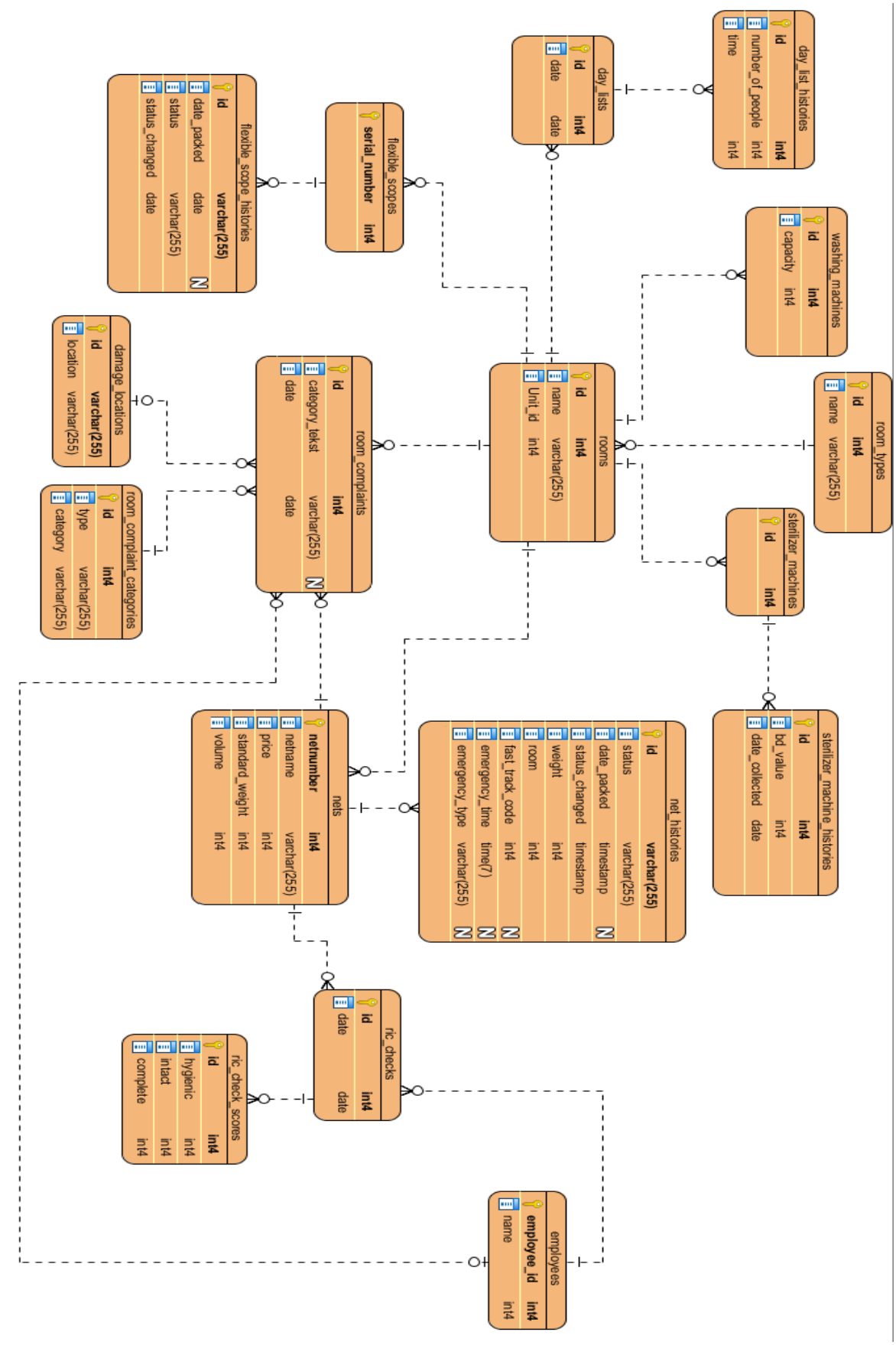
3.2 Package diagram

Het onderstaande package diagram toont de indeling van de Cockpit.



3.3 ERD

Het onderstaande ERD diagram toont de tables die aan de database toegevoegd moeten worden.



Bijlage F: Testrapport

Testrapport

shawn steinz

24-9-2015

Versie 1.0

Inhoudsopgave

1. Opdrachtformulering 2
1.1 Trajectnaam 2
1.2 Beoogd resultaat 2
1.3 Opdrachtgever 2
1.4 Opdrachtnemer 2
1.5 Testsoorten 2
2 Testbasis 3
2.1 Documenten 3
2.2 Applicatie 3
2.3 Testware 3
3 Teststrategie 4
3.1 Omschrijving testaanpak 4
3.2 Gebruikte testtechnieken en benodigdheden 4
3.2.1 Use Case test 4
3.2.2 CRUD tests 4
3.2.3 Unit tests 4
4. Testrapport 5
4.1 Use Case tests 5
4.1.1 Logische testcases 5
4.1.2 Fysieke testcases 10
4.2 CRUD test 15
4.2.1 CRUD matrix 15
4.2.2 Testcases 15

1. Opdrachtformulering

1.1 Trajectnaam

CSA Cockpit

1.2 Beoogd resultaat

De eerste doelstelling is het aanpassen van de OK Cockpit. De OK Cockpit moet niet meer afhankelijk zijn van het uitvoeren van een operatie zoals hierboven beschreven is. De tweede doestelling is het toevoegen van functionaliteiten die samen met de CSA van het Erasmus MC vastgesteld zullen worden. De eerste doelstelling moet volbracht zijn om de tweede uit te kunnen voeren.

1.3 Opdrachtgever

NewCompliance

1.4 Opdrachtnemer

Shawn Steinz

1.5 Testsoorten

Logische en fysieke testcases, unit testen en CRUD testen.

2 Testbasis

In dit hoofdstuk worden de documenten en software waarop de tests worden uitgevoerd naar voren gebracht. Dit wordt gedaan door eerst de documenten die als basis dienen naar voren te brengen. Deze documenten beschrijven de applicatie op functioneel en technisch niveau.

2.1 Documenten

Documentnaam
Requirements rapport
Functioneel ontwerp
Technisch ontwerp

2.2 Applicatie

Applicatienaam	Opmerking
MED Cockpit	Deze bevat interne PHPunit tests

2.3 Testware

Testware	Opmerking
PHPunit tests	Geïntrigeerd in de MED Cockpit

3 Teststrategie

3.1 Omschrijving testaanpak

Voor de gebruikers is het van belang dat de use cases, naar wens werken. Hiervoor moeten de use cases worden gewaarborgd.

3.2 Gebruikte testtechnieken en benodigdheden

De focus van het tests licht voornamelijk op bruikbaarheid en functionaliteit. Om deze kwaliteitsattributen te kunnen testen worden de CRUD, unitests en een use case test uitgevoerd. Hiermee worden de kwaliteitsattributen: bruikbaarheid en functionaliteit afgedekt

3.2.1 Use Case test

Het is belangrijk om de functionaliteiten te testen. Om de functionaliteit vast te stellen wordt er gebruik gemaakt van een Use Case test. Met deze use case test worden de use cases van de applicatie logisch en fysiek uitgewerkt. De use cases worden doorlopen door de tester, Shawn Steinz. Hierbij wordt van iedere use case stap bekeken of het verwachte resultaat naar voren komt.

3.2.2 CRUD tests

In de CRUD test wordt gekeken of de Create, Read, Update en Delete werken van een object.

3.2.3 Unit tests

In de Cockpit zijn unit tests aanwezig. Deze unit tests worden uitgevoerd en moeten voor de volle 100% slagen. Ook unit tests uit voorgaande releases moeten worden uitgevoerd als onderdeel

4. Testrapport

4.1 Use Case tests

4.1.1 Logische testcases

LOGISCHE CASE Invoer één gevaarlijke situatie	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘vul gevaarlijke situatie formulier in’	Systeem toont het gevaarlijke situatie formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.
Medewerker selecteert de radio button ‘ mesje’ bij categorie	Systeem vinkt de radio button aan.
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.
Medewerker drukt op opslaan	Systeem geeft aan dat het gevaarlijke situatie formulier is opgeslagen.

LOGISCHE CASE Invoer één gevaarlijke situatie als data systeem niet beschikbaar is	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘vul gevaarlijke situatie formulier in’	Systeem toont melding dat op dit moment niet mogelijk is om een gevaarlijke situatie formulier in te vullen.

LOGISCHE CASE Invoer één klachten formulier	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘vul klachten formulier in’	Systeem toont het klachten formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.
Medewerker selecteert de radio button ‘ mesje’ bij categorie	Systeem vinkt de radio button aan.
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.
Medewerker drukt op opslaan	Systeem geeft aan dat het klachten formulier is opgeslagen.

LOGISCHE CASE Invoer één klachten formulier met papier beschadiging categorie	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘vul klachten formulier in’	Systeem toont het klachten formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.
Medewerker selecteert de radio button ‘papier beschadiging’ bij categorie	Systeem toont een reeks aan radio buttons waar de locatie aangegeven kan worden.
Medewerker selecteert de radio button ‘links’ bij locatie	
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.
Medewerker drukt op opslaan	Systeem geeft aan dat het klachten formulier is opgeslagen.

LOGISCHE CASE Invoer één klachten formulier met overig categorie	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘vul klachten formulier in’	Systeem toont het klachten formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.
Medewerker selecteert de radio button ‘overig bij categorie	Systeem toont een uitleg invoerveld.
Medewerker vult een uitleg in	
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.
Medewerker drukt op opslaan	Systeem geeft aan dat het klachten formulier is opgeslagen.

LOGISCHE CASE Invoer één klachten formulier als data systeem niet beschikbaar is	
Main flow / actie	**Main flow / actie**
Medewerker selecteert de knop ‘vul klachten formulier in’	Systeem toont melding dat op dit moment niet mogelijk is om een klachten formulier in te vullen.

LOGISCHE CASE Invoer één daglijst	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop 'daglijsten'	Systeem toont het de daglijsten pagina.
Medewerker selecteerd de datum in het dropdown menu	Systeem toont de datum in het daarvoor bestemde veld.
Medewerker vult het aantal medewerkers in.	
Medewerker drukt op opslaan	Systeem geeft aan dat de daglijst is opgeslagen.

LOGISCHE CASE Invoer één RIC controle	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop 'RIC controle'	Systeem toont het RIC invul formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.
Medewerker zet de button van reinheid op Goed	
Medewerker zet de button van intact op Goed	
Medewerker zet de button van compleet op Goed	
Medewerker drukt op opslaan	Systeem geeft aan dat de RIC controle is opgeslagen.

LOGISCHE CASE Invoer één RIC controle met fout bij reinheid	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop 'RIC controle '	Systeem toont het RIC invul formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.
Medewerker zet de button van reinheid op Fout	Systeem geeft melding dat het net opnieuw gewassen moet worden

LOGISCHE CASE Invoer één RIC controle met fout bij intact	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop 'RIC controle '	Systeem toont het RIC invul formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.
Medewerker zet de button van reinheid op Goed	
Medewerker zet de button van intact op Fout	Systeem geeft melding dat het net opnieuw in reparatie moet

LOGISCHE CASE Invoer één RIC controle met fout bij compleet	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop 'RIC controle'	Systeem toont het RIC invul formulier.
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.
Medewerker zet de button van reinheid op Goed	
Medewerker zet de button van intact op Goed	
Medewerker zet de button van compleet op Fout	Systeem geeft melding dat het net opnieuw in reparatie moet

LOGISCHE CASE Wijzig één daglijst	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop 'daglijsten'	Systeem toont het de daglijsten pagina.
Medewerker selecteerd de datum in het dropdown menu	Systeem toont de datum in het daarvoor bestemde veld.
Medewerker verandert het aantal medewerkers.	
Medewerker drukt op opslaan	Systeem geeft aan dat de daglijst is gewijzigd.

LOGISCHE CASE Kamer bekijken	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘OK-2’ met layout ‘operatiekamer’	Systeem toont Dasboard van OK-2 met de layout van een operatiekamer met functionerende modules.

LOGISCHE CASE type toewijzen	
Main flow / actie	**gewenste resultaat**
Medewerker selecteert de knop ‘Layouts’	Systeem toont layouts pagina.
Medewerker selecteert het drop down menu kamer type en selecteert ‘operatiekamer’.	Systeem toon bij type ‘operatiekamer’
Medewerker klikt op de knop ‘opslaan’	Systeem slaat de layout wijziging op.

4.1.2 Fysieke testcases

FYSIEKE TEST Invoer één gevaarlijke situatie			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop ‘vul gevaarlijke situatie formulier in’	Systeem toont het gevaarlijke situatie formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.	√	
Medewerker selecteert de radio button ‘ mesje’ bij categorie	Systeem vinkt de radio button aan.	√	
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.	√	
Medewerker drukt op opslaan	Systeem geeft aan dat het gevaarlijke situatie formulier is opgeslagen.	√	

FYSIEKE TEST Invoer één gevaarlijke situatie als data systeem niet beschikbaar is			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop ‘vul gevaarlijke situatie formulier in’	Systeem toont melding dat op dit moment niet mogelijk is om een gevaarlijke situatie formulier in te vullen.	√	

FYSIEKE TEST Invoer één klachten formulier			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop ‘vul klachten formulier in’	Systeem toont het klachten formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.	√	
Medewerker selecteert de radio button ‘ mesje’ bij categorie	Systeem vinkt de radio button aan.	√	
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.	√	
Medewerker drukt op opslaan	Systeem geeft aan dat het klachten formulier is opgeslagen.	√	

FYSIEKE TEST Invoer één klachten formulier met papier beschadiging categorie			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'vul klachten formulier in'	Systeem toont het klachten formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.	√	
Medewerker selecteert de radio button 'papier beschadiging' bij categorie	Systeem toont een reeks aan radio buttons waar de locatie aangegeven kan worden.	√	
Medewerker selecteert de radio button 'links' bij locatie		√	
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.	√	
Medewerker drukt op opslaan	Systeem geeft aan dat het klachten formulier is opgeslagen.		

FYSIEKE TEST Invoer één klachten formulier met overig categorie			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'vul klachten formulier in'	Systeem toont het klachten formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker vult het net nummer in	Systeem controleert of netnummer bestaat in de database.	√	
Medewerker selecteert de radio button 'overig bij categorie	Systeem toont een uitleg invoerveld.	√	
Medewerker vult een uitleg in		√	
Medewerker selecteerd een kamer in het dropdown menu	Systeem toont de kamer naam in het daarvoor bestemde veld.	√	
Medewerker drukt op opslaan	Systeem geeft aan dat het klachten formulier is opgeslagen.		

FYSIEKE TEST Invoer één klachten formulier als data systeem niet beschikbaar is			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'vul klachten formulier in'	Systeem toont melding dat op dit moment niet mogelijk is om een klachten formulier in te vullen.	√	

FYSIEKE TEST Invoer één daglijst			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'daglijsten'	Systeem toont het de daglijsten pagina.	√	
Medewerker selecteerd de datum in het dropdown menu	Systeem toont de datum in het daarvoor bestemde veld.	√	
Medewerker vult het aantal medewerkers in.		√	
Medewerker drukt op opslaan	Systeem geeft aan dat de daglijst is opgeslagen.	√	

FYSIEKE TEST Invoer één RIC controle			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'RIC controle'	Systeem toont het RIC invul formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.	√	
Medewerker zet de button van reinheid op Goed		√	
Medewerker zet de button van intact op Goed		√	
Medewerker zet de button van compleet op Goed		√	
Medewerker drukt op opslaan	Systeem geeft aan dat de RIC controle is opgeslagen.		

FYSIEKE TEST Invoer één RIC controle met fout bij reinheid			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'RIC controle '	Systeem toont het RIC invul formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.	√	
Medewerker zet de button van reinheid op Fout	Systeem geeft melding dat het net opnieuw gewassen moet worden	√	

FYSIEKE TEST Invoer één RIC controle met fout bij intact			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'RIC controle '	Systeem toont het RIC invul formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.	√	
Medewerker zet de button van reinheid op Goed		√	
Medewerker zet de button van intact op Fout	Systeem geeft melding dat het net opnieuw in reparatie moet	√	

FYSIEKE TEST Invoer één RIC controle met fout bij compleet			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'RIC controle'	Systeem toont het RIC invul formulier.	√	
Medewerker selecteerd zijn naam in het dropdown menu	Systeem toont werknemers naam in het daarvoor bestemde veld.	√	
Medewerker selecteerd de naam in het dropdown menu van de persoon die hij aan het controleren is	Systeem toont de naam in het daarvoor bestemde veld.	√	
Medewerker zet de button van reinheid op Goed		√	
Medewerker zet de button van intact op Goed		√	
Medewerker zet de button van compleet op Fout	Systeem geeft melding dat het net opnieuw in reparatie moet		

FYSIEKE TEST Wijzig één daglijst			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'daglijsten'	Systeem toont het de daglijsten pagina.	√	
Medewerker selecteerd de datum in het dropdown menu	Systeem toont de datum in het daarvoor bestemde veld.	√	
Medewerker verandert het aantal medewerkers.		√	
Medewerker drukt op opslaan	Systeem geeft aan dat de daglijst is gewijzigd.	√	

FYSIEKE TEST Kamer bekijken			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'OK-2' met layout 'operatiekamer'	Systeem toont Dasboard van OK-2 met de layout van een operatiekamer met functionerende modules.	√	

FYSIEKE TEST type toewijzen			
Main flow / actie	**gewenste resultaat**	**werkelijke resultaat**	**opmerkingen**
Medewerker selecteert de knop 'Layouts'	Systeem toont layouts pagina.	√	
Medewerker selecteert het drop down menu kamer type en selecteert 'operatiekamer'.	Systeem toon bij type 'operatiekamer'	X	Toonden geen types
Medewerker klikt op de knop 'opslaan'	Systeem slaat de layout wijziging op.	√	

4.2 CRUD test

4.2.1 CRUD matrix

Attribuut	Create	Read	Update	Delete
Net		X	X	
FlexibleScope		X	X	

4.2.2 Testcases

	Geval 1	Geval 2
	Net	FlexibleScope
getNet()	R	
editNet()	RU	
getFlexibleScope ()		R
EditFlexibleScope ()		RU

Bijlage G: Trello bord

Vuile ruimte

Hardware: non touch scherm nu 20inch (groter?)

Per OK-Complex (aantal 5)

Bron: Harmony/NC? T-DOC

[i] # Flex scopen
2 1 16/18

[s] Netnaam, net nummer, #netten met afwijkend gewicht
2 1 17/18

[s] # Afwijkend gewicht (samengevoegd met netnaam, netnummer)
1 0/18

Capaciteit apparatuur tov volume eenheden in aantocht voor wasmachines.
1 0/18

[a] Spoed net, weergave op net naam
1 1 17/18

[a] Spoed plan, weergave op aantal
2 1 0/18

Add a card...

Verpakruimte 1

Bron: T-DOC 3M

Harmony NewCompliance

Display non touch 42inch

[i] Verlopen eenheden flex scopen
1 12/18

[i] Klacht categorie + net data
1 0/18

[a] Spoed net weergave op net naam(zelfde als vuile ruimte + extra module voor vuile ruimte.)
1 1 0/18

[a] Spoed plan (vervalt?)
1 0/18

[s] # en % Klachten (t.o.v. productie)
1 17/18

[s] RIC controle 100% correct
1 18/18

[alg] Norm doorlooptijd
1 17/18

[alg] Werklast verpakruimte
1 0/18

[alg] Werklast minus # personeel verpakruimte
2 1 0/18

Add a card...

Verpakruimte 2

Bron: T-DOC 3M

Harmony NewCompliance

Touchscreen 22inch (groter?)

[i] Netten in reparatie (# / serie, welke netten?)
1 17/18

[c] Dagelijkse checklist

[c] Wekelijkse checklist apparatuur.
1 1/18

[c] Maandelijkse checklist

[c] Invoer: RIC-controle
1 1 18/18

[a] BD-waarde (nieuwe software vanuit 3M later bespreken)
1 0/18

[s] Verwachte bezetting medewerkers
1 1 0/18

Capaciteit apparatuur tov volume eenheden in aantocht voor wasmachines.
1 0/18

Add a card...

Uitgifte CSA

Display non touch 42 inch

Per OK Complex (5)

[i] Flex scopen
1 0/18

[i] Netnaam + nummer
1 0/18

[a] SPOED plan
1 0/18

[a] SPOED net
1 0/18

Add a card...

Dagoudste

Bron: T-DOC 3M

Harmony NewCompliance

Werkstation

[i] Verlopen eenheden (1 week voor)
1 0/18

[a] BD-waarde (3M bij nodig)
1 0/18

[a] 1 of meer klacht OK
1 0/18

[c] Checklisten (om te zien of alle 3 de checklisten van verpakruimte 2 zijn ingevuld)
0/18

[s] 1 of meer gevaarlijke situaties
1 0/18

[s] RIC controle
0/18

[s] # Afwijk gewicht (OK, vuile ruimte en verpakruimte)
1 0/18

[alg] Werklast (alle 3 de variates)
1 0/18

[alg] Norm doorlooptijd
1 0/18

Add a card...

OK (overzicht per OK locatie)

[i] Verlopen eenheden (1 week van tevoren)
1 0/18

[i] Netten uitgeleend
1 0/18

[i [Netten reparaties
0/18

[a] SPOED plan
1 0/18

[a] SPOED net: OK wil graag weten wanneer aankomsttijd van net op OK
1 0/18

[s] Gevaarlijke situaties
0/18

[c] Op OK link in t-doc naar NC invoer: invoerveld klachten
1 1 18/18

Add a card...

- Algorithme
- Invoerveld
- Signalering
- Alarmering
- Checklist/interactief
- Informatief
- 3M
- T-doc
- Harmony