Afstudeerverslag

Optimalisering en automatisering van interne foutafhandeling procedures

De integratie van een module waarin fouten op websites worden afgehandeld en getraceerd naar de oorsprong

Geschreven door: E.Soek
Studentnummer: 11004010

Soort document: Afstudeerverslag
Examinator 1: A.M.J.J. Lousberg – Orbons
Examinator 2: E.M. van Doorn

Onderwijsinstelling: De Haagse Hogeschool
Faculteit: IT&Design
Opleiding: Informatica

In uitvoering voor: Floro Webdevelopment B.V.

Rotterdam, februari 2016

Versie 1.0

://FLORO

DE HAAGSE HOGESCHOOL

Optimalisering en automatisering van interne foutafhandeling procedures

De integratie van een module waarin fouten op websites worden afgehandeld en getraceerd naar de oorsprong

Geschreven door: E.Soek

Referaat

Soek, Eva

Optimalisering en automatisering van interne foutafhandeling procedures. De integratie van een module waarin fouten op websites worden afgehandeld en getraceerd naar de oorsprong.
Rotterdam, Floro Webdevelopment B.V., 2016

Afstudeerverslag van Eva Soek, geschreven in het kader van het afstuderen bij de opleiding Informatica aan de academie IT&Design aan De Haagse Hogeschool.

Het verslag behandelt het proces dat is doorlopen tijdens de afstudeerperiode van Eva Soek bij Floro Webdevelopment B.V. te Rotterdam. Deze opdracht stond in het teken van het ontwerpen, bouwen en testen van een proof of concept. Dit proof of concept automatiseert en verbetert de huidige foutafhandeling in de websites van het bedrijf en breidt dit uit door het bieden van een overzicht en traceerbaarheid. Er is een toolselectie uitgevoerd voor het selecteren van tools die geïntegreerd zullen worden in het proof of concept. Deze tools bieden een framework voor het schrijven van testscripts en zullen fouten op websites achterhalen.

Descriptoren:
- Afstudeeropdracht
- PHP
- CRM
- CMS
- Toolselectie
- Module integratie
- Exception afhandeling
- Exception tracering
- Framework
- Git
- Proof of concept

Voorwoord

De opdracht voor dit rapport ‘Optimalisering en automatisering van interne foutafhandeling procedures’ is uitgevoerd in opdracht van Floro Webdevelopment B.V.

Dit rapport is tot stand gekomen in het kader van mijn afstuderen aan de opleiding Informatica op De Haagse Hogeschool. Van februari 2016 tot en met juni 2016 ben ik bezig geweest met het uitvoeren van de opdracht en het schrijven van dit rapport.

Bij deze wil ik graag mijn bedrijfsbegeleider bedanken voor het bieden van de kans om bij Floro te mogen afstuderen en mij hierin te begeleiden. Daarnaast wil ik mijn stagebegeleider bedanken voor de fijne samenwerking en de begeleiding tijdens de afstudeerperiode.

Verder wil ik graag mijn collega’s bedanken voor de gezellige sfeer op kantoor en de fijne samenwerking. Als ik vragen had kon ik altijd bij hen terecht. Ten slotte wil ik mijn medestudenten, ouders en vriend bedanken voor de steun tijdens de afstudeerperiode.

Eva Soek

Rotterdam, 2016

Inhoudsopgave

1 Inleiding

In opdracht van Floro Webdevelopment B.V. zal ik een proof of concept ontwikkelen van een module die zich richt op het geautomatiseerd testen van websites. Deze module optimaliseert het bedrijfsproces waarin fouten worden afgehandeld op websites die door het bedrijf ontwikkeld zijn.

Voor de opdracht zal ik een proof of concept maken, een plan van aanpak opstellen, requirements definiëren en zal ik een toolselectie uitvoeren.

Later in het proces is er een herziening van de opdracht. Dit resulteert in een nieuw doel voor de module en het omgaan met wijzigende en nieuwe requirements. De aanleiding voor de nieuwe opdracht is het vertrek van mijn begeleider en de toewijzing van een nieuwe begeleider met andere wensen en eisen voor de te bouwen module.

De nieuwe opdracht omvat het verbeteren van het bedrijfsproces waarin fouten op websites worden afgehandeld. In de begin situatie is dit proces niet optimaal. De fouten worden opgeslagen en afgehandeld als supporttickets waardoor er verminderd overzicht in de support is. Daarnaast is er geen tracering van de fouten naar versiebeheer en zijn de fouten niet beheerbaar.

Voor de nieuwe opdracht zal ik het huidige proces in kaart brengen en zal ik de aanpassingen en toevoegingen bepalen om dit proces te verbeteren. Het verdere verloop van de opdracht, zoals de vastgestelde iteraties zullen betrekking hebben op de nieuwe opdracht.

Ik zal een proof of concept bouwen die het afhandelen van fouten zal encapsuleren en daarnaast zal ik een onderdeel maken in het Customer Relation Management (CRM) systeem van het bedrijf. Dit is het systeem waarin alle interne zaken worden geregeld. In dit onderdeel zal ik de fouten overzichtelijk en beheerbaar maken.

2 Context

In dit hoofdstuk komt de context van de opdracht aan de orde. Er wordt een beschrijving gegeven van de organisatie, de werkomgeving en de cultuur van het bedrijf.

2.1 De organisatie

De opdracht wordt uitgevoerd in opdracht van Floro Webdevelopment B.V. Floro is een relatief klein bedrijf met circa 25 werknemers. Het bedrijf richt zich op webdevelopment en cloudoplossingen.

Het bedrijf ontwikkelt websites en webshops die gebruikmaken van een intern ontwikkeld framework en Content Management Systeem (CMS). Het CMS is de omgeving waarin content van een website beheerd kan worden via een user interface. Het framework biedt standaard functionaliteit voor websites dat herbruikbaarheid, modularisatie en generalisatie promoot.

2.2 De werkomgeving

Floro Webdevelopment B.V. is gevestigd in het centrum van Rotterdam. Het kantoor bestaat uit twee grote ruimtes met werkplekken en een vergaderruimte. Daarnaast zijn er gedeelde toiletten en een keuken met de andere bedrijven op de verdieping.

Iedere werkplek heeft beschikking tot een PC en één of twee schermen en geeft de gelegenheid om je eigen laptop aan te sluiten op de schermen.

Er zijn geen vaste regels wat betreft ontwikkelsoftware waar de werknemers gebruik van moeten maken. Wel wordt er gebruik gemaakt van versiebeheer via Git. De Git repositories worden beheerd via Gitlab.

Daarnaast wordt er gewerkt met OTAP-omgevingen. OTAP staat voor Ontwikkel, Test, Acceptatie en Productie. Dit zijn verschillende omgevingen die gebruikt worden voor de ontwikkeling van software.

Verder heeft iedere werknemer heeft een eigen RVM (Remote Virtual Machine) ontwikkelomgeving wat binnen het bedrijf een ‘development container’ wordt genoemd. Deze container neemt de plaats in van een lokale ontwikkelomgeving op een PC of laptop.

2.3 De cultuur

Het bedrijf hanteert een informele cultuur. Er zijn korte lijnen met collega’s. Het bedrijf bestaat uit het management, accountmanagers, projectmanagers, backend developers, frontend developers, designers en interactief vormgevers. Het bedrijf heeft meerdere stagiaires in dienst.

Er is de mogelijkheid om gezamenlijk te lunchen met collega’s, maar je bent ook vrij om in je pauze even het centrum van Rotterdam in te gaan.

3 Opdrachtomschrijving

De opdracht tijdens deze afstudeerperiode is het ontwikkelen van een proof of concept van een module die zich richt op het geautomatiseerd testen van websites. Hoewel het te bouwen proof of concept in het afstudeerplan een ‘testsysteem’ wordt genoemd, is ‘module’ een meer toepasselijke term. De termen ‘module’ en ‘proof of concept’ zullen vanaf nu dan ook in het verdere verslag gebruikt worden.

De module wordt geïntegreerd in het zelf ontwikkelde framework van het bedrijf. De functie van de module is het testen van functionaliteit en het opsporen van fouten op websites die gebruikmaken van het framework. De functionaliteit van het opsporen en het achterhalen van fouten zal mede ontwikkelt worden met behulp van tools. Deze tools richten zich op het bieden van een framework voor het schrijven van testscripts. Deze tools worden geselecteerd door een toolselectie en worden geïntegreerd in de module. Soms zal er functionaliteit voorkomen die de module wel moet kunnen uitvoeren, maar wat niet ondersteund wordt door de tools. In dit geval worden de functies zelf geschreven als aanvulling op de tools.

Als de module fouten heeft opgespoord zullen deze gekoppeld worden aan wijzigingen in de code die deze fout waarschijnlijk hebben veroorzaakt. Deze wijzigingen worden opgehaald aan de hand van een connectie met de versiebeheer tool Gitlab. Deze connectie zal later nader worden toegelicht.

De opdracht is redelijk onvolledig beschreven in de komende hoofdstukken. Dit komt doordat er veel informatie mist wegens het vertrek van de opdrachtgever van deze opdracht. Later in het verslag zal er een nieuwe opdracht worden geformuleerd waarin meer in details getreden zal worden.

3.1 Probleemstelling

In de huidige situatie worden websites handmatig getest na het uitvoeren van wijzigingen in het framework. Het bedrijf telt teveel ontwikkelde websites om handmatig één voor één te kunnen doorlopen. Hierdoor wordt er bij het testen van wijzigingen een selectie van websites gemaakt die doorlopen worden. Het handmatig doorlopen van de websites en het opsporen van fouten kost veel tijd. Daarnaast is het niet erg betrouwbaar, omdat niet gegarandeerd iedere functionaliteit doorlopen zal worden. Er is een kans dat er delen over het hoofd gezien worden doordat er geen standaard set aan functionaliteit is die doorlopen wordt.

De selectie van websites geeft ook geen goed globaal beeld van alle websites, omdat niet iedere website hetzelfde functioneert en er fouten kunnen optreden op websites die buiten de selectie vallen. Dit heeft tot gevolg dat er fouten zullen zijn die niet achterhaald worden.

3.2 Doelstelling

Door het ontwikkelen van een module die geautomatiseerd test zal er veel tijd bespaard worden doordat het handmatig testen vervalt. Verder zal het proces betrouwbaarder worden, omdat de module álle websites die gebruik maken van het framework zal doorlopen in plaats van een selectie hiervan. Daarnaast loopt de module een vaste set met functionaliteiten af die getest worden zodat op iedere website hetzelfde wordt getest en er niets zal worden vergeten.

4 Plan van aanpak

Het plan van aanpak beschrijft de aanpak van de uit te voeren opdracht. Dit hoofdstuk beschrijft alleen de belangrijkste punten uit het plan. Het volledige plan van aanpak is te vinden in Bijlage B.

4.1 Iteratieve en incrementele ontwikkelmethode

Ik heb ervoor gekozen om deze opdracht iteratief en incrementeel uit te voeren volgens het agile principe. Omdat de opdracht niet tot in de details is uitgewerkt voor de start van de afstudeerperiode moet er ruimte zijn voor wijzigingen en aanvullingen. Niet alle requirements zijn aan het begin van de opdracht bekend. Er zullen gedurende het proces requirements bijkomen waardoor het project wendbaar moet zijn.

De volgende werkzaamheden zal ik aan het begin van de opdracht uitvoeren en vallen niet onder de iteraties, omdat deze maar eenmalig uitgevoerd hoeven te worden.

Toolselectie
Voor de opdracht zal ik een toolselectie uitvoeren voor het bepalen van de tools die geïntegreerd zullen worden in het proof of concept. De tools zullen een framework bieden voor het schrijven van testscripts. Met deze testscripts zullen er fouten op de websites die gebruik maken van het framework achterhaald worden.

Tijdens het maken van de toolselectie zal ik een longlist en een shortlist opstellen en zal ik een pilot uitvoeren waarin de tools zullen worden getest op het gebruik, de compatibiliteit en functionaliteit. Het eindresultaat van de selectie is een of meerdere tools die geschikt zijn om te integreren in het proof of concept. Deze stap wordt eenmalig uitgevoerd en zal verder niet meer geïtereerd worden, omdat dit niet benodigd is.

Inwerken
Ik zal inwerken in de interne systemen om zo kennis te vergaren en ervaring op te doen met de diversen systemen en methodes waar het bedrijf mee werkt. De systemen waarin ingewerkt zal worden zijn het Customer Relationship Management systeem (CRM), het Content Management Systeem (CMS) en het framework. Deze systemen zullen later worden toegelicht. Tijdens het inwerken zal ik gebruik maken van de documentatie van het bedrijf. Deze documentatie geeft uitleg over de diversen aspecten wat betreft de ontwikkeling van websites volgens de werkwijze van het bedrijf.

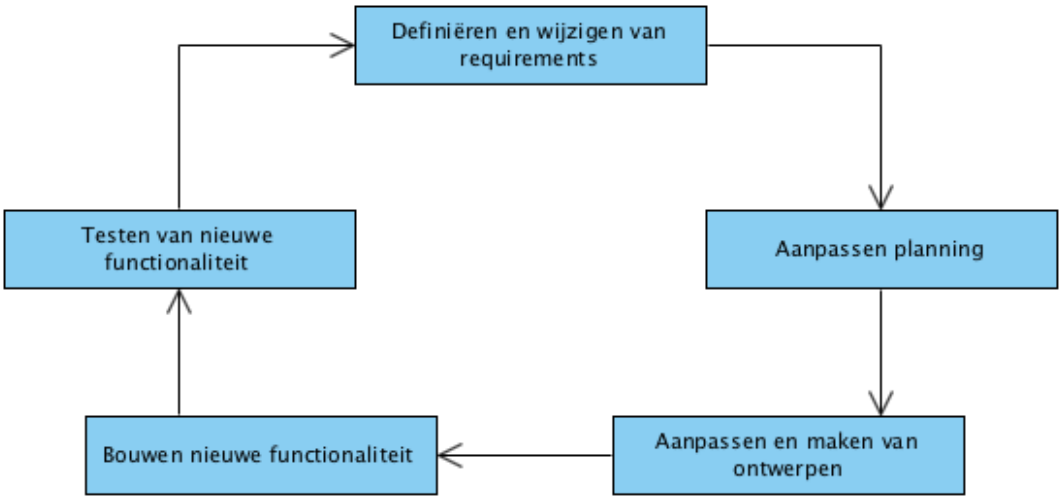
4.1.1 Iteraties

Gedurende de opdracht vinden er iteraties plaats gebaseerd op timeboxing en functionaliteit. De duur van de iteraties wordt bepaald door timeboxing. Voor iedere iteratie hou ik een timebox van twee weken aan. Op deze manier kan het proof of concept worden opgeleverd voor het einde van de afstudeerperiode. Iedere iteratie wordt er een nieuwe functionaliteit toegevoegd. Zo wordt het proof of concept incrementeel tot stand gebracht na iedere iteratie.

Er kunnen op dit moment nog geen iteraties worden vastgelegd, omdat er nog geen requirements bekend zijn. De iteraties zullen later in het proces gespecificeerd worden wanneer de requirements bekend zijn en er ingewerkt is.

Er zijn een aantal stappen die zullen worden uitgevoerd tijdens de iteraties. Deze stappen worden hier nader toegelicht (zie Afbeelding 1).

Afbeelding 1: De stappen tijdens de iteraties



Definiëren van (nieuwe) requirements
Aan het begin van de opdracht zal ik requirements bepalen en prioriteren aan de hand van gesprekken met de opdrachtgever. Tijdens de iteraties zullen er requirements wijzigen en requirements bijkomen. Voor het vaststellen van de requirements zal ik het boek 'handboek requirements' (Swart, 2010) erbij houden. Deze stap komt terug tijdens iedere iteratie. De requirements zullen worden vastgelegd in een requirementsrapport (zie Bijlage C).

Planning bijstellen
Tijdens de iteraties wordt indien nodig de planning aangepast. Dit kan het geval zijn als een iteratie langer of korter zal duren dan initieel ingeschat is door bijvoorbeeld gewijzigde of toegevoegde requirements. Wanneer er geen wijziging in de planning wordt vermeldt bij de iteratie, kan er vanuit worden gegaan dat de iteratie volgens de planning is verlopen.

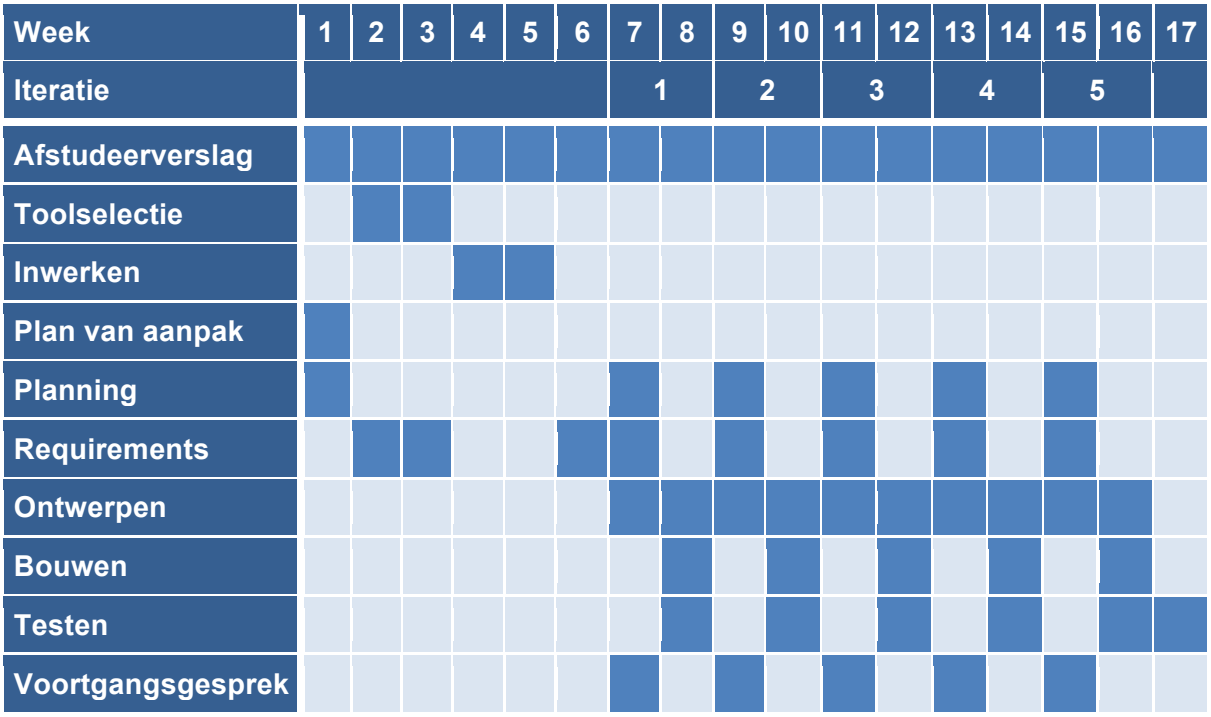
Ontwerpen
Gedurende de opdracht zal ik ontwerpen maken om onderdelen en functies uit te werken en in kaart te brengen voordat ik deze daadwerkelijk zal bouwen. De ontwerpen zullen gemaakt worden volgens de UML notatie, waarbij gebruik wordt gemaakt van het boek 'UML 2 and the unified process' (Jim Arlow, 2011). Tijdens de iteraties zal ik ontwerpen maken voor de nieuwe functionaliteit. De ontwerpen zullen fungeren als verheldering en lijdraad voor de functionaliteit en biedt het een visueel overzicht van de werking.

Het bouwen van nieuwe functionaliteit
Wanneer de functionaliteit gedefinieerd is in de requirements en er ontwerpen zijn gemaakt kan ik beginnen met bouwen. Tijdens iedere iteratie worden er nieuwe functionaliteiten toegevoegd aan het proof of concept.

Het testen van nieuwe functionaliteit
De laatste stap van iedere iteratie is het testen van de gebouwde functionaliteit. Voor het testen zal ik het boek 'TestGoal' (Grood, 2008) raadplegen. Ik zal tijdens het testen kijken of de functionaliteit aan de requirements voldoet en of het gebouwd is volgens het ontwerp. Voor iedere functionaliteit zal er een ander type test gewenst zijn. De gebruikte testmethoden worden nader toegelicht wanneer de functionaliteiten en iteraties bekend zijn.

4.2 Planning

De totale afstudeerperiode zal 17 kalenderweken in beslag nemen. Dit zijn 85 werkdagen. Er zijn 70 werkdagen ingepland om aan de opdracht te werken. De overige 15 werkdagen zullen worden besteed aan het afstudeerverslag.



4.2.1 Gemaakte afspraken

Er zullen voortgangsgesprekken plaatsvinden met mijn begeleider binnen het bedrijf, Stein Fortuin. Stein is tevens de opdrachtgever. De gesprekken zullen plaatsvinden om te zorgen dat de module aan de verwachtingen blijft voldoen. Naar aanleiding van deze gesprekken zal ook het requirementsrapport worden bijgeschaafd wanneer nodig.

5 Definitie van requirements

Dit hoofdstuk beschrijft alle requirements die ik heb vastgelegd aan de hand van een aantal gesprekken met de opdrachtgever. Om de opdracht een goede start te geven is het belangrijk dat er aan het begin van de opdracht al requirements worden achterhaald.

In dit hoofdstuk komen alleen de requirements met de prioriteit ‘Must have’ aan bod. ‘Must have’ is een term die voortkomt uit het principe van prioriteren volgens het MoSCoW principe. ‘Must have’ geeft aan dat dit een functionaliteit is die in de module moet voorkomen om het te laten slagen. MoSCoW staat voor ‘must have’, ‘should have’, ‘could have’ en ‘won’t have’. Ik heb ervoor gekozen om MoSCoW te hanteren, omdat er niet zo snel verwarring over zal ontstaan.

Bij het opstellen van de requirements is er rekening gehouden met herleidbaarheid. Dit is terug te vinden in de tabellen aan de hand van de kolom ‘BR’ wat verwijst naar de business requirements en de kolom ‘UC’ wat verwijst naar de use cases. De bijbehorende use cases zijn opgenomen in Bijlage C. Herleidbaarheid wil zeggen dat de requirements altijd zijn te herleiden naar hun bronnen. Het volledige overzicht van alle requirements waarin de bronnen en herleidbaarheid zijn verwerkt kan Bijlage C worden geraadpleegd.

5.1 Business requirements

Deze paragraaf behandeld de business rules en business requirements.

Business rules
Business rules definiëren of beperken een bepaald aspect van het bedrijf. De module moet zich aan deze regels houden.

Voor het bedrijf is het belangrijk dat er geen foutmeldingen worden weergeven op de websites die zij ontwikkelen. De websites moeten voor de bezoeker functioneren en geen verwarring of ongemak veroorzaken door bijvoorbeeld het niet kunnen bezoeken van pagina’s door foutmeldingen.

ID	Beschrijving	Stakeholder
R01	Bezoekers van een intern ontwikkelde website moeten in 99% van alle bezoeken een functionerende website zien.	Stein Fortuin

Business requirements
Business requirements zijn requirements die de bestaande bedrijfsprocessen verbeteren of het zijn nieuwe processen die de opdrachtgever met de module wil uitvoeren.

Tijdens de gesprekken is naar voren gekomen dat het proces van het opsporen en afhandelen van fouten geautomatiseerd moet worden. Fouten moeten op alle ontwikkelde websites geautomatiseerd worden achterhaald. Op deze manier wordt er tijd bespaard op de afhandeling van fouten op de websites.

De opdrachtgever wil daarnaast graag dat de module geïntegreerd wordt in de bestaande systemen, om zo een centraal punt te behouden waarin alle bedrijfsprocessen worden afgehandeld.

Verder wil de opdrachtgever de oorzaak van de opgespoorde fouten kunnen herleiden naar wijzigingen in de code. Deze tracering wordt tot stand gebracht om een richtlijn te

kunnen geven voor het oplossen van de fouten, waardoor deze sneller opgelost kunnen worden.

ID	Beschrijving	Stakeholder
B01	De opdrachtgever wil dat het afhandelen van fouten met ingang van de module geautomatiseerd wordt.	Stein Fortuin
B02	De opdrachtgever wil dat met ingang van de module 99% van de fouten veroorzaakt door wijzigingen in het framework afgevangen worden.	Stein Fortuin
B03	De opdrachtgever wil met ingang van de module álle websites kunnen testen, in tegenstelling tot een selectie van de websites.	Stein Fortuin
B04	De opdrachtgever wil dat het proces van foutafhandeling met ingang van de module inzichtelijk gemaakt wordt.	Stein Fortuin
B05	De opdrachtgever wil dat de module geïntegreerd wordt in één van de bestaande systemen	Stein Fortuin
B06	De opdrachtgever wil dat de code die de fout veroorzaakt achterhaald wordt d.m.v. een koppeling met versiebeheer	Stein Fortuin

5.2 User requirements

Deze paragraaf beschrijft de user requirements. Dit zijn de requirements die specificeren wat de gebruiker verwacht wat de module kan doen. De gebruiker is tijdens deze opdracht een werknemer van het bedrijf.

De gebruiker moet met de module alle fouten op de ontwikkelde websites kunnen opsporen om deze te kunnen afhandelen. Dit gebeurt met zo min mogelijk ingrepen, om het zoveel mogelijk geautomatiseerd te laten verlopen.

ID	Beschrijving	Stakeholder	BR	UC
U01	De gebruiker wil met zo min mogelijk handelingen de module aansturen om een test uit te voeren die fouten opspoort op de ontwikkelde websites.	Stein Fortuin	B02, B03	C01

5.3 Systeem requirements

Systeem requirements zijn de eisen die aan de module gesteld worden. Systeem requirements zijn onder te verdelen in functionele en niet-functionele systeem requirements.

Functionele requirements

Functionele requirements zijn eisen die gesteld worden aan het gedrag van de module.

Wat de opdrachtgever van de module verwacht is dat de testscripts voor het schrijven van de tests die de fouten op de websites zullen achterhalen, zoveel mogelijk geschreven worden met behulp van de geïntegreerde tools. Er is gekozen voor deze manier om tijdens de ontwikkeling niet opnieuw het wiel te hoeven uitvinden. Testscripts die niet ondersteund worden door de tools, maar wel benodigd zijn voor het opsporen en afhandelen van fouten, worden zelf geschreven.

ID	Beschrijving	Stakeholder	BR	UC
F01	De functionaliteit in de module wordt zoveel mogelijk geschreven met behulp van de geïntegreerde tools.	Stein Fortuin	B01, B02	C01
F02	De functionaliteiten die niet met behulp van een tool geschreven kan worden, maar wel benodigd zijn worden losstaand van de tools zelf geschreven.	Stein Fortuin	B02	C01
F03	De module moet als onderdeel van het intern ontwikkelde framework geïntegreerd worden.	Stein Fortuin	B05	-
F04	De module moet de mogelijkheid bieden om door websites te kunnen navigeren.	Stein Fortuin	B01, B02	C01
F05	De module moet tijdens het uitvoeren van tests iedere door het bedrijf ontwikkelde website doorlopen.	Stein Fortuin	B01, B02, B03	C01

Niet functionele requirements
De niet-functionele systeem requirements zijn criteria om het functioneren van de module te beoordelen. Deze requirements zijn opgesteld aan de hand van de ISO 25010. Dit is een norm die kwaliteiteskenmerken van software beschrijft.

Aangezien er met meerdere systemen zal worden gewerkt en gecommuniceerd, is het belangrijk dat de module informatie kan uitwisselen met de andere systemen.

ID	Requirement	ISO	Stakeholder	BR	UC
N01	De module moet informatie kunnen uitwisselen met diverse interne systemen.	Uitwisselbaarheid - Koppelbaarheid	Stein Fortuin	B05	-

Technische beperkingen
Onderstaand zijn de technische beperkingen van de module weergeven. Deze beperkingen geven de mogelijkheden en onmogelijkheden van de module aan.

Zoals eerder werd benoemd wil de opdrachtgever de module integreren in de bestaande systemen. Een bijkomende restrictie en reden hiervoor is dat er in de interne systemen functionaliteit beschikbaar is die ik nodig zal hebben voor het ontwikkelen van de module. Daarnaast moet de module, net als de andere interne systemen in PHP ontwikkelt worden.

ID	Technische beperking
T01	De module moet als onderdeel van de interne systemen geïntegreerd worden wegens beschikbare functionaliteit.
T02	De module moet PHP als hoofdtaal hebben.

6 Toolselectie

Dit hoofdstuk beschrijft de uitgevoerde toolselectie. Deze toolselectie selecteert aan de hand van een longlist, shortlist, onderzoek en een pilot welke tools geschikt blijken voor de integratie in het proof of concept.

6.1 Criteria

Voorafgaand aan de toolselectie heb ik aan de hand van gesprekken en onderzoek criteria achterhaald. In de onderstaande tabel staan deze criteria gedefinieerd. De criteria zijn vastgesteld om aan eisen van de opdrachtgever te voldoen en om het integreren van de tools zo optimaal te laten verlopen.

ID	Beschrijving
C01	De tool die geïntegreerd wordt in de module moet de optie bieden om zelf testscripts te schrijven.
C02	De tool die geïntegreerd wordt in de module moet gratis beschikbaar zijn.
C03	De tool die geïntegreerd wordt in de module moet open-source zijn.
C04	De tool die geïntegreerd wordt in de module moet documentatie aanbieden.
C05	De tool die geïntegreerd wordt in de module moet een losstaande applicatie zijn om een geheel met de module te kunnen vormen.
C06	De tool die geïntegreerd wordt in de module heeft geen GUI om zo feilloos te kunnen integreren in de module.
C07	De tool die geïntegreerd wordt in de module moet JavaScript of PHP ondersteunen.
C08	De tool die geïntegreerd wordt in de module moet de websites en de content hiervan kunnen doorlopen om fouten op te sporen.
C09	De tool die geïntegreerd wordt in de module moet up-to-date zijn. De laatste update is niet langer dan een jaar geleden.

6.2 Longlist

De eerste stap van de toolselectie is het opstellen van een longlist. Deze longlist bevat een uitgebreide verzameling van tools die voldoen aan de opgestelde criteria. Er wordt onderscheidt gemaakt tussen tools die JavaScript ondersteunen en tools die PHP ondersteunen. Dit onderscheidt is gemaakt omdat dit twee verschillende type tools zijn die beide benodigd zijn om zowel de server-side als de client-side van de websites te kunnen testen.

6.2.1 Criteria

De criteria C01 en C07 zijn in acht genomen bij het opstellen van de longlist. De keuze voor deze criteria heb ik gemaakt, omdat dit de meest algemene en belangrijke criteria zijn waaraan de tools moeten voldoen. Wanneer er geen testscripts geschreven kunnen worden met de tool heeft deze geen functie binnen de module, hetzelfde geldt voor tools die geen JavaScript of PHP ondersteunen. Deze ondersteuning is nodig om de websites, die ook geschreven zijn in JavaScript en PHP, te kunnen testen.

6.2.2 Search

Door verschillende bronnen via Google te raadplegen zal ik een eerste selectie van tools maken voor de longlist. De termen waarop ik zoek op Google zijn: 'web test tools', 'web testing tools' en 'test framework'. Voor de longlist wil ik vooral zoveel mogelijk opties verzamelen. Deze verzameling heeft geen maximale lengte. Er zullen zowel JavaScript als PHP tools geïntegreerd moeten worden om de websites zo volledig mogelijk te kunnen testen.

6.2.3 De lijst

De volgende longlist(s) heb ik opgesteld n.a.v. de criteria C01 en C07. Deze tools bieden een framework en/of gelegenheid voor het schrijven van testscripts. De lijst is gescheiden in tools die JavaScript ondersteunen en tools die PHP ondersteunen.

JavaScript

- Jasmine http://jasmine.github.io
- CasperJS http://casperjs.org
- PhantomJS http://phantomjs.org
- SlimerJS https://slimerjs.org
- Mocha http://mochajs.org
- JSUnit http://www.jsunit.net
- YUI Test http://yuilibrary.com/yui/docs/test/
- NodeUnit https://github.com/caolan/nodeunit
- Tyrtle https://github.com/spadgos/tyrtle

PHP

- PHPUnit https://phpunit.de/index.html
- SimpleTest http://www.simpletest.org
- Snaptest https://github.com/Jakobo/snaptest
- Kahlan https://github.com/crysalead/kahlan
- PHPInlineTest https://github.com/ptrofimov/phpinlinetest
- Atoum https://github.com/atoum/atoum
- Enhance PHP https://github.com/Enhance-PHP/Enhance-PHP

6.3 Shortlist

Na het opstellen van de longlist, wordt er verder gespecificeerd naar een shortlist. Bij deze shortlist wordt er rekening gehouden met de overige criteria. Om te bepalen welke tools aan welke criteria voldoen zijn er afwegingstabellen gemaakt. Deze tabellen wegen de tools uit de longlist af tegen de opgestelde criteria.

6.3.1 JavaScript

Onderstaand zijn de JavaScript tools onderworpen aan de criteria. Ik heb er voor gekozen om C08 niet als knock-out criterium te beschouwen. Deze keuze is gemaakt omdat tools elkaar kunnen aanvullen in functionaliteit. Hoewel sommige tools geen websites kunnen doorlopen, hebben ze mogelijk wel andere bruikbare testmogelijkheden.

Tool	C02	C03	C04	C05	C06	C08	C09
Jasmine							
CasperJS							
PhantomJS							

SlimerJS							
Mocha							
JSUnit							
YUI Test							
NodeUnit							
Tyrtle							

Hoewel CasperJS niet losstaand is, is er toch voor gekozen deze wel op te nemen in de shortlist. Deze keuze heb ik gemaakt, omdat de tool waarvan CasperJS afhankelijk is PhantomJS of SlimerJS is. Aangezien deze tools wel aan alle criteria voldoen is het dus geen probleem om CasperJS te integreren naast PhantomJS of SlimerJS.

JSUnit heeft een GUI. Dit is niet gewenst, omdat de tool geïntegreerd wordt in de module. De tool moet feilloos opgaan in de module en alleen functionaliteit en een framework bieden die door de module gebruikt kan worden.

Daarnaast vallen JSUnit, YUI Test en Tyrtle af, omdat deze niet up to date zijn. De laatste aanpassingen aan deze tools zijn al langer dan een jaar geleden.

Naar aanleiding van de afweging is de volgende shortlist opgesteld:
- Jasmine
- CasperJS
- PhantomJS
- SlimerJS
- Mocha
- NodeUnit

6.3.2 PHP

Onderstaand zijn de PHP tools zijn onderworpen aan de criteria. Criterium C08 is hierbij niet in acht genomen, omdat dit alleen van toepassing is voor de JavaScript tests.

Tool	C02	C03	C04	C05	C06	C09
PHPUnit						
SimpleTest						
SnapTest						
Kahlan						
PHPInlineTest						
Atoum						
Enhance PHP						

SimpleTest is niet geschikt gebleken. De reden hiervoor is dat de tool een GUI biedt om te kunnen testen en de tool niet up-to-date is.

Verder zijn Snaptest, PHPInlineTest en Enhance PHP afgevallen, omdat de laatste aanpassing aan deze tools al langer dan een jaar geleden zijn uitgevoerd. PHPInlineTest heeft daarnaast ook geen beschikbare documentatie, wat nog een reden geeft om deze tool niet te gebruiken.

Naar aanleiding van de afweging is de volgende shortlist opgesteld:
- PHPUnit
- Kahlan
- Atoum

6.4 Onderzoek
Om een selectie te kunnen maken worden de tools hier nader onderzocht op functionaliteit en compatibiliteit. De tools die geschikt blijken voor integratie worden meegenomen in de pilottest en de overige vallen alsnog af.

6.4.1 JavaScript tools
Jasmine
Jasmine is een BDD (Behaviour Driven Development) open source testing framework voor JavaScript. Jasmine claimt een heldere en duidelijke syntax te hebben waarmee je gemakkelijk tests kan schrijven. Jasmine is gemakkelijk te installeren door het te downloaden en direct in de bestanden van de te testen web applicatie op te nemen.

CasperJS
CasperJS is een navigation scripting en testtool geschreven in JavaScript voor gebruik in combinatie met PhantomJS of SlimerJS. CasperJS stelt je in staat navigatie scenario's op websites te schrijven.

PhantomJS
PhantomJS is een tool gebaseerd op JavaScript waarmee headless getest kan worden. Dit betekent dat er via de commandline tests geschreven kunnen worden zonder dat er een GUI aan te pas komt. De tool kan functionele tests uitvoeren met frameworks zoals Jasmine en Mocha. PhantomJS op zichzelf is geen test framework. PhantomJS stelt je in staat om web pagina's te laden en manipuleren, screen captures te maken en performance te monitoren.

SlimerJS
SlimerJS is vergelijkbaar met PhantomJS. Net als PhantomJS kan SlimerJS functionele tests uitvoeren, web pagina's laden en manipuleren, screen captures maken en performace monitoren. Het enige verschil met PhantomJS is dat SlimerJS runt op Gecko, de browser engine van Mozilla Firefox in plaats van Webkit en dat SlimerJS niet volledig headless is.

Omdat SlimerJS dusdanig vergelijkbaar is met PhantomJS, met enkele nadelen ten opzichte van PhantomJS, is besloten SlimerJS niet meer mee te nemen in de selectie.

Mocha
Mocha is een JavaScript testing framework die runt op Node.js en in de browser. Node.js is een open-source runtime omgeving voor het ontwikkelen van server-side web applicaties. Voor het gebruik van Mocha zal Node.js ook geïnstalleerd moeten worden. Omdat Node.js zo'n wijds gebruikt en bekende tool is, zie ik dit niet als een knock-out factor voor deze tool. Mocha heeft verschillende features zoals: synchroon testen, asynchroon testen en veel meer features die je in staat stellen verschillende testen uit te voeren.

NodeUnit
NodeUnit maakt net als Mocha gebruik van Node.js en de browser en stelt je in staat om asynchrone unit testen te schrijven voor Node.js en de browser. NodeUnit is te installeren via een download van Github of via npm. Npm is de standaard package manager van Node.js.

6.4.2 PHP tools

PHPUnit
PHPUnit is een test framework voor PHP en is een onderdeel van xUnit. xUnit is een architectuur voor unit testing frameworks. xUnit is ontstaan uit sUnit en is bekend geworden met de tool JUnit.

Om PHPUnit te kunnen gebruiken kan er een PHP Archive (PHAR) bestand worden gedownload. Dit bestand kan geïnstalleerd worden met behulp van de command line.

Kahlan
Kahlan is een Unit en BDD test framework. Deze tool maakt gebruik van een describe-it syntax en is ontwikkeld op basis van het KISS (Keep It Stupid Simple) principe. De tool van gedownload worden vanaf Github en geïnstalleerd worden via Composer of Git clone.

Atoum
Atoum is wederom een PHP testing framework. Atoum is ontwikkelt met de volgende principes: het kan snel geimplementeerd worden, het versimpelt test development en het laat het toe om betrouwbare, leesbare en heldere unit tests te schrijven. Atoum kan net als PHPUnit met een PHAR geïnstalleerd worden, maar er kan ook gekozen worden om het via Composer te installeren of het te clonen vanaf Github.

6.5 Pilot

De pilot is de testrun van de tools. De pilot zal uitwijzen of de tools geschikt zijn voor integratie in de module en of deze fijn in gebruik en makkelijk te begrijpen zijn.

De tools die meegenomen in de pilot zijn:

JavaScript
- Jasmine
- CasperJS
- PhantomJS
- Mocha
- NodeUnit

PHP
- PHPUnit
- Kahlan
- Atoum

6.5.1 Wijziging

De pilot is vroegtijdig gestopt wegens een herziening van de afstudeeropdracht waarbij de toolselectie niet meer benodigd is. De herziening van de opdracht wordt beschreven in het volgende hoofdstuk.

7 Herziening van de opdracht

De opdracht zoals deze beschreven is in de opdrachtsomschrijving en het afstudeerplan is na vier weken genoodzaakt veranderd. De verandering is tot stand gekomen wegens omstandigheden binnen het bedrijf waardoor mijn begeleider is vertrokken en ik een andere begeleider kreeg toegewezen. De verandering van begeleider/opdrachtgever, heeft gezorgd voor een nieuwe visie voor de opdracht en botsende requirements ten opzichte van de oorspronkelijke opdracht. Dit hoofdstuk beschrijft uitgebreid de herziening van de opdracht.

7.1 Vernieuwde opdrachtsomschrijving

De vernieuwde opdracht tijdens deze afstudeerperiode is het optimaliseren en automatiseren van het bedrijfsproces waarin fouten op websites en in het CMS van deze websites worden afgehandeld. Er zal naar het proces van het afhandelen van fouten (exceptions) verwezen worden als ‘exception handeling’. Ik zal een proof of concept bouwen van de module die dit zal realiseren. De realisatie van de vernieuwde opdracht is anders dan de oorspronkelijke opdracht.

Veranderingen ten opzichte van de orginele opdracht

- De module zal niet testen met behulp van tools en testscripts zoals orgineel het plan was. De module zal exceptions opsporen op websites en deze traceerbaar maken zodat deze snel kunnen worden opgelost.
- De module heeft betrekking op het CMS, het CRM en het framework.
- Er wordt een onderdeel toegevoegd dat de exceptions inzichtelijk en beheerbaar maakt.
- Het proces van de afhandeling van exceptions zal worden verbeterd.

Probleemstelling

In de huidige situatie is de exception handeling van het bedrijf niet optimaal. Exceptions worden in het bedrijf behandelt als supporttickets. Supporttickets bevatten vragen van klanten en zijn het middel van communicatie met het supportteam. De tickets met exceptions komen terecht bij werknemers die bezig zijn met support en klantcontact, maar niet per definitie developers zijn. Doordat er tickets met exceptions tussen de reguliere supporttickets staan is er verminderd overzicht op de supporttickets die wel relevant zijn voor de supportafdeling.

Daarnaast heeft het bedrijf in de huidige situatie geen mogelijkheid voor het traceren van exceptions naar de oorzaak. Wanneer er een exception optreed moet er handmatig worden achterhaald waar de exception door veroorzaakt wordt om deze te kunnen oplossen. Verder is er geen inzicht in alle opgetreden exceptions en zijn deze ook niet beheerbaar, waardoor er geen inzicht is in het oplossen van de exceptions.

Doelstelling

Het afhandelen van exceptions kan efficiënter worden uitgevoerd door deze te scheiden van de supporttickets. Om de afhandeling van exceptions te optimaliseren zal ik een proof of concept van een module bouwen waarin ik de exceptions overzichtelijk en beheerbaar zal maken. Daarnaast zal ik een tracering tot stand brengen die de oorzaak van de exception kan herleiden naar wijzigingen in de code via versiebeheer.

7.2 Gevolgen

De herziening van de opdracht heeft ook wijzigingen tot stand gebracht in de op te leveren (tussen)producten. Deze worden hier nader toegelicht.

De toolselectie
De toolselectie, zoals uitgevoerd in het vorige hoofdstuk en in Bijlage D, is niet meer benodigd voor de vernieuwde opdracht. Hoewel de toolselectie wel voor een groot deel is uitgevoerd, worden de geselecteerde tools niet meer geïntegreerd in het proof of concept en zal de pilot niet meer worden uitgevoerd.

Requirementsrapport
Voordat bekend werd dat de opdracht herzien zou worden had ik al requirements opgesteld. Omdat de opdracht een andere draai heeft gekregen kwamen er botsende requirements. Daardoor zijn de requirements die opgesteld zijn vòòr de herziening van de opdracht in een pre herziening requirementsrapport (zie Bijlage C) vastgelegd en de nieuwe requirements in een apart requirementsrapport bijgehouden (zie Bijlage E). In de volgende paragraaf worden de botsende requirements nader toegelicht.

Beroepstaken
Door de wijziging van de opdracht is de gekozen beroepstaak 1.3: ‘Selecteren van standaardsoftware’ komen te vervallen en is er een nieuwe beroepstaak voor in de plaats gekomen, namelijk beroepstaak 1.4: ‘Uitvoeren analyse door definitie van requirements’. Deze keuze is gemaakt, omdat door het afvallen van de toolselectie beroepstaak 1.3 niet op niveau 3 kan worden uitgevoerd. Beroepstaak 1.4 daarentegen is door de herziening heel uitgebreid uitgevoerd, daardoor is ervoor gekozen deze op niveau 4 uit te voeren.

Nieuwe begeleider/opdrachtgever
De nieuwe opdracht heeft een nieuwe begeleider/opdrachtgever met zich meegebracht. De opdracht is overgenomen van Stein Fortuin door Stephan Dorst. Stephan zal mij begeleiden gedurende de afstudeerperiode en is tevens de nieuwe opdrachtgever.

7.3 Vernieuwde requirements
Door de herziening van de opdracht waren er veel requirements niet meer relevant en kwamen er veel nieuwe requirements bij door gesprekken met de opdrachtgever. De opdrachtgever waarmee de oorspronkelijke opdracht is opgesteld (Stein) en de nieuwe opdrachtgever (Stephan) hebben een andere visie voor het proof of concept van de module. Dit heeft automatisch geleid tot botsende requirements. Zo moest in de oude situatie er een module gebouwd worden als uitbreiding van het framework, in de nieuwe situatie is de module geïntegreerd in de diversen interne systemen. Daarnaast was het in de oude situatie de bedoeling dat de module functionaliteit zou testen, in de nieuwe situatie worden er exceptions opgespoord en wordt er een tracering tot stand gebracht. Hier valt het daadwerkelijk testen van functionaliteit niet meer onder.

Deze paragraaf bevat alle nieuwe en gewijzigde requirements met een prioriteit ‘Must have’ volgens het MoSCoW principe. De oude requirements zijn te vinden in Bijlage C en de nieuwe requirements in Bijlage E. De nieuwe requirements zijn achterhaald door gesprekken met zowel Floris (de eigenaar van Floro) als mijn begeleider Stephan.

7.3.1 Business requirements
Onderstaand worden de vernieuwde business rules en requirements benoemd.

Business rules
De business rule is ten opzichte van de oude situatie niet gewijzigd. Daarom is deze hier niet apart genoteerd.

Business requirements

ID	Beschrijving	Stakeholder
B01	De opdrachtgever wil dat exception handeling en tracering met ingang van de module in het CRM geautomatiseerd wordt.	Stephan Dorst
B02	De opdrachtgever wil dat met ingang van de module in het CRM alle exceptions op websites die draaien op het interne framework afgevangen worden.	Floris van Leerdam
B03	De opdrachtgever wil dat met ingang van de module in het CRM alle fouten in het CMS van websites die draaien op het interne framework afgevangen worden.	Floris van Leerdam
B04	De opdrachtgever wil met ingang van de module alle websites die draaien op het interne framework en het CMS van deze websites kunnen aflopen op exceptions.	Floris van Leerdam
B05	De opdrachtgever wil met ingang van de module in het CRM de oorzaak van de exceptions kunnen traceren.	Stephan Dorst
B06	De opdrachtgever wil dat er voor de module een nieuw onderdeel wordt toegevoegd aan het CRM waarin een overzicht van de exceptions wordt getoond en waarin de exceptions beheerd kunnen worden.	Floris van Leerdam

7.3.2 User requirements

De gebruiker is tijdens de nieuwe opdracht nogsteeds een werknemer van het bedrijf. De bijbehorende use cases zijn te vinden in Bijlage E.

ID	Beschrijving	Stakeholder	BR	UC
U01	De gebruiker wil met het onderdeel in het CRM een overzicht kunnen opvragen van de exceptions die plaatsvonden op een domein.	Floris van Leerdam	B06	C01
U02	De gebruiker wil met de module exceptions kunnen beheren.	Floris van Leerdam	B06	C02

7.3.3 Systeem requirements

Onderstaand worden de vernieuwde requirements met betrekking op de module nader verklaard.

Functionele requirements

ID	Beschrijving	Stakeholder	BR	UC
F01	De module bestaat o.a. uit een onderdeel in het CRM van het bedrijf.	Floris van Leerdam	-	-
F02	De module moet iedere website die op het interne framework draait en het CMS van deze websites doorlopen.	Floris van Leerdam	B04	-

F03	De module kan exceptions traceren naar de oorzaak in de broncode.	Floris van Leerdam	B05	-
F04	De module maakt exceptions beheerbaar.	Floris van Leerdam	B06	C02

Niet-functionele requirements

ID	Requirement	ISO	Stakeholder	BR	UC
N01	De module moet informatie kunnen uitwisselen met diverse interne systemen.	Uitwisselbaarheid - Koppelbaarheid	Floris van Leerdam	-	C01, C02

Technische beperkingen

ID	Technische beperking
T01	De module moet een onderdeel in het CRM hebben worden wegens beschikbare functionaliteit.
T02	De module moet PHP als hoofdtaal hebben.

7.4 Aanpassingen aan plan van aanpak

Vanzelfsprekend zijn er aanpassingen aan het plan van aanpak als gevolg van de herziening van de opdracht. Deze aanpassingen worden onderstaand beschreven.

7.4.1 Werkzaamheden

Er worden twee werkzaamheden toegevoegd die eenmalig uitgevoerd dienen te worden. Dit zijn een procesbeschrijving van het huidige proces en het herschrijven van de huidige exception handeling.

Procesbeschrijving

Tijdens de procesbeschrijving zal ik het huidige proces wat betreft de exception afhandeling van de websites vastleggen. Er zal naar dit proces verwezen worden als ‘exception handeling’. Om dit proces te optimaliseren zal het huidige proces in kaart worden gebracht en zal de werking ervan moeten worden begrepen.

Herschrijven van de huidige exception handeling

Omdat de huidige exception handeling niet optimaal functioneert en in de nieuwe situatie anders moet werken zijn er aanpassingen nodig aan dit onderdeel. De aanpassingen omvatten een verandering van de huidige exception handeling en een aantal uitbreidingen. De aanpassingen en uitbreidingen zal in tijdens deze werkzaamheid vastleggen.

7.4.2 Iteraties en planning

De planning zal worden aangepast op de herziening van de opdracht wanneer er bekend is wat er aangepast en toegevoegd moet worden aan het huidige proces. Wanneer deze werkzaamheden bekend zijn zullen de iteraties worden gespecificeerd en zal de planning worden aangepast.

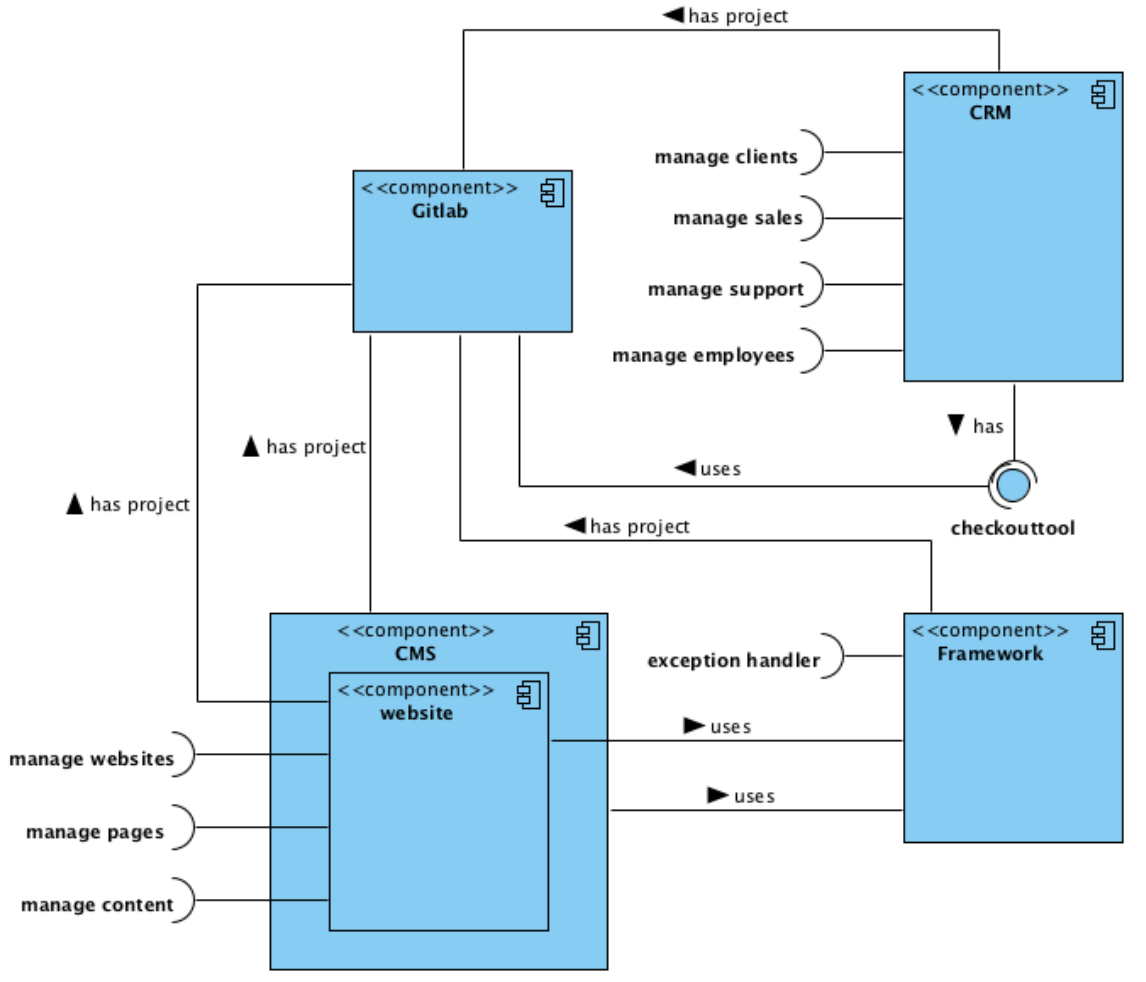
8 Inwerken

Dit hoofdstuk beschrijft het proces van inwerken in de diverse systemen van het bedrijf. Deze systemen zijn: het CRM, het CMS en het interne framework. Tijdens het inwerken heb ik gebruik gemaakt van de interne documentatie van Floro. Deze documentatie geeft uitleg over de werkwijze en methodes die het bedrijf gebruikt.

8.1 De diversen systemen

Ik zal inwerken in de diversen systemen waarmee het bedrijf werkt en waarmee ik in aanraking kom tijdens het ontwikkelen van het proof of concept. Deze systemen heb ik in kaart gebracht in een UML component diagram (zie Afbeelding 2). De afbeelding laat de diversen systemen binnen het bedrijf zien en de relaties met de andere systemen. Daarnaast zijn de interfaces van de systemen weergeven in lollipop notatie.

Afbeelding 2: Component diagram interne systemen van Floro



CRM

Het CRM is het Customer Relationship Management systeem van het bedrijf. Dit systeem wordt ook wel CusWeb genoemd. Dit systeem beheert alle klanten en interne zaken van het bedrijf, zoals sales, support, betalingen en werknemers. Voor ieder onderdeel bevat het CRM een aparte module.

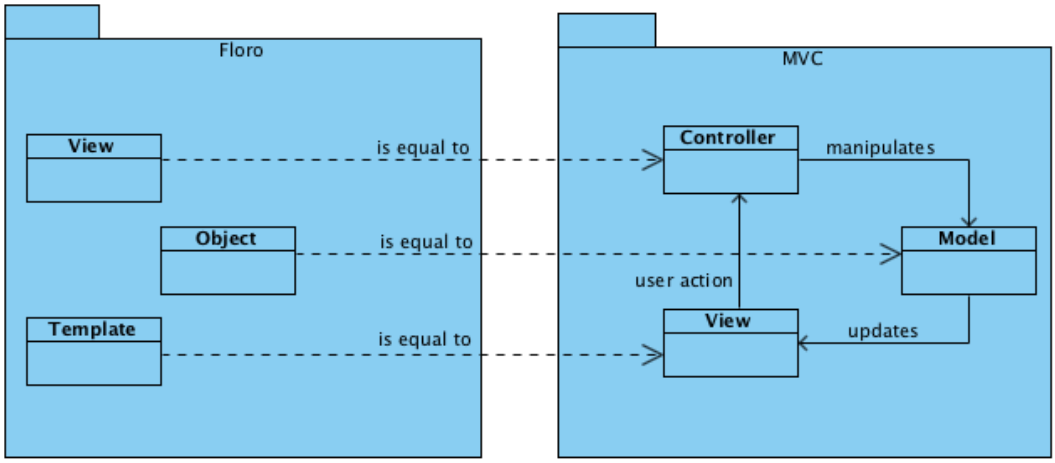
Iedere werknemer heeft een account op CusWeb, maar niet iedere werknemer heeft toegang tot alle gegevens die in CusWeb staan. Hiervoor zijn er gebruikersrechten die aangeven wie toegang tot wat heeft. Deze gebruikersrechten bestaan omdat niet iedere werknemer alle elementen in CusWeb mag beheren. Zo gaan bijvoorbeeld de ontwikkelaars niet over interne zaken zoals betalingen.

CMS
Het CMS is het Content Management Systeem genaamd 'One'. Het CMS beheert de content van alle websites. Via One kunnen de werknemers bij alle content van alle websites. Ook kunnen o.a. alle gebruikers worden beheerd, kunnen er statistieken worden opgehaald en kunnen er supporttickets worden beheerd. Daarnaast zijn er diversen tools aanwezig zoals overzichten van domein statussen, logs, ip-adressen en vertalingen.

Framework
Het framework is het geraamte waarmee bijna alle websites zijn gebouwd. Het framework bevat standaard functionaliteit die gebruikt kan worden voor het ontwikkelen van de websites. Het framework bevindt zich in een map 'framework' in de 'libraries' map van de website. Het framework bestaat uit een verzameling classes die ieder een eigen verantwoordelijkheid hebben. De functionaliteit van deze classes kan gebruikt worden in de views van de websites. De views bevatten de functionaliteit van de templates. De templates zijn de voorkant van de websites. Deze bevatten de HTML output die de bezoeker van de website te zien krijgt.

Om de termen van binnen het bedrijf te vertalen in MVC begrippen, zijn de views vergelijkbaar met controllers en de templates met views (zie Afbeelding 3).

Afbeelding 3: Vergelijking van framework met de MVC structuur

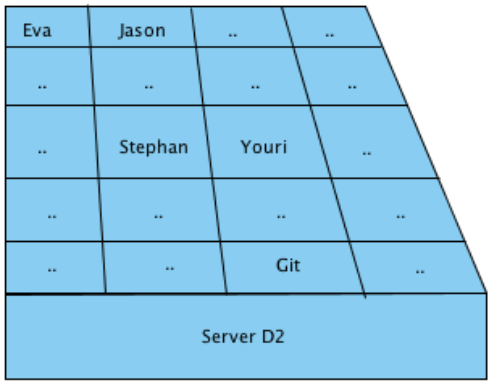


8.2 Ontwikkelomgeving

Tijdens het inwerken heb ik kennis gemaakt met de interne manier van werken in een ontwikkelomgeving. Binnen het bedrijf werken werknemers niet lokaal op een PC of laptop wanneer er een website ontwikkeld wordt, maar wordt er gebruik gemaakt van een 'development container'. Deze container is een persoonlijk stukje server waarover iedere developer beschikt. Deze omgeving dient als vervanging voor de lokale ontwikkelomgeving.

Het bedrijf heeft meerdere servers die allemaal zijn opgedeeld in stukjes. Deze stukken heten partities. Op iedere partitie staat een andere omgeving. Zo staat ook iedere development container op een eigen partitie op een server (zie Afbeelding 4).

Afbeelding 4: Server met partites voor development containers

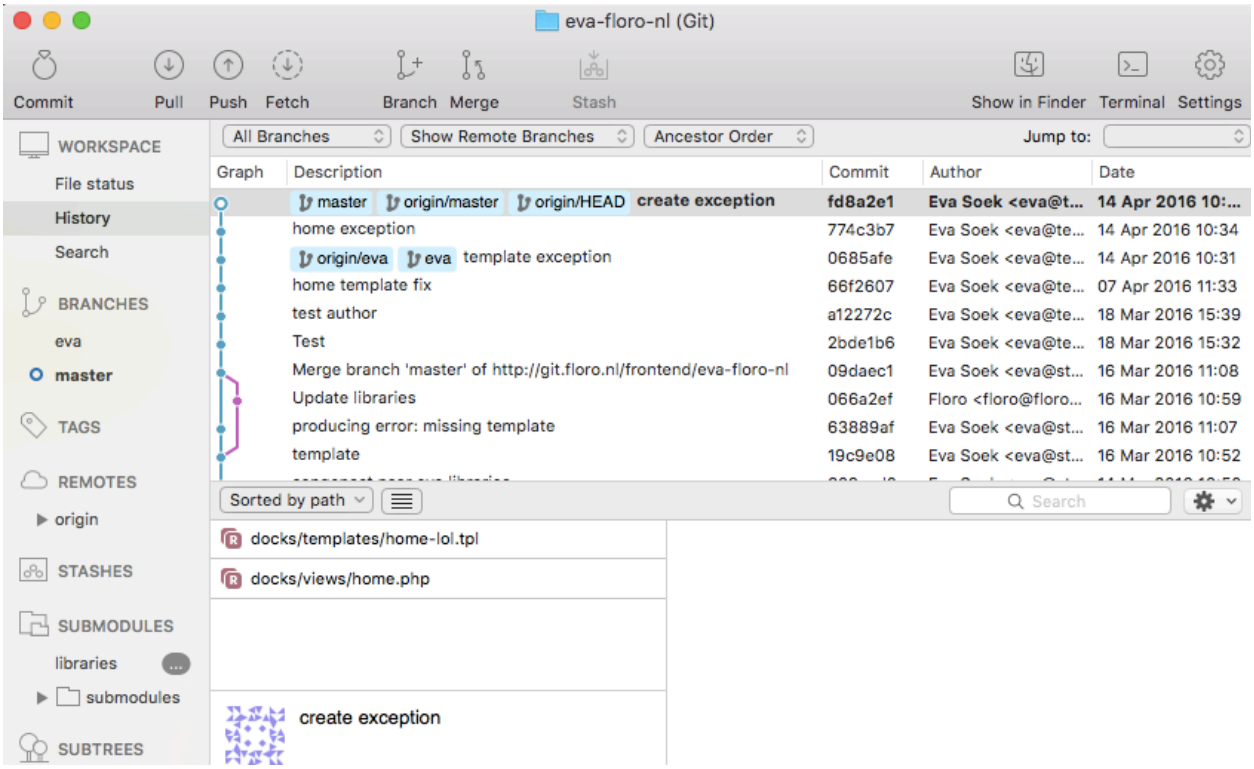


8.3 Versiebeheer met Gitlab

Het bedrijf beheert hun Git repositories via Gitlab. Gitlab houdt veranderingen in de bestanden en broncode bij van websites. Op deze manier kunnen verschillende versies van de websites worden geraadpleegd. Alle bestanden en versies van websites worden opgeslagen in zogenaamde ‘repositories’.

Voor deze opdracht gebruik ik de tool SourceTree voor het beheren van de Gitlab projecten (zie Afbeelding 5). Deze tool is mij aangeraden door een collega en is beschikbaar voor Mac. SourceTree stelt mij in staat alle mogelijke commando’s van Git uit te voeren in een toegankelijke interface.

Afbeelding 5: Versiebeheer met Git via SourceTree



Checkouts

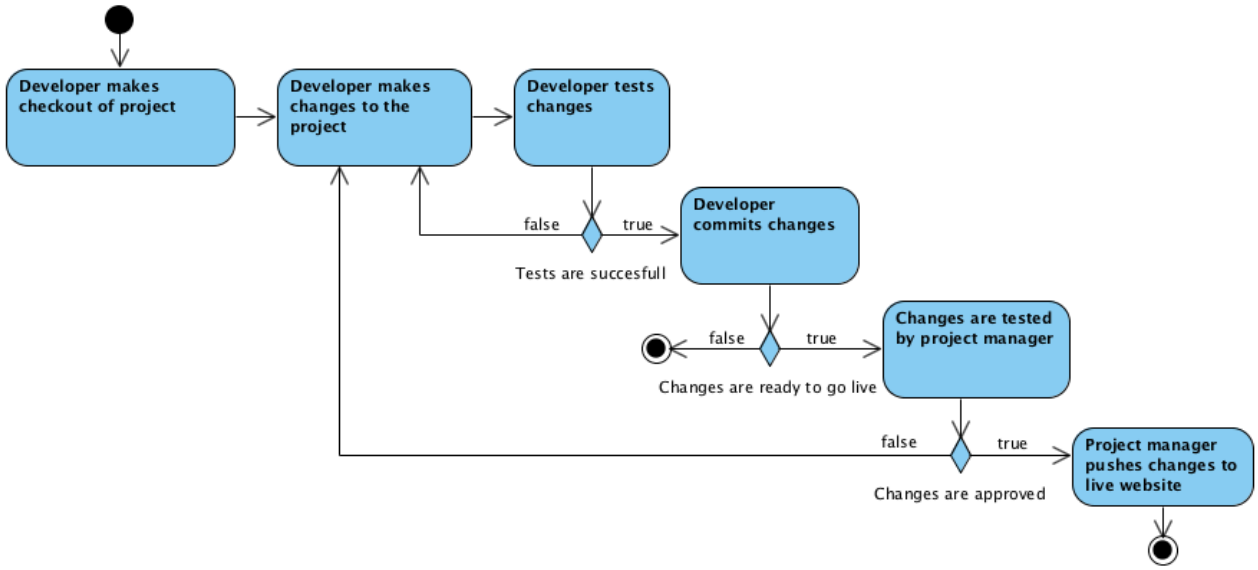
Versiebeheer staat het toe om checkouts te maken van repositories. Een checkout is een complete werkende kopie van een repository. Via een ingebouwde checkout tool in het

CRM van het bedrijf, kan er via een interface voor iedere website met een Git project een checkout gemaakt worden op een eigen development container.

Aanpassingen aan projecten worden altijd in een lokale checkout van een repository gemaakt. Deze checkout is een eigen kopie van een project waarin ongestoord gewerkt kan worden aan een website zonder dat hierbij de andere versies gewijzigd worden.

Commits
Wanneer er wijzigingen in een project zijn aangebracht en je wilt deze doorvoeren in het orginele project moeten deze gecommit worden. Een commit is het doorvoeren van de lokale wijzigingen naar de repository in Gitlab. Op dit moment staan deze wijzigingen nog niet live. Binnen het bedrijf is het gebruikelijk dat de wijzigingen, wanneer deze klaar zijn om live te worden gezet, nogmaals worden gecontroleerd door de project manager. Wanneer de wijzigingen zijn goedgekeurd worden deze door hem live gezet (zie Afbeelding 6). Deze tweede controle door de project manager vindt niet plaats wanneer een commit niet live gezet hoeft te worden.

Afbeelding 6: Commit proces voor Git projecten



Pull
Wanneer er meerdere mensen aan het Git project werken moet ervoor worden gezorgt dat de wijzigingen die anderen gecommit hebben binnengehaald worden. Hiervoor wordt een ‘pull’ request uitgevoerd. Pull haalt alle wijzigingen van anderen binnen en verwerkt deze een jou eigen lokale checkout zodat je de nieuwste versie van het project tot je beschikking hebt.

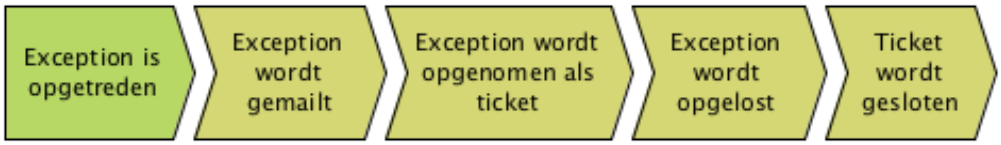
9 Procesbeschrijving exception handeling

Dit hoofdstuk beschrijft het huidige proces van de afhandeling van exceptions. Dit proces heb ik in kaart gebracht door de code te bestuderen.

9.1 Het proces

Om het huidige proces in kaart te brengen heb ik een procesdiagram gemaakt (zie Afbeelding 7). Dit diagram beschrijft het huidige proces van de afhandeling van exceptions die optreden op de websites die gebruik maken van het framework.

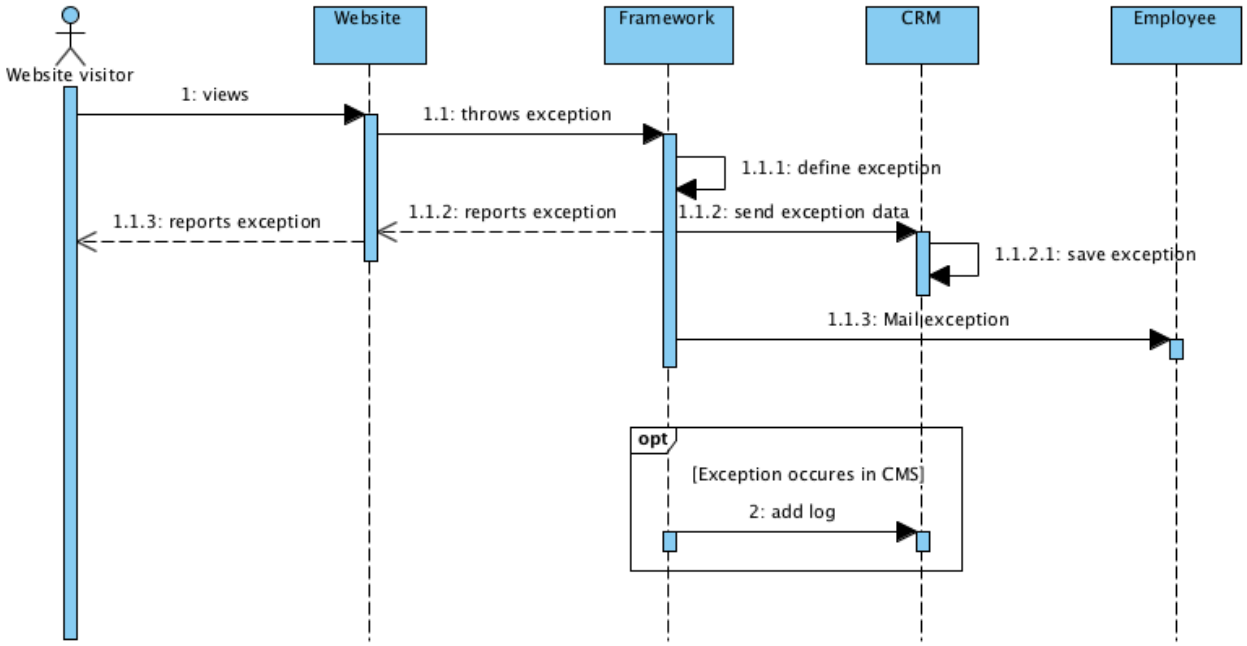
Afbeelding 7: proces van het afhandelen van exceptions



Als eerst worden de exceptions opgevangen door de exception handler. Deze exceptions worden per mail verstuurd naar de supportafdeling. Deze mail stelt de supportafdeling ervan op de hoogte dat er een exception is opgetreden. Via de mail worden de exceptions als tickets opgenomen in de database van het CRM. De ticket met exception wordt toegekend aan een werknemer die de exception zal oplossen. Wanneer de exception is verholpen, wordt de ticket gesloten en is het proces afgerond.

In de vorm van een sequence diagram ziet het proces er als volgt uit (zie Afbeelding 8).

Afbeelding 8: Sequence diagram exception handler



9.2 Huidige exception handler

Om zowel tickets vanuit de frontend als de backend op te vangen is er een exception handler aanwezig in het framework en in het CMS. Deze exception handler is een class

genaamd ‘class.exception_handler’. Deze class wordt aangeroepen wanneer een bezoeker op een website of in het CMS van deze website op een foutmelding stuit.

Wanneer de exception zich voordoet op een ontwikkelomgeving wordt de exception in z’n geheel op de betreffende pagina op de website getoond. Dit is handig, omdat de ontwikkelaars direct tijdens het ontwikkelen met deze informatie de exception kunnen verhelpen.

Als de exception zich voordoet op een live website wordt er een mail naar de supportafdeling gestuurd. Via de mail komt de exception in de database van het CRM terecht. In de database van het CRM wordt de exception als ticket opgeslagen en behandeld als een reguliere supportticket. Naast de mail wordt er een aangepast bericht gegenereerd om aan de bezoeker van de website te laten zien. Dit bericht komt op de pagina waarop de exception zich heeft voorgedaan te staan. Voor een bezoeker is het van toepassing dat de melding in leesbare taal aangeeft dat er een fout is opgetreden. In de mail naar de supportafdeling wordt er wel technische informatie van de exception meegestuurd. Dit is omdat de ontwikkelaars met de informatie uit de mail de exception zullen gaan oplossen.

Een extra onderdeel van de exception handler in het CMS dat niet in het framework voorkomt is dat er een log wordt bijgehouden voor de exceptions. De log bevat technische informatie van de exception. Er is geen reden dat er geen log wordt bijgehouden voor exceptions in het framework. Dit is gewoon een kleine bug in het systeem gebleken.

9.3 Problemen met het huidige proces

Zoals beschreven in de probleemstelling is de huidige situatie is de exception handeling van het bedrijf niet optimaal. Exceptions worden in het bedrijf behandeld als supporttickets waardoor er verminderd overzicht is op de supporttickets die wel relevant zijn voor de supportafdeling. De tickets komen terecht bij werknemers die bezig zijn met support en klantcontact, maar niet per definitie developers zijn.

Daarnaast heeft het bedrijf in de huidige situatie geen mogelijkheid voor het traceren van exceptions naar de oorzaak. Wanneer er een exception optreed moet er handmatig worden achterhaald waar de exception door veroorzaakt wordt om deze te kunnen oplossen. Verder is er geen inzicht in alle opgetrede exceptions en zijn deze ook niet beheerbaar. Deze problemen zullen zo goed mogelijk worden verholpen door het herschrijven van de huidige exception handeling en het bouwen van het proof of concept.

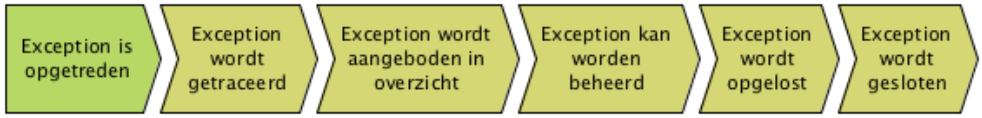
10 Herschrijven van de huidige exception handeling

Omdat het huidige proces niet optimaal functioneert zijn er aanpassingen en toevoegingen nodig. Dit hoofdstuk beschrijft de wijzigingen die benodigd zijn om het huidige systeem te optimaliseren en de problemen die er zijn te verhelpen.

10.1 Het vernieuwde proces

Voor het in kaart brengen van het nieuwe proces, heb ik wederom een procesdiagram gemaakt (zie Afbeelding 9). Dit diagram beschrijft het nieuwe proces van de exception handeling waarin de exceptions worden gescheiden van de algemene supporttickets. De exceptions zullen een eigen onderdeel krijgen binnen het CRM onder de categorie ‘Support’.

Afbeelding 9: Vernieuwde proces van het afhandelen van exceptions



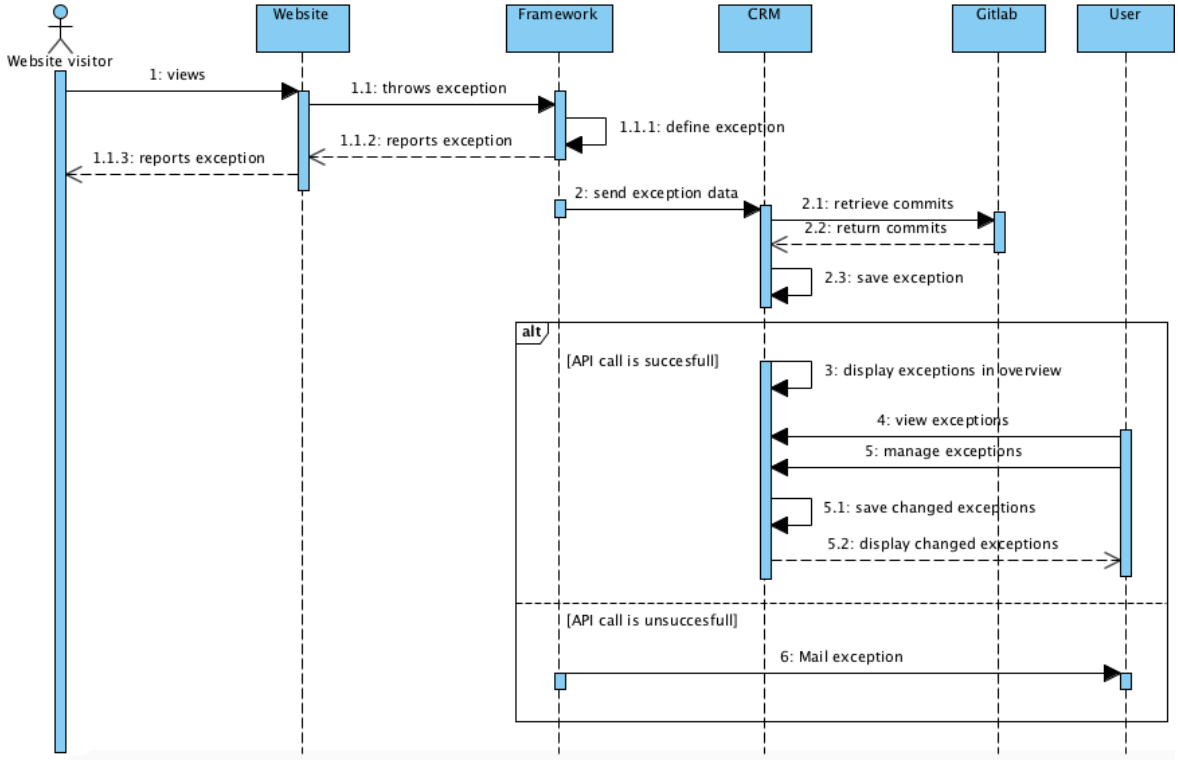
In het vernieuwde proces worden exceptions opgevangen door de exception handlers die ik zal gaan herschrijven. Deze handlers zorgen ervoor dat er verbinding wordt gemaakt met Gitlab. Via Gitlab worden de laatste wijzigingen van het Git project van de website waarop de exception zich heeft voorgedaan getraceerd. Vervolgens worden de exceptions met de getraceerde data opgeslagen in de database via een API call en worden deze getoond in een overzicht in het CRM. Dit overzicht kan geraadpleegd worden door werknemers die rechten hebben om de support module in het CRM te bekijken.

De werknemers kunnen de exceptions beheren door het overzicht te filteren op domein, status, prioriteit of werknemer. Daarnaast kan er een status, prioriteit en werknemer aan de exceptions worden toegekend. Wanneer een exception opgelost dient te worden kent de werknemer een prioriteit en ontwikkelaar toe aan de exception. Deze ontwikkelaar kan vervolgens de details van de exception bekijken waarin o.a. de tracering staat aangeduidt. De ontwikkelaar kan aan de slag met het oplossen van de exception en kent hierbij de status ‘being resolved’ toe aan de exception die hij/zij dient op te lossen. Als de exception is opgelost wordt de status gewijzigd naar ‘resolved’. De exception is hiermee gesloten en het proces afgerond.

Verder is er ook de mogelijkheid om een werknemer te herinneren aan het oplossen van de exception d.m.v. het sturen van notificaties en reminders. Dit valt ook onder beheerbaarheid.

In de vorm van een sequence diagram ziet het proces van het exception handelen er als volgt uit (zie Afbeelding 10).

Afbeelding 10: Sequence vernieuwde exception handeling



10.2 Vernieuwde exception handler

Het te bouwen proof of concept van de module bestaat uit meerdere onderdelen in de diversen interne systemen (zie Afbeelding 11).

Exception handlers

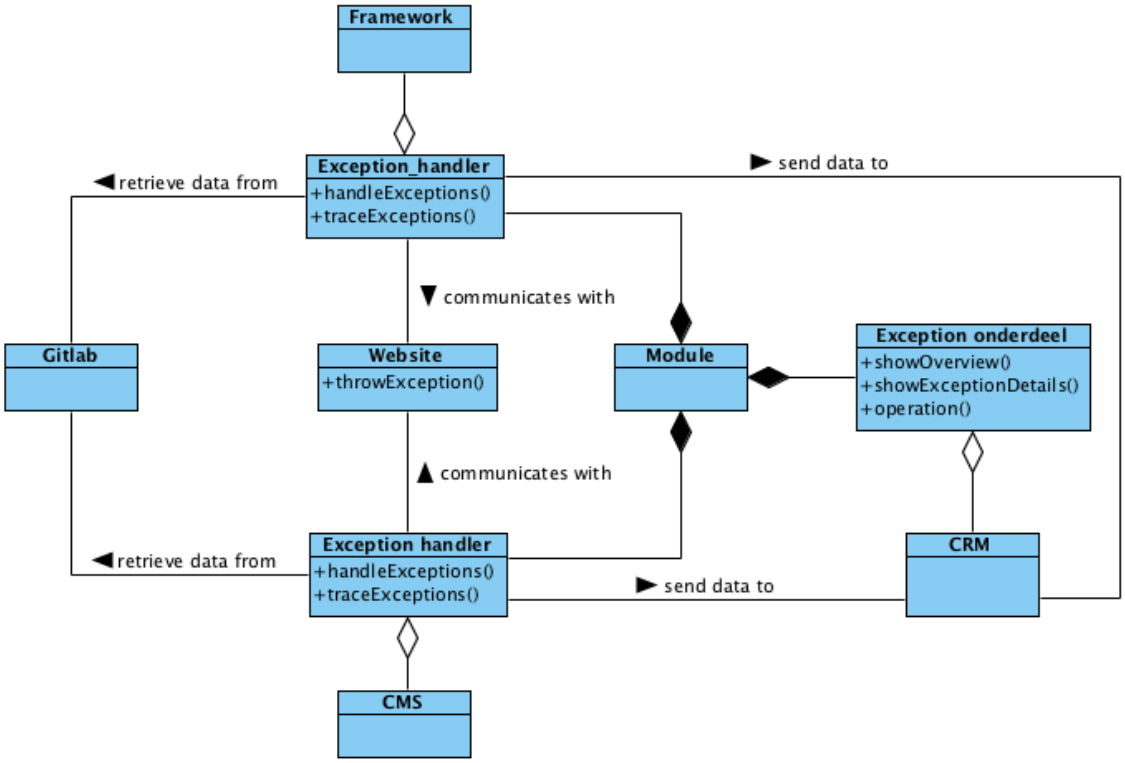
De module bestaat uit twee exception handlers (één in het framework en één in het CMS). Deze exception handlers handelen beide exceptions af, maar in verschillende omgevingen. Deze handlers bestaan al, maar zal ik herschrijven om aan het nieuwe proces te voldoen.

De exception handler in het framework communiceert met de websites om de gevonden exceptions in de frontend vast te stellen. Daarnaast communiceren ze met Gitlab om de tracering van de exceptions tot stand te brengen.

CRM exception onderdeel

De module uit een onderdeel in het CRM. Dit onderdeel bevat het overzicht van alle exceptions waarin deze inzichtelijk en beheerbaar worden gemaakt. De exceptions kunnen in een overzicht worden bekeken, maar ook op detail niveau. Daarnaast kunnen ze hier worden beheerd.

Afbeelding 11: Context diagram proof of concept (module)



De scope

Om een beeld te krijgen van de scope van de systemen waarin ik zal werken heb ik een diagram gemaakt (zie Afbeelding 12). De structuur die weergeven is in het diagram is de standaard structuur van het bedrijf waarin het proof of concept zal integreren.

De afbeelding laat zien dat de ‘exception_handler’ class is opgenomen in het CMS en het in het framework. Deze class achterhaalt de exceptions die voorkomen op de websites en verzamelt alle data van de exceptions en de tracering. De class communiceert met het CRM en Gitlab via een API.

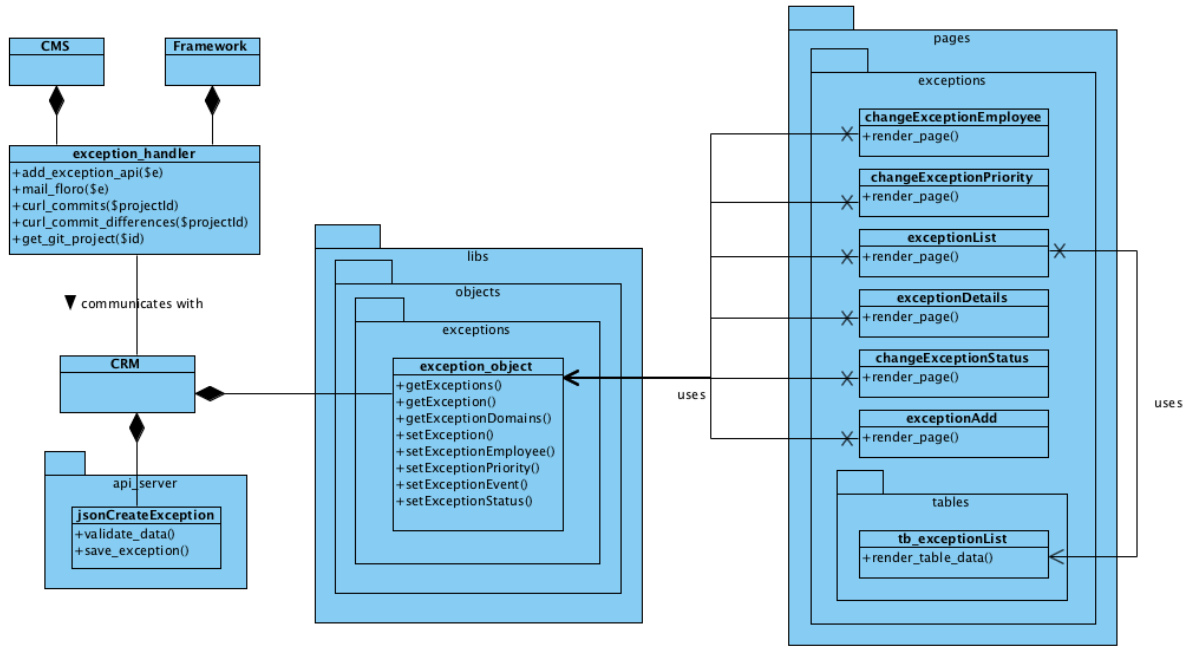
Het CRM bevat een map ‘api_server’ waarin een JSON bestand staat die de data van de exceptions valideert en opslaat in de database. Dit bestand wordt aangeroepen in een API call in de ‘exception_handler’ class.

Verder bevat het CRM een map ‘libs’ waarin een map ‘objects’ staat. In de map ‘objects’ staan de mappen van alle objecten van het systeem, zo ook de map ‘exceptions’. De map ‘exceptions’ bevat het exceptions object wat alle functies van de exception bevat. Dit object is gelijk aan een ‘Model’ volgens de MVC structuur.

Tot slot is er in het CRM een map ‘pages’. In de map ‘pages’ staan alle pagina’s van het CRM. Hierin staat een map ‘exceptions’ waarin alle pagina’s van de exception module staan. Deze pagina’s gebruiken de functies die geschreven staan in het exception object. Verder is er volgens de interne structuur een map ‘tables’ waarin alle tabellen staan die de data renderen.

De structuur is niet helemaal object georiënteerd, maar dit is iets wat ik niet kan veranderen. Ik bouw de module zoals dit gepast is in de interne structuur.

Afbeelding 12: Scope van de exception handler en het werkgebied



10.3 Geplande iteraties

De volgende iteraties heb ik gepland op basis van de benodigde onderdelen van het proof of concept die naar voren zijn gekomen in dit hoofdstuk.

Iteratie 1: Afhandelen van exceptions in de interne systemen
De exceptions moeten worden afgehandeld in verschillende systemen. Zo moeten ze worden afgehandeld in het framework, het CMS en het CRM.

Iteratie 2: Traceerbaar maken van de exceptions
Voor het efficiënt kunnen oplossen van de exceptions moeten deze traceerbaar gemaakt worden d.m.v. een connectie met Gitlab.

Iteratie 3: Tonen van een overzicht in het CRM
In het CRM zal ik een onderdeel maken voor exceptions onder de categorie ‘Support’. Dit onderdeel regelt de inzichtelijkheid en de beheerbaarheid van de exceptions.

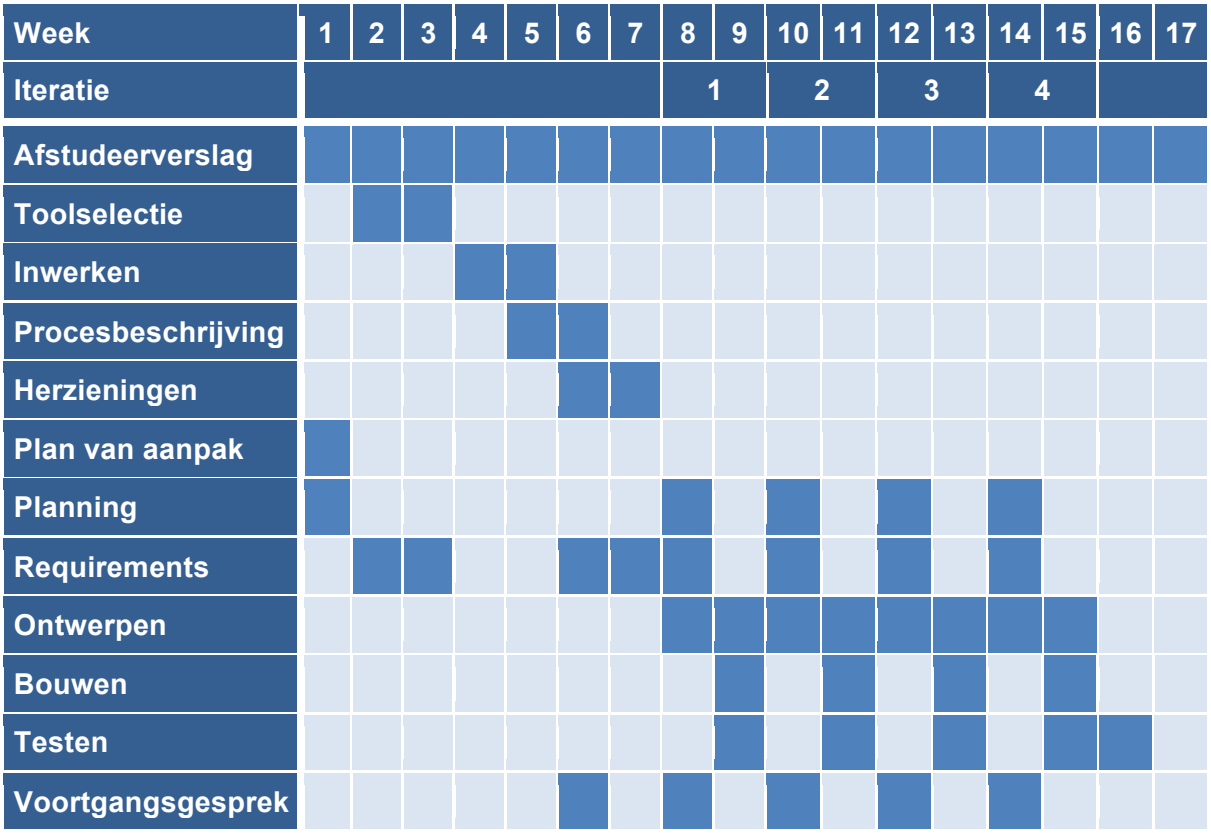
Iteratie 4: Beheerbaar maken van exceptions
De beheerbaarheid van de exceptions in het CRM onderdeel zal ik tijdens deze iteratie beheerbaar maken.

10.4 Planning

Doordat de opdracht is gewijzigd, is vanzelfsprekend ook de planning gewijzigd. Onderstaand is de nieuwe planning te vinden.

Er zijn geen drastische veranderingen ten opzichte van de orginele planning. De werkzaamheden ‘procesbeschrijving’ en ‘herzieningen’ zijn toegevoegd.

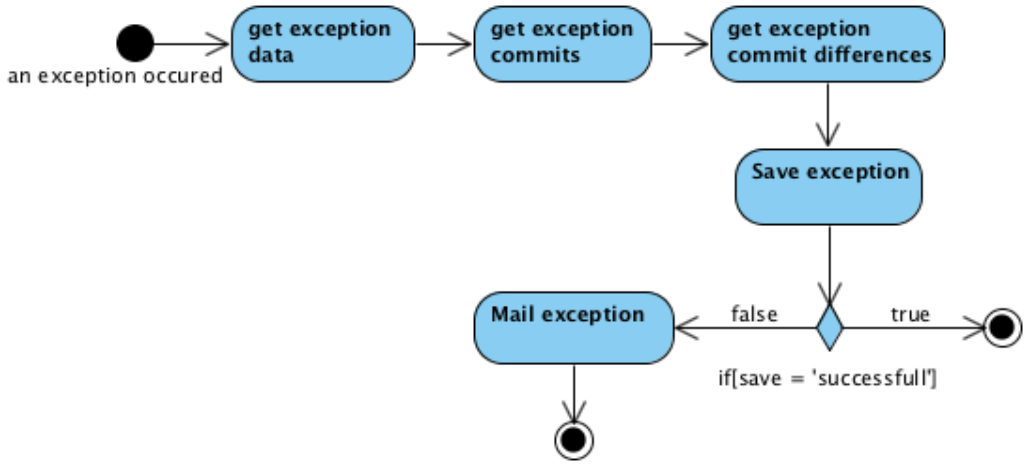
Verder zijn de iteraties nu vastgelegd. Dit betekend dat deze nauwkeuriger kunnen worden ingepland. Er zijn vier iteraties ingepland in tegenstelling tot vijf in de orginele planning.


Week
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
Iteratie
1
2
3
4
Afstudeerverslag
Toolselectie
Inwerken
Procesbeschrijving
Herzieningen
Plan van aanpak
Planning
Requirements
Ontwerpen
Bouwen
Testen
Voortgangsgesprek

11 Iteratie 1: Afhandelen van exceptions in de interne systemen

Exceptions zullen worden toegevoegd aan de database via een API call vanuit de ‘exception_handler’ class. De exceptions worden opgenomen in een eigen tabel, waarin zowel alle informatie van de exception staat, als de informatie van de tracering en beheerbare attributen. Wanneer de API call niet mocht lukken, wordt de exception per mail naar de supportafdeling gestuurt als fallback plan (zie Afbeelding 13).

Afbeelding 13: Diagram toevoegen van exceptions



11.1 Requirements

Omdat er nu bekend is hoe de afhandeling van de exceptions zal verlopen kunnen er requirements aan het requirementsrapport worden toegevoegd. De volgende requirements heb ik vastgelegt.

Functionele requirements

ID	Beschrijving	BR	UC
F05	De module neemt de exceptions op in de database via een API call.	B06	-
F06	Exceptions worden opgeslagen in een aparte exceptions tabel in de database	B06	C02
F07	Het opslaan van exceptions bevat een fallback plan, waarin de exceptions gemailt worden indien deze niet via een API call kunnen worden opgeslagen.	-	-

11.2 Toevoegen van exceptions via een API call

De exceptions moeten via API calls opgeslagen worden in de database van CusWeb om deze te kunnen benaderen vanuit de diversen onderdelen van de module in de interne systemen. Exceptions zullen niet meer worden opgeslagen als tickets. Deze keuze heb ik gemaakt omdat het de support processen verstoord als er vele exception mails voorbij komen.

Ik heb ervoor gekozen om de exceptions op te slaan via een API call, omdat het de standaard is om voor het afhandelen van data in de interne systemen een API te

gebruiken. Deze methode is dus een gepaste manier van het afhandelen van de data om de consistentie in de interne systemen te behouden.

Via een API call vanuit de exception handlers in het framework en het CMS naar CusWeb worden de exceptions opgeslagen in de database van CusWeb. De API call verloopt via een PHP cURL functie. Via de API call wordt het 'createexception' JSON bestand aangeroepen in de CusWeb API. Deze API call stuurt de benodigde data mee. Het JSON bestand zorgt er vervolgens voor dat de data die is meegestuurd wordt gevalideerd en opgeslagen. Het opslaan gebeurt aan de hand van een query naar de database van CusWeb.

De exception
De gepostte data die via de API call verstuurd wordt is de data die de exceptions tabel ontvangt. De exceptions tabel ziet er als volgt uit (zie Afbeelding 14).

Afbeelding 14: Create statement voor de tabel 'exceptions'

```
CREATE TABLE `exceptions` (
  `exception_id` int(11) NOT NULL AUTO_INCREMENT,
  `date_created` datetime DEFAULT NULL,
  `domain` varchar(150) DEFAULT NULL,
  `git_project` text,
  `status` enum('unresolved','being solved','resolved') DEFAULT
'unresolved',
  `priority` enum('high','medium','low') DEFAULT NULL,
  `employee_id` int(11) DEFAULT NULL,
  `exception_class` varchar(45) DEFAULT NULL,
  `exception_code` varchar(45) DEFAULT NULL,
  `exception_msg` text,
  `exception_file` varchar(150) DEFAULT NULL,
  `exception_line` varchar(45) DEFAULT NULL,
  `commit` text,
  `reminder_sent` tinyint(4) DEFAULT '0',
  PRIMARY KEY (`exception_id`)
) ENGINE=InnoDB AUTO_INCREMENT=50 DEFAULT CHARSET=utf8;
```

Exception data
Er wordt uiteraard data van de exception opgeslagen, dit zijn een ID, de datum van optreden, het domein waarop de exception optreed en de technische informatie. Deze technische informatie omvat de exception class, code, message, file en line.

Git project
Er wordt een link opgeslagen van het Git project van de website. Niet alle websites hebben echter een Git project. Om te achterhalen of de website een Git project heeft, heb ik een functie aangemaakt get_git_project();. Deze functie maakt wederom verbinding met de CusWeb API en roept hierbij een JSON bestand aan genaamd 'getGitProject'. In het JSON bestand wordt er via een query het ID van het Git project opgehaald uit de tabel 'gitProjects' in de CusWeb database.

Met het ID van het Git project haal ik vervolgens de link op van het betreffende project via aan API call naar Gitlab. Ik heb hiervoor een functie gemaakt die via de API call een functie aanroept die een project ophaalt aan de hand het het ID. Uit de JSON response van de API call, sla ik de naam en de link van het project op in een array, om deze vervolgens op te slaan in JSON formaat.

Beheerbare attributen
De exception bevat een aantal beheerbare attributen zoals een status, prioriteit, werknemer en reminder.

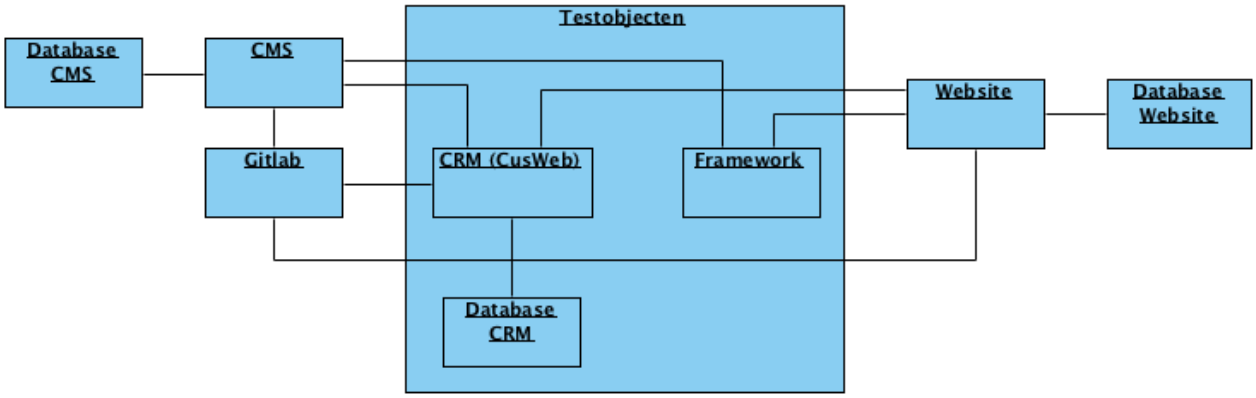
Traceerbaarheid
Tot slot wordt de getraceerde data van de exception opgeslagen in het attribuut ‘commit’. Deze traceerbaarheid word tot stand gebracht in de volgende iteratie.

Fallback plan
Indien het toevoegen van de exceptions via een API call onverhoopt verstoort raakt willen we alsnog zorgen dat de exceptions worden genoteerd om deze te kunnen oplossen. Om deze reden heb ik een fallback plan bedacht voor het toevoegen van de exceptions. Wanneer de API call niet lukt wordt er een mail verstuurd naar de support afdeling waar alle technische informatie van de exception in is verwerkt. Op deze manier worden de exceptions alsnog geregistreerd. De exceptions kunnen later indien gewenst nog handmatig worden toegevoegd aan de database d.m.v. een button ‘add exception’ die geplaatst zal worden in het te bouwen overzicht in het CRM. Op deze functionaliteit wordt verder ingegaan in iteratie 3.

11.3 Testen
Voor de afhandeling van de data en samenwerking met de verschillende interne systemen voer ik een gerichtte ketentest uit. Deze test richt zich op het specifieke onderdeel van data overdracht tussen de onderdelen van de module in de diversen interne systemen. Deze test zal vaststellen of de systemen dusdanig gekoppeld zijn dat de data feilloos kan worden overgespeeld tussen de diversen systemen.

Testbasis
In de testbasis geef ik aan welke systemen getest worden en welke buiten het beschouwingsgebied liggen. De systemen die weergeven worden in het het blok ‘testobjecten’ vallen binnen het testgebied en de rest daarbuiten (zie Afbeelding 15).

Afbeelding 15: De objecten binnen en buiten de testbasis

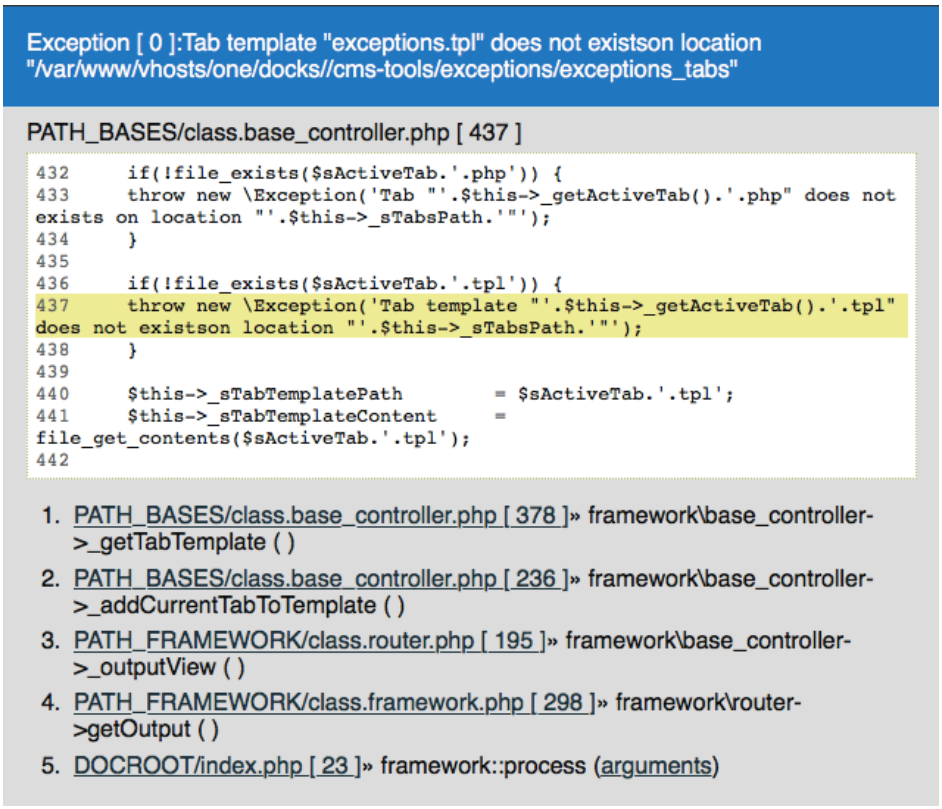


Teststrategie
De verwachting is dat de data volledig en correct opgeslagen kan worden via de API calls en dat de data feilloos overgespeeld kan worden tussen de diverse onderdelen van de module in de interne systemen.

Testdata
Om de data te kunnen testen heb ik een aantal exceptions veroozaakt in mijn ontwikkelomgeving. Dit doe ik door een template bestand tijdelijk te hernoemen, waardoor

de view deze niet kan vinden en een exception geeft. Met deze exception kan ik testen (zie Afbeelding 16).

Afbeelding 16: De exception die is veroorzaakt en wordt gebruikt als test data



Uitvoering
De test wordt uitgevoerd in mijn ontwikkelomgeving waarin ik een checkout van het CRM, CMS en mijn eigen floro website heb.

De eerste stap van de test is al uitgevoerd bij het creëren van de testdata, namelijk het opwekken van een exception. Door in mijn ontwikkelomgeving naar de pagina op de website te navigeren met de missende template werd er een exception opgewekt. De exception kwam in zijn geheel binnen als exception in de database (zie Afbeelding 17). De gegevens van de tracering en beheerbare attributen ontbreken nog. Deze worden getest nadat de tracering en beheerbaarheid is gebouwd in latere iteraties.

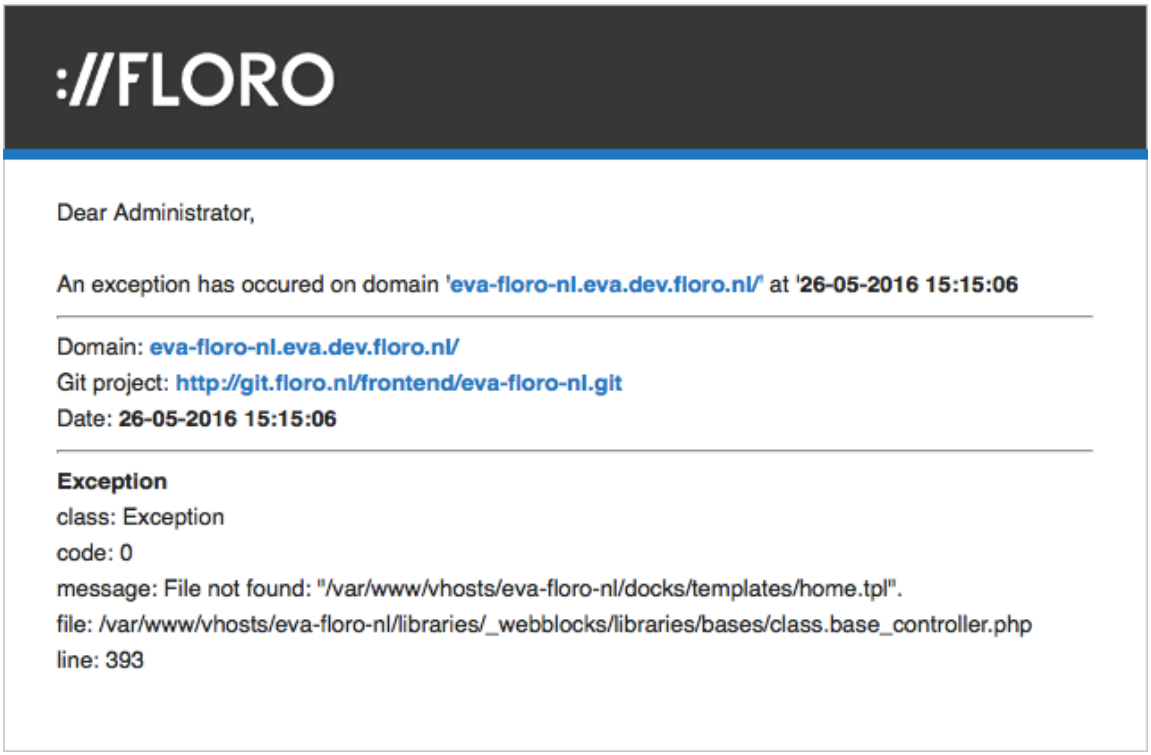
Afbeelding 17: Database records van de exceptions

exc...	date_created	domain	git_project	status	priority	employee_id	exceptio
57	2016-05-20 1...	one.eva.dev.flo...	127	NULL	NULL	NULL	Excepti
58	2016-05-20 1...	one.eva.dev.flo...	127	NULL	NULL	NULL	Excepti
59	2016-05-20 1...	eva-floro-nl.ev...	127	NULL	NULL	NULL	Excepti
60	2016-05-24 1...	one.eva.dev.flo...	127	NULL	NULL	NULL	Excepti
61	2016-05-24 1...	eva-floro-nl.ev...	127	NULL	NULL	NULL	Excepti
62	2016-05-24 1...	one.eva.dev.flo...	127	NULL	NULL	NULL	Excepti
63	2016-05-24 1...	one.eva.dev.flo...	127	NULL	NULL	NULL	Excepti

Er kan geconcludeerd worden dat de add_exception functie werkt en de data succesvol opslaat via de API call. Ook het ophalen van het Git project is succesvol verlopen. Hiermee wordt tevens gevalideerd dat het ophalen van het Git project naar behoren functioneert.

Het tweede scenario van de test is om bewust een fout te creëren in de API afhandeling, waardoor de fallback mail getriggert wordt. Door een cruciale regel code uit de functie weg te halen en de mail naar mijn eigen Floro mail te verwijzen kan ik dit scenario testen. Na het herladen van de pagina zou de exception niet meer als exception opgenomen moeten worden in de database, maar zou ik een mail moeten ontvangen waarin de gegevens van de exception staan. Na het herladen van de pagina heb ik ondervonden dat dit ook gebeurd (zie Afbeelding 18).

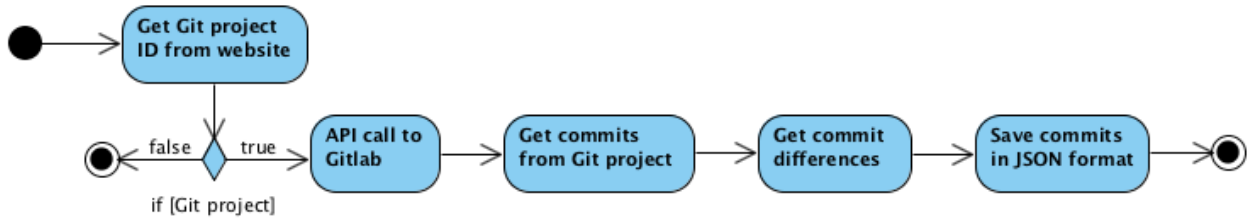
Afbeelding 18: Fallback email met de exception



12 Iteratie 2: Traceerbaar maken van de exceptions

Gezien we niet alleen de exceptions willen overzien, maar ook het proces van het oplossen van deze exceptions willen verbeteren, wordt er een tracering tot stand gebracht die exceptions herleidt naar de oorzaak via versiebeheer. Wanneer de website waarop een exception optreed een Git project heeft, wordt van dit Git project de laatste commit opgehaald voor de optreding van de exception. Deze commit wordt in JSON formaat opgeslagen in het 'commit' attribuut van de exception in de database (zie Afbeelding 19).

Afbeelding 19: Diagram met globaal overzicht van de traceerbaarheid



De functionaliteit voor het achterhalen van de laatst uitgevoerde commit voor de optreding van de exception wordt later nader toegelicht en is niet opgenomen in het diagram, omdat dit te specifiek is voor dit globale overzicht.

12.1 Requirements

Met betrekking tot de traceerbaarheid zijn er een aantal requirements opgesteld.

Functionele requirements

ID	Beschrijving	BR	UC
F08	De module kan exceptions die optreden op websites die draaien op het interne framework met een Git project herleiden naar commits.	B05	-
F09	De module communiceert met Gitlab via een API.	B05	-
F10	De module haalt van websites die draaien op het interne framework die een Git project hebben d.m.v. een API request naar Gitlab de laatst uitgevoerde commit op vòòr de optreding van een exception.	B05	-
F11	De module kan d.m.v. een API request naar Gitlab de wijzigingen van de laatst uitgevoerde commit voor de optreding van een exception ophalen	B05	-

12.2 Verbinding met Gitlab

Voor de tracering breng ik een verbinding tot stand met de versiebeheer tool Gitlab. Gitlab wordt benaderd via de API van de versiebeheertool en kan benaderd worden via http://git.floro.nl. Voor het ophalen van commits van een specifiek project moet het pad naar dit project worden benaderd. Daarnaast is het voor het gebruik van de API authenticatie nodig. Hiervoor wordt private token gebruikt. Een request naar Gitlab ziet er als volgt uit:

```
http://git.floro.nl/api/v3/projects/'. $sProjectId
.'/repository/commits?private_token=ZVsEmbMb726sdZR66MFX
```

12.3 Ophalen van commits

Voor het ophalen van de commits heb ik een functie aangemaakt in de ‘exception_handler’ class. Via de cURL functie van PHP maak ik verbinding met Gitlab en benader het project wat hoort bij de website waarop de exception is opgetreden. Van dit project roep ik de functie aan voor het ophalen van de commits.

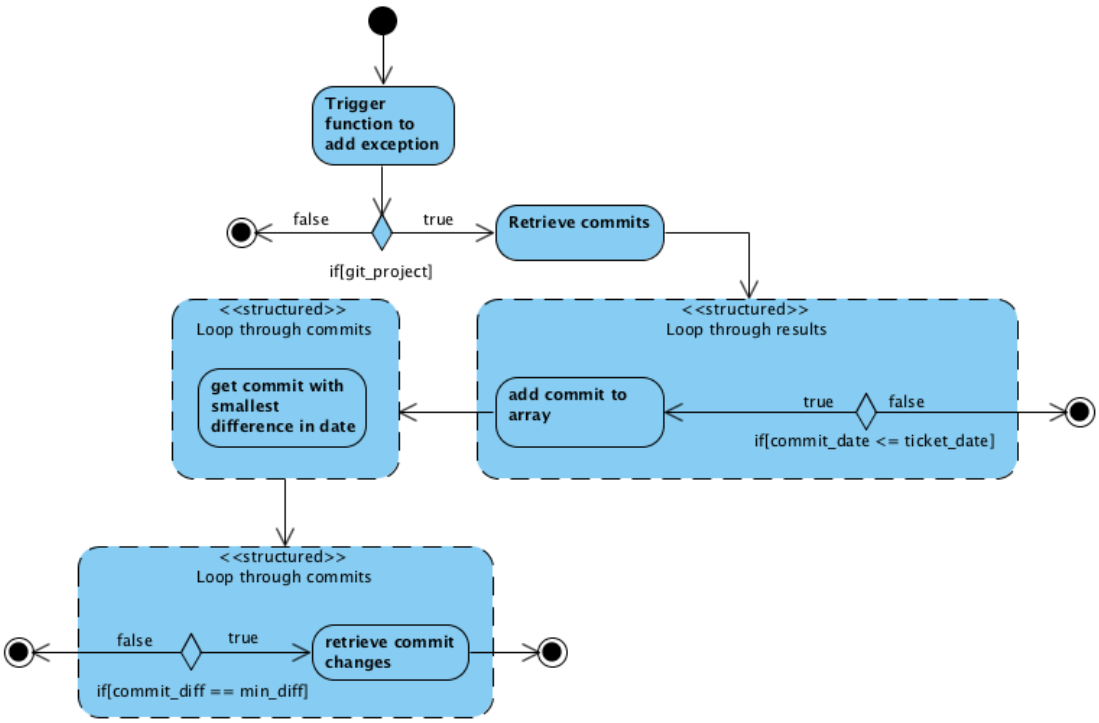
Wanneer de commits zijn opgehaald zorg ik ervoor dat er over de resultaten geloopt wordt, waarna er wordt gekeken of de datum van de commit vòòr de datum van de opgetreden exception ligt. Zo worden de commits die zijn uitgevoerd na het optreden van de exception eruit gefilterd, omdat deze nooit de exception veroorzaakt kunnen hebben.

Van de commits zijn uitgevoerd na het optreden van de exception, sla ik de relevante gegevens op in een array. Deze gegevens zijn de datum van de commit, de titel, de naam en het e-mail adres van de persoon die de commit heeft uitgevoerd en het bericht wat is toegevoegd bij het uitvoeren van de commit.

Vervolgens maak ik een loop, die door de array met gefilterde commits loopt om zo de commit over te houden die als laatst is uitgevoerd voor de optreding van de exception. Dit gebeurt door het verschil in commit datum en exception datum te berekenen en de commits met het kleinste verschil over te houden. Omdat dit mogelijk niet maar één exception is maar meerdere wordt er wederom door de array met gefilterde commits heen geloopt en wordt er gekeken of er nog meer commits zijn die net zo’n klein verschil hebben als de dichtsbijzijnde commit.

Hetgeen wat overblijft uit deze filtering van commits zijn een of meerdere commits die als laatst zijn uitgevoerd voor de optreding van de exception. Omdat de exception na deze commit is opgetreden is het waarschijnlijk dat er een wijziging in deze commit zit die de exception heeft veroorzaakt. Van deze commit(s) zullen de wijzigingen in de code worden opgehaald (zie Afbeelding 20).

Afbeelding 20: Ophalen Gitlab commits



12.3.1 Ophalen van wijzigingen in de code

Om de wijzigingen in de code op te halen van de commit(s) voor de optreding van de exception voer ik nog een request uit naar de Gitlap API. Ditmaal wordt de functie voor het ophalen van verschillen in de commits aangeroepen.

Deze verschillen laten zien welke code en bestanden er zijn gewijzigd ten opzichte van de vorige commit. De response van Gitlab is in JSON formaat en ziet er als volgt uit:

```
[
  {
    "diff": "--- a/docks/templates/home.tpl +++ b/docks/templates/home-
lol.tpl",
    "new_path": "docks/templates/home-lol.tpl",
    "old_path": "docks/templates/home.tpl",
    "a_mode": "100644",
    "b_mode": "100644",
    "new_file": false,
    "renamed_file": 1,
    "deleted_file": false
  }
]
```

Deze response geeft bijvoorbeeld aan dat het bestand ‘home.tpl’ is verwijderd en ‘home-lol.tpl’ is toegevoegd. In het attribuut ‘renamed_file’ zie je dat het gaat om een hernoeming van het bestand.

De response geef ik mee tijdens het opslaan van de exception. De commit wordt in zijn geheel opgeslagen met het datatype ‘TEXT’ in het attribuut ‘commit’ van de exception. Deze data zal later worden meegenomen om een overzicht te genereren van de exception met de getraceerde commit(s) om de exception efficiënt te kunnen oplossen.

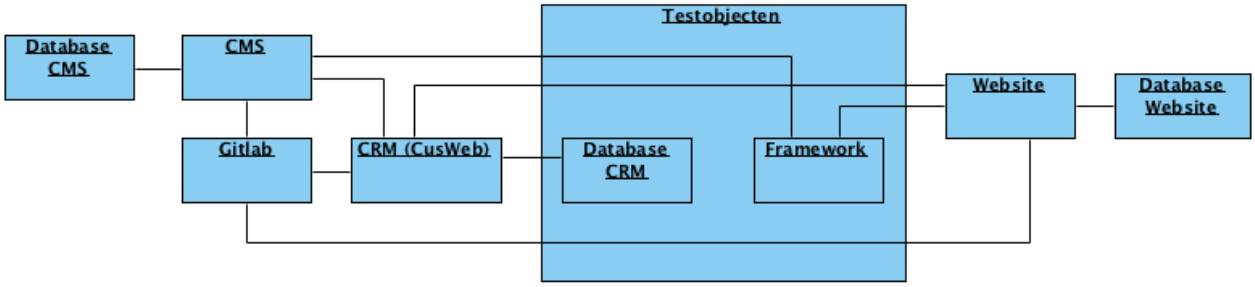
12.4 Testen

Het is belangrijk dat de juiste informatie uit Gitlab wordt opgehaald. Tijdens het testen zal ik de outputted data testen en zorgen dat ik de juiste data tot mijn beschikking heb en de minder belangrijke data buiten beschouwing laat. De functionaliteit van dit onderdeel wordt getest met een moduletest.

Testbasis

De testbasis geeft aan welke systemen getest worden en welke buiten het beschouwingsgebied liggen. De systemen die weergeven worden in het het blok ‘testobjecten’ vallen binnen het testgebied (zie Afbeelding 21).

Afbeelding 21: Objecten binnen en buiten de testbasis



Teststrategie
De verwachting is dat er zonder problemen een verbinding met Gitlab kan worden gemaakt en de commits en wijzigingen van een project opgehaald kunnen worden. Daarnaast moet de opgehaalde data succesvol worden opgeslagen in de database van CusWeb.

Testdata
De data die gebruikt wordt tijdens deze test is afkomstig van mijn eigen test website. In deze website heb ik een exception veroorzaakt en van het Git project van deze website heb ik de commits opgehaald.

Uitvoering
De testen vinden wederom plaats in mijn ontwikkelomgeving waarin een checkout van het CRM, CMS en mijn eigen floro website zijn gemaakt. In het framework van deze omgevingen, waar de exception handler zich bevindt kan ik de data testen door data te outputten met de functies. Dit doe ik op de pagina via de var_dump(); functie van PHP. Wanneer de connectie met Gitlab niet goed is krijg ik hiervan een melding. Op deze manier kan ik de code aanpassen tot deze werkt. Uiteindelijk kon ik de verbinding succesvol laten verlopen en de juiste data ophalen.

Het wegschrijven van deze data naar de database van CusWeb verliep zonder problemen. De exceptions worden nu opgeslagen met zowel de data van de exceptions als de getraceerde data in JSON formaat.

13 Iteratie 3: Tonen van een overzicht in het CRM

Voor het overzicht en het beheer van de exceptions is ervoor gekozen deze in het CRM te verwerken. Deze keuze is gemaakt, omdat het CRM het centrale platform is waarin alle interne zaken van het bedrijf beheerd worden. Het integreren van het overzicht in het CRM is tevens een requirement van de opdrachtgever.

13.1 Requirements

Er zijn een aantal requirements toegevoegd aan het rapport betreft de afhandeling van fouten. De use cases behorende bij de user requirements zijn opgenomen in Bijlage E.

User requirements

ID	Beschrijving	BR	UC
U03	De gebruiker wil het overzicht van exceptions kunnen filteren op domein, status, prioriteit en werknemer	B06	C03
U04	De gebruiker wil de details van een exception kunnen bekijken op een detailpagina	B06	C01

Functionele requirements

ID	Beschrijving	BR	UC
F12	Er wordt een exceptions onderdeel gebouwd in het CRM onder de categorie ‘Support’.	B06	C01
F13	Er wordt een overzicht getoond waarbij een filter kan worden geselecteerd voor het filteren op domein, status, prioriteit en werknemer	B06	C01, C03

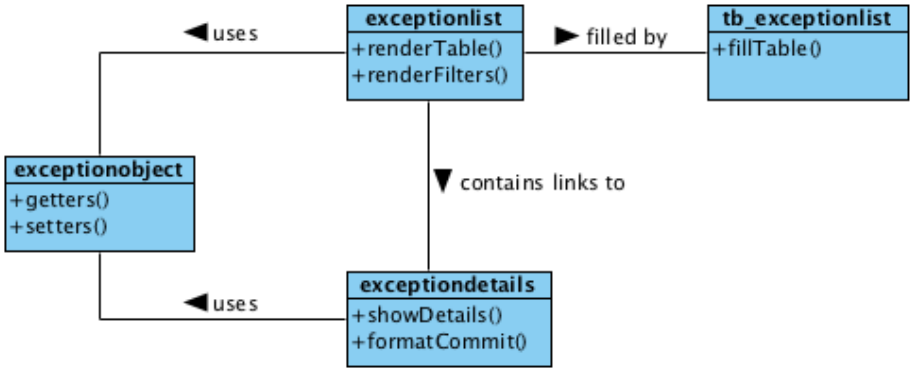
13.2 Bouwen van een nieuwe module voor exceptions

In het CRM heb ik een onderdeel voor exceptions gebouwd onder de categorie ‘Support’. Ik heb ervoor gekozen om de exceptions onder support te plaatsen omdat dit het meest toepasselijk is. Het oplossen van exceptions is namelijk een onderdeel van de support van de websites.

Voor het aanmaken van het exceptions onderdeel heb ik gekeken naar de andere onderdelen in het CRM. Voor het onderdeel moet ik de benodigde pagina’s aanmaken in de database. De database van CusWeb bevat een tabel ‘pages’. In deze tabel zijn alle pagina’s van CusWeb opgeslagen.

De pagina’s die ik heb aangemaakt voor het overzicht zijn: ‘exceptionlist’, ‘tb_exceptionlist’, ‘exceptiondetails’ en ‘exceptionadd’. De pagina’s ‘exceptionlist’ en ‘tb_exceptionlist’ maken gezamelijk het overzicht van de exceptions. De pagina ‘exceptiondetails’ is de pagina met alle details van de exception en de pagina ‘exceptionadd’ is de pagina waarop handmatig exceptions kunnen worden toegevoegd. Voor de pagina’s uit de database moet ik ook corresponderende pagina’s in de code aanmaken. Dit zijn de bestanden met code die de HTML output renderen. De pagina’s maken gebruik van het exception object waarin ik de functionaliteit van de exceptions zal schrijven (zie Afbeelding 22).

Afbeelding 22: Rendering van de overzicht pagina's in de exception module



Het overzicht
Het bestand 'exceptionlist' is het geraamte voor de tabel uit het overzicht. In dit bestand zijn de filters aangegeven, is de query geschreven voor het ophalen van de data en zijn de te tonen kolommen aangegeven. De tabel wordt vervolgens gevuld door het bestand 'tb_exceptionlist'. Dit bestand bevat alleen een foreach loop waarin er wordt geloopt over de data die opgehaald wordt in het het 'exceptionlist' bestand. Het bestand vult vervolgens de data in de tabel in. Het principe van het apart behandelen van de content van de tabel heb ik overgenomen van de structuur van bestaande modules. Op deze manier blijft er consistentie in de manier van coderen.

De detailpagina
De pagina 'exceptiondetails' haalt alle data van de gekozen exception uit het overzicht op. De commit van de exception moet leesbaar worden gemaakt, omdat dit is opgeslagen in JSON formaat en de wijzigingen allemaal leestekens bevatten. Hiervoor is er een button 'view commit on Gitlab' toegevoegd, waarbij ik de link naar de commit gegenereerd heb aan de hand van een aantal losse attributen. De button was echter niet mijn eerste keuze.

Mijn eerste gedachte was om de commit te ontcijferen en zelf te formatteren. De commits hebben echter zoveel leestekens en regels dat dit niet mogelijk was binnen de resterende tijd van de iteratie en de opdracht. Mijn volgende optie was om een formatter te downloaden. Na wat onderzoek ben ik er achter gekomen dat dit niet beschikbaar is voor het formaat waarin Gitlab de commits opslaat. De volgende optie die ik had was de commit via een iframe toevoegen aan de pagina. Dit was echter niet mogelijk door een restrictie van Gitlab waardoor de pagina niet als iframe mag worden gebruikt. De eerstvolgende mogelijkheid was het plaatsen van een button met de link naar de commit in Gitlab om deze daar te kunnen bekijken. Gezien de tijd was dit de beste oplossing voor het leesbaar tonen van de commit behorende bij de exception.

Handmatig toevoegen van exceptions
Op de 'exceptionadd' pagina staat een formulier waarmee handmatig een exception kan worden toegevoegd. Dit kan handig zijn wanneer het toevoegen via de API call niet is gelukt en je de exception die als fallback via de mail binnen is gekomen alsnog wilt opnemen in de module.

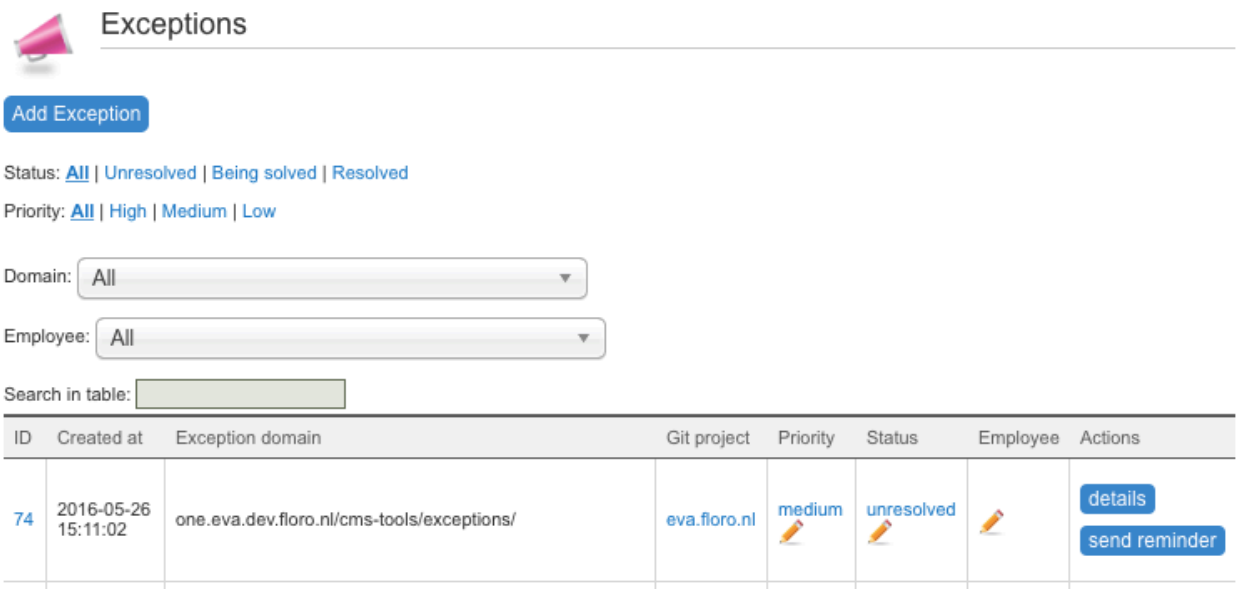
13.3 Relevante data voor in het overzicht
Omdat het overzicht beperkte ruimte heeft voor het tonen van data zijn de meest algemene punten eruit gehaald om te tonen. De informatie die niet in het overzicht past wordt getoond op een detailpagina. Deze pagina bevat alle informatie van de exception.

De gegevens die worden opgenomen in het overzicht zijn: het ID van de exception, de datum waarop de exception is opgetreden, de website waarop de exception is opgetreden, een link naar het Git project behorende bij de website (indien deze een Git project heeft), de prioriteit, de status en de werknemer waaraan de exception is toegekent. Verder is er een kolom met acties die kunnen worden uitgevoerd zoals het bekijken van de detailpagina en het sturen van een reminder (zie Afbeelding 23). De beheerbare attributen prioriteit, status en werknemer zijn alvast opgenomen in het overzicht. De gegevens die getoond worden voor deze attributen zijn standaard waarden die ik heb opgegeven in de database, omdat de functionaliteit voor het beheren van de exceptions nog niet gebouwd is. Deze functionaliteit wordt in de volgende iteratie geschreven.

De keuze voor de gegevens in het overzicht heb ik gemaakt door me te verplaatsen in de gebruiker van de module. Wanneer de gebruiker exceptions wil bekijken en beheren zal hij of zij willen weten wanneer de exception is opgetreden, waar hij is opgetreden en of het een belangrijke exception is. Verder wil hij ook de status van de exception gemakkelijk kunnen zien en wil hij weten welke werknemer bezig is met het oplossen van de exception.

De overige informatie is vooral relevant voor de gebruiker die bezig is met het oplossen van de exception. Deze gebruiker zal op de detailpagina van de exception de specificaties en de commit details van de exception kunnen bekijken. De detailpagina bevat een compleet overzicht van alle gegevens van de exception wat helpt bij het oplossen van deze exception.

Afbeelding 23: Overzicht van exceptions in het CRM



13.4 Het filteren van het overzicht

In het overzicht heb ik ervoor gezorgd dat er gefilterd kan worden op status, prioriteit, werknemer en domein. Ik heb gekozen voor het inbouwen van deze filters omdat er efficiënt gefilterd moet kunnen worden wanneer er veel exceptions in het overzicht zijn.

De filters heb ik bepaald door me wederom in te leven in de gebruiker. Zo kan het voor de gebruiker interessant zijn om alleen de exceptions te tonen die nog niet opgelost zijn of exceptions die op het moment worden opgelost.

Daarnaast is het handig een overzicht te kunnen opvragen van alleen de belangrijkste exceptions met een prioriteit ‘high’, omdat deze snel bekeken en opgelost moeten kunnen worden.

Verder is er gekozen voor de filter domein, omdat het van belang kan zijn alleen de exceptions te overzien van een bepaalde website, wanneer er bijvoorbeeld contact is met de klant van de betreffende website.

Er is gekozen voor de filter werknemer, omdat een werknemer waaraan een exception is toegekent snel wil kunnen zien welke exceptions hij of zij moet oplossen.

13.5 Testen

Het is belangrijk dat de juiste gegevens in het overzicht staan en dat het voorziet in de behoefte van de gebruiker. Ik heb scenario’s opgesteld die uitgevoerd zullen worden door een aantal toekomstige gebruikers van de module. Hieruit kan ik leiden of het overzicht de juiste werking heeft en gepast is voor de mogelijke omstandigheden waarin het gebruikt zal worden. Voor het testen van deze iteratie wordt er een gebruikersacceptatietest (GAT) op de ontwikkelomgeving uitgevoerd. Verder moet er worden getest of het handmatig toevoegen van de exception werkt. Dit zal ik testen door exceptions toe te voegen en te controleren of deze correct worden opgeslagen in de database.

Testbasis
Tijdens deze test valt alleen het overzicht in het CRM binnen het beschouwingsgebied.

Teststrategie
Het is de bedoeling dat de gebruikers met het overzicht kunnen werken. Daarom wil ik er met behulp van de GAT achterkomen of het overzicht voldoet aan de verwachtingen van de gebruiker en het gebruik efficiënt is en voorzich spreekt.

Testdata
De data uit het overzicht is afkomstig van door mij zelf gegenereerde exceptions. Deze exceptions zijn getriggert op mijn development container in de checkout van het CMS en mijn eigen Floro website. Daarnaast heb ik in de database de beheerbare attributen van de exceptions handmatig ingevult zodat het filteren ook getest kan worden, ondanks dat de functionaliteit hiervoor nog niet gebouwd is.

Uitvoering
Tijdens deze test heb ik een tweetal werknemers het overzicht laten gebruiken. Er is gekozen voor een developer en een projectmanager om twee verschillende kanten van het gebruik van het proof of concept te achterhalen. Er is gekozen voor twee personen omdat deze personen de twee groepen representeren die met de module zullen omgaan, namelijk managers en developers.

Voor de test moeten zij scenario’s uitvoeren die in het dagelijksleven zouden kunnen voorkomen. De focus hierbij is: ‘is het overzichtelijk en is de juiste data beschikbaar?’.

Ik heb de testpersonen ingeleid in het proof of concept. Vervolgens heb ik ze een aantal scenario’s laten uitvoeren. De scenario’s en de bevindingen zijn onderstaand genoteerd.

Scenario 1: Je bent toegekent aan een exception om deze op te lossen.	
De verwachting	Het is de bedoeling dat de gebruiker in het CRM navigeert naar het kopje ‘exceptions’ onder de categorie support. Vervolgens kan de

	werknemer het overzicht filteren, door zichzelf te kiezen in de werknemer filter. Vervolgens moet de werknemer de detailpagina van de exception bekijken om de gewijzigde gegevens te kunnen bekijken.	
De uitvoering	Testpersoon 1	**Naam**: Edwin Vogelaar **Functie**: Frontend developer **Bevindingen**: de testpersoon kreeg als eerste indruk dat hij moet zoeken naar een website. Na uitleg over de mail die hij ontvangt na de toekenning met de exception specificaties zoals de domein, werd duidelijk dat hij in het overzicht kon zoeken op zijn naam. Vervolgens bekeek de testpersoon de detailpagina. **Verbeteringen**: Het overzicht kan duidelijker weergeven worden. Op het eerste gezicht is het zonder uitleg niet duidelijk waar het overzicht over gaat en wat je er kan doen. Een eenmalige korte inleiding zou hier mogelijk een oplossing voor bieden. Verder merkte de testpersoon op dat de kolom ‘exception domain’ wellicht geen duidelijke benaming is. **Opmerkingen**: -
	Testpersoon 2	**Naam**: Stephan Dorst **Functie**: Projectmanager **Bevindingen**: de testpersoon wist direct de juiste filter te vinden en bekeek via de detail button de details van de exception. **Verbeteringen**: De testpersoon heeft het liefst de exception message met een link naar de detailpagina in het overzicht zodat hij direct kan zien om wat voor exception het gaat. **Opmerkingen**: De testpersoon acht het wellicht handig om te noteren wie de exception heeft toegekent aan de werknemer.

Scenario 2: Je wilt kijken of een bepaalde exception op het domein ‘eva-floro-nl.eva.dev.floro.nl al is opgelost.		
De verwachting	De gebruiker navigeert naar het kopje ‘exceptions’ onder de categorie support. Vervolgens filtert de gebruiker het overzicht op de status ‘resolved’.	
De uitvoering	Testpersoon 1	**Naam**: Edwin Vogelaar **Functie**: Frontend developer **Bevindingen**: Na even zoeken had de testpersoon de filter gevonden. Er werd opgemerkt dat de filter wel redelijk klein weergeven is. Het overzicht trekt meer de aandacht dan de filters. **Verbeteringen**: hoewel de filters in overeenstemming zijn met de andere onderdelen van het CRM is het misschien handig om de filters duidelijker te maken door deze allemaal een dropdown optie te geven, i.p.v. teksuele linkjes. **Opmerkingen**: -

	Testpersoon 2	**Naam**: Stephan Dorst **Functie**: Projectmanager **Bevindingen**: De testpersoon maakt geen gebruik van de filters, maar zoekt direct de domein in de zoekbalk. **Verbeteringen**: Zoals eerder vermeld vind de testpersoon het handig om de exception al in het overzicht te kunnen zien. **Opmerkingen**: Zoeken via een zoekbalk i.p.v. met filters is de testpersoon gewent en vind dit een handige manier van filteren.

Scenario 3: Een klant neemt telefonisch contact met je op, en vertelt dat er een exception is opgetreden op hun website. Zoek de exception van de klant op.		
De verwachting	De gebruiker navigeert naar het kopje ‘exceptions’ onder de categorie support. Vervolgens filtert de gebruiker het overzicht op de domein van de klant.	
De uitvoering	Testpersoon 1	**Naam**: Edwin Vogelaar **Functie**: Frontend developer **Bevindingen**: Na al wat ervaring met het proof of concept en de filters te hebben opgedaan, kon de testpersoon de filter gelijk vinden. **Verbeteringen**: De testpersoon merkte op dat een klant ook meerdere domeinen kan hebben. Het is wellicht handig om naast alleen een domein filter ook een klant filter te hebben. Wanneer er een klant gekozen is, kan de domein filter hierop inspelen door alleen de domeinen van de gekozen klant te tonen. **Opmerkingen**: -
	Testpersoon 2	**Naam**: Stephan Dorst **Functie**: Projectmanager **Bevindingen**: De testpersoon zoekt wederom in de zoekbalk naar het betreffende domein. **Verbeteringen**: - **Opmerkingen**: -

Conclusie
Na de GAT is gebleken dat er zeker nog wat verbetering te behalen valt in het overzicht. De exceptions zullen worden getoond in het overzicht en er zal extra aandacht besteed worden aan de naamgeving van de kolommen.

Verder zal de opmerking over de filters uit scenario 2 van testpersoon 1 in acht worden genomen, maar met een lage prioriteit. De filters zoals deze zijn ingebouwd in het proof of concept zijn namelijk wel in overeenstemming met de filters die gebruikt zijn in de andere onderdelen van het CRM.

De verbeterpunten worden meegenomen in toekomstige uitbreidingen van het proof of concept. De verbeterpunten worden nader gespecificeerd in hoofdstuk 15.

14 Iteratie 4: Beheerbaar maken van de exceptions

Om de exceptions te kunnen beheren zijn er attributen toegevoegd aan de exception en komen er nog een aantal functionaliteiten bij zoals notificaties en reminders. De functionaliteit van het beheren wordt gebouwd tijdens deze iteratie.

14.1 Requirements

Er zijn een aantal requirements toegevoegd aan het rapport betreft de beheerbaarheid van exceptions. De use cases behorende bij de user requirements zijn opgenomen in Bijlage E.

User requirements

ID	Beschrijving	BR	UC
U05	De gebruiker wil een status kunnen aangeven voor een exception. De status kan zijn: ‘unresolved’, ‘being resolved’ of ‘resolved’.	B06	C04
U06	De gebruiker wil een prioriteit aan een exception kunnen toekennen. De prioriteit van zijn: ‘high’, ‘medium’ of ‘low’.	B06	C05
U07	De gebruiker wil een werknemer aan een exception kunnen toekennen.	B06	C06

Functionele requirements

ID	Beschrijving	BR	UC
F14	De module maakt het mogelijk om een status aan een exception te kunnen toewijzen.	B06	C04
F15	De module maakt het mogelijk om een prioriteit aan een exception te kunnen toewijzen.	B06	C05
F16	De module maakt het mogelijk om een werknemer aan een exception te kunnen toewijzen.	B06	C06
F17	De module maakt het mogelijk een herrinneringsmail te kunnen versturen naar de developer die verantwoordelijk is voor het oplossen van een exception.	B06	C07

14.2 Het beheren van de exception

Vanuit het overzicht kan de gebruiker klikken op het attribuut dat ze willen beheren. Via een link wordt de gebruiker vervolgens naar een pagina gestuurd waarop zij het attribuut kunnen wijzigen.

Het aanmaken van de pagina’s volgt dezelfde methode als beschreven in het vorige hoofdstuk. De pagina’s moet ik net zoals alle andere pagina’s aanmaken in de database van CusWeb. De pagina’s die ik voor het beheren heb aangemaakt zijn: ‘changeExceptionEmployee’, ‘changeExceptionPriority’ en ‘changeExceptionStatus’. Wederom heb ik voor de pagina’s ook de bestanden aangemaakt in de code die de HTML van de pagina renderen. Deze pagina’s maken gebruik van de get en set functies uit het exception object die de gewijzigde data opslaan.

14.3 Beheerbare attributen

Door de module te bekijken vanuit het perspectief van de gebruiker heb ik geconstateerd dat er een prioriteit, status en werknemer aan een exception toegekent moet kunnen worden. Deze attributen houden ook sterk verband met de keuze voor de filters in het overzicht. Omdat er filters zijn vastgelegt voor prioriteit, status en werknemer moeten deze logischerwijs ook kunnen worden aangegeven.

Prioriteit

Ik voeg functionaliteit toe om de exceptions een prioriteit te kunnen toekennen. Deze prioriteit kan zijn: ‘high’, ‘medium’ of ‘low’. Door een prioriteit toe te kennen aan een exception kan er worden aangegeven hoe belangrijk het is dat deze exception moet worden opgelost. Zoals aangegeven in het vorige hoofdstuk zorgt het aangeven van een prioriteit er ook voor dat er in het overzicht gefilterd kan worden op prioriteit wat handig is als je een groot aantal exceptions hebt, maar alleen de belangrijkste wil tonen.

Status

Verder maak ik het mogelijk om een status te kunnen toekennen aan een exception. De status kan zijn: ‘unresolved’, ‘being solved’ of ‘resolved’. De status geeft aan of er aan de exception gewerkt wordt. Dit is handig om de voortgang van het oplossen van de exception te kunnen bijhouden. Ook dient het, net als prioriteit, als een handige filter in het overzicht.

Werknemer

Wanneer er exceptions zijn binnengekomen is het voor alsnog de vraag wie de exception zal oplossen, daarom maak ik het mogelijk om een werknemer te kunnen toekennen aan de exceptions. De werknemer die de laatste wijziging aan het project heeft doorgevoerd bekend is door het ophalen van de gegevens van de commit. Het kan echter zijn dat deze werknemer op het moment van het optreden van de exception niet aanwezig is op kantoor om dit op te lossen. Om deze situatie voor te zijn, heb ik er in overleg met de opdrachtgever voor gekozen om handmatig een werknemer toe te kennen. Het beheren van werknemers staat het toe om het overzicht efficiënt te kunnen filteren. Daarnaast is er met het toewijzen van een werknemer, een verantwoordelijke en aanspreekpunt voor de exception vastgelegt.

14.3.1 Notificaties

Verder heb ik bedacht dat het handig kan zijn om mail notificaties en reminders te integreren in de module. N.a.v. een gesprek met de opdrachtgever kwamen we tot de vraag: ‘hoe gaan we ervoor zorgen dat de exceptions die zijn toegekent aan de werknemers ook daadwerkelijk opgelost gaan worden?’. Het is belangrijk om hier overzicht op te houden en de werkzaamheden hierin te kunnen beheren.

Mail notificatie

Ik heb gekozen voor het mailen van een werknemer wanneer deze toegekent is aan een exception. De werknemer wordt hiermee op de hoogte gebracht van zijn taak en kan hiermee aan de slag.

Reminder

Wanneer een werknemer zijn taak niet op tijd uitvoerd of mogelijk vergeet, moet hij of zij eraan herinnert worden dat er vaart moet zitten achter het oplossen van de exception. Met betrekking tot deze situatie heb ik ervoor gezorgt dat er een reminder kan worden verstuurd.

De reminder kan handmatig worden verstuurd, maar kan ook door middel van het toekennen van een deadline aan een prioriteit geautomatiseerd plaatsvinden. Wanneer bijvoorbeeld de exception een prioriteit ‘high’ heeft, wordt er een reminder verstuurd

wanneer de exception niet binnen een dag opgelost is. Wanneer een prioriteit lager is zal de deadline wat langer zijn. Dit is echter een functionaliteit die in een latere versie zal worden ingebouwd wegens het gebrek aan tijd en de niet noodzakelijke aard van de functie.

14.4 Testen

Het is belangrijk dat de exceptions zonder problemen beheerd kunnen worden. Ik heb scenario's opgesteld die uitgevoerd zullen worden door een aantal toekomstige gebruikers van de module. Voor het testen van deze iteratie wordt er een gebruikersacceptatietest (GAT) op de ontwikkelomgeving uitgevoerd.

Testbasis
Tijdens deze test valt alleen het overzicht in het CRM binnen het beschouwingsgebied.

Teststrategie
Het is de bedoeling dat uiteindelijk de gebruikers de exceptions kunnen beheren. Daarom wil ik er met behulp van de GAT achterkomen of het beheer voldoet aan de verwachtingen en het gebruik efficiënt is en voorzich spreekt.

Testdata
De data uit het overzicht is afkomstig van door mij zelf gegenereerde exceptions. Deze exceptions zijn getriggert op mijn development container in de checkout van het CMS en mijn eigen Floro website.

Uitvoering
Tijdens deze test heb ik een tweetal werknemers het overzicht laten gebruiken. Er is gekozen voor een developer en een projectmanager om twee verschillende kanten van het gebruik van het proof of concept te achterhalen. Er is gekozen voor twee personen omdat deze personen de twee groepen representeren die met de module zullen omgaan, namelijk managers en developers.

Voor de test moeten zij scenario's uitvoeren die in het dagelijksleven zouden kunnen voorkomen. De focus hierbij is: 'is de juiste data beschikbaar en zijn deze gemakkelijk te beheren?'.

Ik heb de testpersonen ingeleid in het proof of concept. Vervolgens heb ik ze een aantal scenario's laten uitvoeren. De scenario's en de bevindingen zijn onderstaand genoteerd.

Scenario 1: Er is gevraagt om de exceptions op het domein eva-floro-nl.eva.dev.floro.nl te controleren en deze een prioriteit en werknemer toe te kennen.		
De verwachting	De gebruiker overziet het overzicht van de exceptions en kiest d.m.v. de beheerbare links in de kolommen een prioriteit en werknemer.	
De uitvoering	Gebruiker 1	**Naam**: Edwin Vogelaar **Functie**: Frontend developer **Bevindingen**: Het kunnen aangeven van een prioriteit en werknemer (en status) is duidelijk weergeven. Het spreekt voor zich hoe je een dergelijk attribuut kan toekennen aan de exception. **Verbeteringen**: Mogelijk is het een verbetering om in het overzicht gelijk een dropdown te weergeven met de mogelijke opties i.p.v. naar een andere pagina

		verwezen worden waar je het attribuut pas kan veranderen. **Opmerkingen**: Hoewel een dropdown voor de beheerbare attributen in het overzicht wellicht handig is, is de huidige oplossing met linkjes wel in overeenstemming met de andere onderdelen in het CRM.
	Gebruiker 2	**Naam**: Stephan Dorst **Functie**: Projectmanager **Bevindingen**: De testpersoon zoekt via de zoekbalk naar het domein. De beheerbare attributen zijn door de testpersoon direct te vinden en aan te passen. **Verbeteringen**: De testpersoon vind de extra handeling van de doorverwijzing naar de extra pagina onnodig. Hij ziet liever een dropdown met keuzes direct in het overzicht. Wellicht zelfs direct een knop 'solved'. **Opmerkingen**: -

Scenario 2: Je hebt zojuist een exception opgelost. Geef dit aan.		
De verwachting	De gebruiker kent de status 'resolved' toe aan een exception.	
De uitvoering	Gebruiker 1	**Naam**: Edwin Vogelaar **Functie**: Frontend developer **Bevindingen**: Het beheren ging foutloos en was duidelijk. **Verbeteringen**: Mogelijk kan er een melding worden gegeven wanneer een exception is opgelost. Bijvoorbeeld naar de klant of toekenner van de exception. **Opmerkingen**:
	Gebruiker 2	**Naam**: Stephan Dorst **Functie**: Projectmanager **Bevindingen**: De testpersoon wist het beheerbare attribuut 'status' direct te vinden. Er zijn hier geen problemen bij ondervonden. **Verbeteringen**: Ook hier is het wellicht handiger om direct een dropdown in het overzicht te zien om de status te kunnen kiezen. **Opmerkingen**: -

Scenario 3: Je hebt een exception toegekent aan een werknemer. De werknemer is nog niet begonnen met het oplossen van de exception. Herrinner de werknemer eraan dat de exception opgelost moet worden.		
De verwachting	De gebruiker klikt op de knop 'send reminder' van de toegekende exception.	
De uitvoering	Gebruiker 1	**Naam**: Edwin Vogelaar **Functie**: Frontend developer **Bevindingen**: De testpersoon kan de button in een

		keer vinden. **Verbeteringen**: - **Opmerkingen**: -
	Gebruiker 2	**Naam**: Stephan Dorst **Functie**: Projectmanager **Bevindingen**: De testpersoon weet direct de reminder button te vinden. Er zijn hierbij geen problemen ondervonden. **Verbeteringen**: De testpersoon geeft aan het handig te vinden om voor het sturen van reminders een log bij te houden. Deze log houdt bij hoevaak is er een reminder is gestuurd. Op deze manier kunnen de toegekende werknemers er op worden aangesproken wanneer zij niet reageren op de reminders. **Opmerkingen**: -

Conclusie
Uit de test is gebleken dat de testpersonen liever een attribuut gemakkelijker kunnen beheren met een dropdown selector in het overzicht. Hierdoor hoeven ze niet naar een aparte pagina om het attribuut aan te passen. Dit is zeker een punt wat meegenomen zal worden in een toekomstige uitbreiding van het proof of concept.

De opmerking over het geven van een melding zodra een exception is opgelost zal zeker worden meegenomen om eventueel op te nemen in een volgende versie van het proof of concept.

Daarnaast zal het bijhouden van een log ook in acht genomen worden bij een volgende versie van het proof of concept.

Er zijn goede verbeterpunten achterhaald die zeker kunnen bijdragen aan de volgende versie van het proof of concept. Deze worden nader besproken in het volgende hoofdstuk.

15 Afronding

Alle iteraties zijn doorlopen. Het proof of concept is gebouwd met alle noodzakelijke functionaliteit. Er is besproken met de opdrachtgever dat het proof of concept nog niet in productie zal worden genomen, omdat dit meer tijd vergt dan er nog over is tijdens deze afstudeerperiode.

15.1 Toekomstige uitbreidingen

Naar aanleiding van de uitgevoerde gebruikersacceptatietesten zijn er een aantal uitbreidingen voor het proof of concept die de module zal verbeteren. Deze uitbreidingen zullen in een later stadium in het proof of concept worden ingebouwd. De mogelijke uitbreidingen zijn onderstaand genoteerd.

ID	Beschrijving
T01	De module stelt de gebruiker in staat om een deadline aan een prioriteit toe te kennen. Wanneer de deadline verstreken wordt terwijl de status nog 'being resolved' is, zal er een reminder verstuurd worden naar de toegekende werknemer wat betreft het oplossen van de exception.
T02	De tracering van de module zal commits van het Git project behorende bij de website waarop de exception is opgetreden ophalen op basis van de file waarin de exception is opgetreden.
T03	Het overzicht in de module zal een filter hebben waarmee gefilterd kan worden op klant.
T04	Het overzicht in de module zal de filters op elkaar afstellen.
T05	Het overzicht in de module zal attributen direct beheerbaar maken vanuit het overzicht door het tonen van een dropdown.
T06	De module zal een melding sturen naar de klant wanneer een exception is opgelost.
T07	De module zal een log bijhouden van de verstuurde reminders.

Prioriteit

ID	Must have	Should have	Could have	Won't have
T01				
T02				
T03				
T04				
T05				
T06				
T07				

16 Evaluatie

In dit hoofdstuk zal ik terugkijken op de afstudeerperiode. Er zal geëvalueerd worden op het product, het proces en de beroepstaken.

16.1 Productevaluatie

Ik ben tevreden over het opgeleverde product. Het opgeleverde proof of concept heeft de doelstellingen behaald en de problemen in de beginsituatie opgelost. Het proof of concept heeft het overzicht van de supporttickets doen toenemen doordat het de exceptions van de supporttickets scheidt.

Verder kunnen de opgetreden exceptions met het proof of concept overzien en beheert worden. Het proof of concept heeft de exception handeling op de websites geoptimaliseerd en grotendeels geautomatiseerd.

De meest belangrijke en cruciale elementen zijn binnen de tijd verwerkt in het proof of concept. Er zijn alleen nog wat optionele uitbreidingen die in de toekomst gerealiseerd zullen worden.

16.2 Procesevaluatie

Het proces van de uitvoering van de opdracht ging niet zonder slag of stoten. Het was voor mij even een omschakeling toen de opdracht werd herzien en ik een andere mindset moest aannemen voor de opdracht.

Uiteindelijk is de omschakeling wel gelukt en heb ik veel geleerd van het proces. De wijzigende requirements en opdrachtgevers met andere belangen waren erg leerzaam om mee te maken.

Verder was het een klus om het proces duidelijk te documenteren in het afstudeerverslag. Na een aantal feedback momenten op school heb ik dit goed goed mogelijk opgelost door zo veel mogelijk een chronologische volgorde aan te houden om het zo begrijpelijk mogelijk te houden.

Soms had ik wat moeite met het terugkoppelen van resultaten naar de opdrachtgever. Ik kreeg hier en daar wat gemixte feedback over het bespreken over de voortgang. Uiteindelijk is dit besproken en duidelijk gemaakt met mijn begeleider en hebben we nog een goed feedback gesprek kunnen houden.

16.3 Evaluatie beroepstaken

Gedurende de doorlooptijd is er tijdens de herziening een beroepstaak komen te vervallen en is daar een nieuwe beroepstaak voor in de plaats gekomen.

Beroepstaak 3.2 zou oorspronkelijk op niveau 4 uitgevoerd worden. Deze is uiteindelijk op niveau 3 uitgevoerd en requirement 1.4 op niveau 4. De wijzigingen worden onderstaand nader toegelicht.

1.3 Selecteren van standaardsoftware (niveau 3)
Deze beroepstaak is door de herziening van de opdracht niet op niveau uitgevoerd. Er is voor gekozen om deze beroepstaak te laten vervallen en beroepstaak 1.4 hiervoor in de plaats te nemen. Hoewel er een goed begin is gemaakt aan de toolselectie is deze niet afgerond.

Over het uitgevoerde gedeelte ben ik wel tevreden. Tijdens het proces heb ik ervaring opgedaan met het researchen naar de tools. Vaak ondervond ik dat de informatie over de tools op de websites wel wat gebrekkig was en dat veel tools waren verouderd. Het onderzoek en het selecteren van de tools bleek lastiger te zijn dan ik had ingeschat.

1.4 Uitvoeren analyse door definitie van requirements (niveau 4)
Deze beroepstaak is er na de herziening bij gekomen. Er zijn veel requirements opgesteld en ook is er omgegaan met veel wijzigende en botsende requirements. Verder moest er gewerkt worden met twee versies van het requirementsrapport om zo de requirements voor en na de herziening van de opdracht te scheiden. Aangezien er tijdens het proces zoveel nadruk is komen te liggen op de requirements heb ik ervoor gekozen om deze beroepstaak op niveau 4 uit te voeren in plaats van beroepstaak 3.2.

Het beheren van de requirements is naar mijn mening goed verlopen. Door het gebruik van prioritering en traceerbaarheid waren de requirements goed te overzien. Het scheiden van requirements van de oude en nieuwe situatie is goed geweest om het overzicht te behouden.

3.2 Ontwerpen systeemdeel (niveau 3)
Gedurende het proces zijn er meerdere ontwerpen gemaakt. Deze ontwerpen zijn gemaakt met UML. In het afstudeerplan is vastgelegd dat deze beroepstaak op niveau 4 uitgevoerd zou worden. Ik heb er echter voor gekozen om deze op niveau 3 uit te voeren, omdat het in mijn situatie beter uitkwam om de requirements op niveau 4 uit te voeren en het ontwerpen op niveau 3 te laten.

Ik heb zoveel mogelijk de juiste modelleringstechnieken toegepast die passen bij bepaalde situaties. Ik heb een use case diagram, sequencediagrammen, een componentdiagram, objectdiagrammen, klassendiagrammen en procesdiagrammen opgesteld.

De ontwerpen hielpen me om de context van de vast te leggen en om funtionaliteiten uit te werken. Het heeft veel inzicht gebracht tijdens het proces om, waar nodig, bepaalde aspecten visueel vast te leggen.

3.3 Bouwen applicatie (niveau 3)
Tijdens het bouwen heb ik gewerkt met meerdere systemen, namelijk het CRM, CMS en het intern ontwikkelde framework. Daarnaast heb ik gebruik gemaakt van een RVM ontwikkelomgeving en versiebeheer met Git.

Ik heb veel gebruik gemaakt van API calls en heb daarbij veel te maken gekregen met het omgaan met data, zoals het versturen, ophalen en samenvoegen van allerlei verschillende data met betrekking tot de exceptions.

Naar mijn mening is het bouwen redelijk soepel verlopen. Het werken in de verschillende systemen brachten allemaal eigen manieren met zich mee. Maar door gebruik te maken van de documentatie en door te kijken naar bestaande elementen in de systemen kon ik goed m’n weg vinden.

3.5 Uitvoeren van en rapporteren over het testproces (niveau 3)
Uiteraard is er gedurende het proces getest. Bouwen en testen gaan hand in hand. Gedurende het testproces heb ik geprobeerd zo goed mogelijk de juiste tests voor de juiste situaties te kiezen. Ik heb een vorm van een ketentest uitgevoerd, moduletests en gebruikersacceptatietests.

Literatuurlijst

Grood, D.-J. d. (2008). TestGoal. Den Haag: Sdu Uitgevers bv.
Jim Arlow, I. N. (2011). UML 2 and the unified process. Pearson Education Inc.
Swart, N. d. (2010). Handboek requirements. Eburon Business.

Verklarende woordenlijst

In dit hoofdstuk zijn alle begrippen opgenomen die een toelichting nodig kunnen hebben.

Begrip	Definitie/verklaring
Application Programming Interface (API)	Een API zorgt ervoor dat een programma kan communiceren met een ander programma of onderdeel (meestal bibliotheken).
Backend developer	Een backend developer is een ontwikkelaar die zich bezighoudt met de achterkant van websites.
Behaviour Driven Development (BDD)	BDD is een software ontwikkel proces dat ontstaan is vanuit een basis van testgedreven ontwikkelen en object georienteerde analyse en design.
Bugs	Een bug is een foutje in een programma of website.
Checkout	Checkout is een term uit versiebeheer en is een complete werkende kopie van een repository.
Cloudoplossingen	Een cloudoplossing is een mogelijkheid om via een netwerk, software en gegevens beschikbaar te stellen.
Commit	Een commit is een term uit versiebeheer en zorgt ervoor dat gemaakte wijzigingen worden gepusht naar de repository.
Composer	Een composer is een package manager voor PHP.
CURL	CURL is een functie van PHP die je in staat stelt om verbinding te maken en te communiceren met verschillende servers met verschillende protocollen.
Customer Relationship Management (CRM)	Een CRM is een systeem, werkwijze en technologie voor de klantenbeheer van een bedrijf.
CusWeb	CusWeb is het CRM van Floro Webdevelopment B.V.
Exception	Een exception is een fout.
Exception handeling	Exception handeling is de afhandeling van exceptions.
Exception tracering	Excepion tracering is de tracering van de exceptions naar de oorzaak.
Features	Een feature is een attribuut of aspect.
Framework	Een framework is een abstract herbruikbaar geraamte met generieke functionaliteit die geïmplementeerd kan worden in meerdere websites.
Frontend developer	Een frontend developer houdt zich bezig met de voorkant van websites. Dit omvat het gedeelte van de website dat je ziet als bezoeker.
Gebruikersacceptatietest (GAT)	Een GAT is een test om te kijken of de module de gebruiker ondersteund.

Generalisatie	Het algemeen maken van bepaalde dingen.
Git	Git is een gratis open source versiebeheer systeem voor zowel grote als kleine projecten.
Git clone	Git clone is een kopie van een Git repository
Gitlab	Gitlab is een Git versiebeheertool.
Graphical User Interface (GUI)	Een GUI is een interface waarin gebruikers kunnen interacteren met systemen.
Headless testing	Headless testing is het testen van een webbrowser zonder GUI.
Interactief vormgever	Een interactief vormgever ontwerpt apps en websites.
Keep It Stupid Simple (KISS)	KISS is een principe dat systemen soms het beste werken als ze simpel zijn gehouden.
Model View Controller (MVC)	MVC is een architectuur voor software wat de implementatie het het design scheidt.
Modularisatie	Het scheiden van eenheden met een zelfde functie of doel.
MoSCoW	MoSCoW is een prioriteringstechniek wat staat voor: must have, should have, could have en wouldn't have.
Navigation scripting	Een navigation script navigeert over een website.
Object georienteerd	Object georienteerd is een benadering voor het bouwen van systemen aan de hand van objecten.
OTAP	OTAP staat voor Ontwikkelen, Testen, Acceptatie en Productie. Een OTAP is een methodiek voor software ontwikkeling.
Package manager	Een package manager is een beheersysteem voor het beheren van software tools
PHP	PHP is een server-side programmeertaal. PHP is een veelgebruikte taal in webdevelopment.
Private token	Een private token is een bewijs van authenticatie
Protocol	Een protocol is een manier van communiceren tussen twee netwerkcomponenten.
Pull	Pull is een term uit versiebeheer en haalt alle wijzigingen van anderen binnen en verwerkt deze in jou eigen lokale checkout.
Remove Virtual Machine (RVM)	Een RVM is software dat een fysieke computer emuleert (met hardware) waar een besturingssysteem op kan draaien.
Repository	Een repository is een term uit versiebeheer en is de plek waarin de versies van een project worden opgeslagen.
Requirements	Requirements zijn wensen en eisen.

Screen capturing	Screen capturing is het opnemen van scherm activiteit.
SMART	SMART is een formuleringsmethode voor doelstellingen en staat voor: specifiek, meetbaar, acceptabel, realistisch en tijdsgebonden.
SourceTree	SourceTree is een programma voor versiebeheer via Git.
Testing framework	Een testingframework is een framework voor het schrijven van testscripts
Testscript	Een testscript is een set van instructies die de module zal uitvoeren om te kijken of de module naar behoren functioneert.
Ticket	Een ticket is een support vraag.
Tools	Een tool is een klein programma voor het uitvoeren van bepaalde taken.