Afstudeerverslag

Ontwikkeling van een generiek formulierenregistratiesysteem bij FourICT B.V.

Opleiding: Informatica deeltijd

Begeleidend examinator: dhr. E.M. van Doorn

Tweede examinator: mw. A.M.J.J Lousberg

Datum: 24-03-2016

Plaats: Zoetermeer

Versie: 1.0

Geschreven door: Mark Barto (12092452)

Versiebeheer

Versie	Datum	Omschrijving
0.1	29-11-2015	Initiële versie.
0.2	05-02-2016	Verbeter versie naar aanleiding van bedrijfsbezoek.
0.3	15-02-2016	Verbeter versie naar aanleiding van bespreking concept afstudeerdossier.
0.4	03-03-2016	Verbeter versie naar aanleiding van bespreking assessment afstudeerdossier.
0.5	12-03-2016	Verbeter versie naar aanleiding van bespreking met bedrijfsmentor.
1.0	24-03-2016	Laatste aanpassingen doorgevoerd.

Referaat

Dit afstudeerverslag beschrijft het proces dat Mark A. Barto heeft doorlopen, bij het ontwikkelen van een generiek formulierenregistratiesysteem, in de periode van november 2015 t/m maart 2016 bij FourICT B.V. te Zoetermeer.

Descriptoren:

- Generiek
- Formulier
- UML
- C#
- Web API
- AngularJS
- Scrum
- Single Page Application
- SPA
- Architectuur
- App
- Webapplicatie
- Identity Hub
- Ionic
- nServiceBus
- .NET
- BreezeJS

Voorwoord

Dit afstudeerverslag is onderdeel van het afstudeerproject van de opleiding Deeltijd Informatica aan de Haagse Hogeschool dat door Mark A. Barto is uitgevoerd.

Hierbij gaat mijn dank uit naar:

Dhr. R. van Leeuwe (Managing Partner, FourICT), voor zijn begeleiding als bedrijfsmentor om hoofdzaken van bijzaken te scheiden en kort en krachtig te beargumenteren. Daarnaast voor het beschikbaar stellen van deze opdracht.

Dhr. H. van Ewijk (Managing Partner, FourICT), voor het beschikbaar stellen van deze opdracht.

Dhr. E.M. van Doorn (Examenbegeleider, HHS), voor zijn feedback gedurende het afstuderen op mijn afstudeerdossier.

Mw. A.M.J.J Lousberg (Examenbegeleider, HHS), voor haar feedback gedurende het afstuderen op mijn afstudeerdossier.

Dhr. A. Versteeg (Eigenaar, Versteeg IT Solutions), voor zijn tijd en zijn feedback.

Inhoudsopgave

1. Inleiding

Dit afstudeerverslag is opgesteld door Mark A. Barto naar aanleiding van het afstuderen van de opleiding Deeltijd Informatica aan De Haagse Hogeschool. De opdracht is uitgevoerd bij FourICT B.V. in de periode van 16 November 2015 t/m 29 Maart 2016.

Het doel van de opdracht is het ontwikkelen van een generiek systeem waarbij klanten digitale formulieren kunnen ontwerpen en beheren en waarbij medewerkers in het veld de formulieren kunnen invullen.

Hoofdstuk 2 gaat in op de beschrijving van de organisatie. Hier wordt ingegaan op de omschrijving van het bedrijf, de probleemstelling, de doelstelling en het uiteindelijke verwachte resultaat van de opdracht. In hoofdstuk 3 wordt het plan van aanpak besproken. Hier wordt duidelijk hoe er te werk is gegaan en op welke manier. In hoofdstuk 4 t/m 11 worden de werkzaamheden besproken die zijn uitgevoerd tijdens de sprints met toelichting en motivatie van de gemaakte keuzes. Ten slotte wordt in hoofdstuk 12 ingegaan op de evaluatie van het proces en de opgeleverde producten.

2. Beschrijving van de organisatie

2.1. Bedrijf

FourICT B.V. is een ICT-bedrijf dat in 2008 is opgericht en zich heeft gespecialiseerd in het ontwikkelen en integreren van complexe ICT-toepassingen in de Telecom, Media en Healthcare sector. FourICT bestrijkt het gehele traject van analyse, ontwerp, ontwikkeling, test en implementatie tot en met projectmanagement en beheer.

Het bedrijf bestaat uit een team van 10 vaste medewerkers en 3 zelfstandige ondernemers welke ondersteund worden door een project secretariaat. FourICT richt zich in het bijzonder op proces automatisering en softwareontwikkeling op basis van Microsoft technologie.

2.2. Probleemstelling

In het verleden is bij FourICT een maatwerk systeem ontwikkeld voor de klant waarbij formulieren beheerd kunnen worden. Aangezien het huidige systeem gebaseerd is op achterhaalde technologie is het steeds moeilijker om het systeem te onderhouden. Daarnaast voldoet het huidige systeem niet meer aan de eisen van klant. Hierbij moet gedacht worden aan het offline kunnen werken zonder internetverbinding en de integratie met bedrijfssystemen. Verder voldoet het huidige systeem niet aan de gebruikte standaarden waarbij nagedacht is over de verschillende kwaliteitseisen als schaalbaarheid en koppelbaarheid. Omdat het uitbreiden van het huidige systeem een kostbare gelegenheid wordt is er gekozen om een nieuw systeem te ontwikkelen. Aangezien ook andere klanten bij FourICT interesse hebben in het systeem is uiteindelijk de vraag ontstaan om een generiek systeem te ontwikkelen gebaseerd op nieuwe technologie.

2.3. Doelstelling

Het ontwikkelen van een generiek systeem waarbij klanten digitale formulieren kunnen ontwerpen en beheren. Zonder enige IT-kennis. En waarbij medewerkers in het veld de formulieren kunnen invullen.

2.4. Resultaat

Een systeem waarbij de klant een digitaal formulier kan ontwerpen ter vervanging van een papieren formulier. Bij het ontwerpen van het formulier ligt de basis op simpel en eenvoudig. Er wordt nog geen gebruik gemaakt van drag-en-drop functionaliteit, het sorteren en positioneren van velden. De nadruk ligt nu op het eenvoudig selecteren van basis velden zoals een tekstveld, checkbox, selectbox, en tekstarea om te ontwerpen. Daarnaast het eenvoudig controleren van velden op inhoud zonder afhankelijkheid op andere velden. Daarentegen moet het systeem wel onderhoudsvriendelijk zijn waarbij rekening moet worden gehouden met nieuwe type velden zoals het gebruik van specifieke functionaliteiten van het mobiele apparaat als foto’s maken, barcode scannen en handtekening plaatsen. Het systeem moet daarnaast offline beschikbaar zijn en op verschillende mobiele apparaten. De kracht van het systeem zal een gebruiksvriendelijk systeem zijn dat goed te onderhouden is. Tevens heeft de applicatie een grote flexibiliteit m.b.t. het integreren van data van het formulier via e-mail of tot een maatwerk interface met het backoffice systeem van de klant.

3. Plan van aanpak

In de eerste twee weken is een plan van aanpak geschreven om helder een beeld te krijgen van het probleem en het gewenste resultaat. Er is eerst een inventarisatie gedaan naar de gebruikte methoden en technieken binnen het bedrijf. Daarna is de scope bepaald om goed vooraf een afbakening van de opdracht te krijgen en risico's en onzekerheden in kaart te brengen. Vervolgens is er gekeken naar de borging van de kwaliteit, de opgeleverde producten en is er een globale planning afgegeven. Dit plan van aanpak is besproken met de opdrachtgever en is goedgekeurd.

3.1. Projectmethodiek

Om te bepalen wat de juiste keuze is als projectaanpak is er gekeken naar de traditionele waterval aanpak en de Agile aanpak. Binnen FourICT wordt er gewerkt volgens een Agile aanpak. Er wordt geen gebruik gemaakt van een watervalmethode omdat vaak de requirements vooraf nog niet duidelijk zijn. Hierdoor kan in deze methode niet ingespeeld worden op veranderende requirements. Er is daarom gekozen voor een Agile aanpak om snel in te spelen op nieuwe requirements. Een Agile aanpak is gebaseerd op het Agile Manifesto[1]:

- Mensen en hun onderlinge interactie boven processen en hulpmiddelen;
- Werkende software boven allesomvattende documentatie;
- Samenwerking met de klant boven contractonderhandelingen;
- Inspelen op verandering boven het volgen van een plan.

Er zijn verschillende methodieken gebaseerd op een Agile aanpak zoals XP (eXtreme Programming) en Scrum. Bij XP ligt de nadruk op samenwerken. Dit betekent dat de opdrachtgever constant betrokken moet zijn bij het ontwikkelproces. In tegenstelling tot Scrum heeft de opdrachtgever meer een adviserende rol. Ik heb getwijfeld tussen het gebruik van XP en Scrum. Beide methodes kunnen goed gebruikt worden als methodiek. Ik heb uiteindelijk gekozen voor de Scrum aanpak. Ten eerste omdat de opdrachtgever heeft aangegeven dat hij niet continu betrokken wil zijn bij het ontwikkelproces. Ten tweede omdat ik hier ervaring in heb.

Scrum is daarnaast een goede keuze in verband met de structuur binnen het bedrijf en de korte communicatielijnen. Hierdoor kunnen beslissingen snel worden genomen. Daarnaast het risico dat de opdrachtgever bij oplevering niet tevreden is over het eindproduct is een stuk kleiner. Dit komt doordat er snel terugkoppeling is van een werkend product naar de opdrachtgever toe. Hierdoor kan er snel ingesprongen worden op wijzigingen. De opdrachtgever voelt zich hierdoor meer betrokken bij het opgeleverde product en dit geeft vertrouwen. Aangezien verder niet alle eisen en wensen vooraf bekend zijn heeft mij ertoe besloten om voor deze aanpak te gaan.

Aangezien Scrum door mij geheel zelfstandig wordt uitgevoerd zijn de volgende afspraken gemaakt:

- Er is voor gekozen om de Daily Scrum te laten vervallen aangezien het team bestaat uit twee teamleden namelijk mij en de opdrachtgever. Verder door de platte structuur is er direct communicatie mogelijk tussen mij en de opdrachtgever.

[1] http://www.agilemanifesto.org/iso/nl/

- Een Sprint heeft een duur van 2 tot 4 weken. Waarbij 2 de minimale lengte is van de duur van een Sprint en 4 het maximale. Er is gekozen voor een duur van 2 weken. De keuze is gemaakt voor een korte Sprint om het risico te beperken dat er werk overnieuw gedaan moet worden. Aangezien een demonstratie aan het einde van een Sprint wordt gegeven kan de opdrachtgever snel feedback geven. Als dit later wordt aangegeven kan dit ervoor zorgen dat er meer tijd verloren gaat aan het herstellen.
- Er is een voorbereidingsfase ingeplant als sprint 0 om zodoende snel van start te kunnen gaan voordat de daadwerkelijke eerste sprint begint.
- Van user stories die geselecteerd zijn voor een sprint wordt aan het begin van de sprint ook een use case gemaakt. Dit aangezien een use case meer details bevat dan een user story.
- Tijdens een sprint is afgesproken dat er unit tests worden uitgevoerd. Dit is belangrijk om de kwaliteit te waarborgen van de software.
- De Scrummaster rol wordt door mij uitgevoerd. Dit aangezien het traject zelfstandig moet worden uitgevoerd.
- De retrospective (evaluatie) wordt verwerkt in het afstudeerverslag.

3.2. Globale requirements

Er is met de opdrachtgever een interview gehouden om voorafgaand aan het project te bepalen welke globale eisen aan het systeem worden gesteld.

Functionele requirements

De volgende functionaliteiten moeten gerealiseerd worden voor het systeem:

- De klant moet zich kunnen aanmelden voor het gebruik van het systeem, zodat er alleen toegang is tot zijn eigen ontworpen formulieren en niet van anderen.
- De klant kan zijn eigen digitale formulier ontwerpen, invoervelden definiëren en restricties opgeven voor het valideren van invoervelden.
- De mogelijkheid dat de klant binnen het systeem gebruikers toegang kan geven tot het digitale formulier.
- De klant moet kunnen bepalen op welke manier het digitale formulier wordt afgehandeld. Standaard via de mail.
- De medewerker van de klant kan zich aanmelden om het digitale formulier te bekijken.
- Het formulier moet offline beschikbaar zijn in de app zodat als er geen verbinding tot stand kan worden gebracht zodat de gegevens lokaal worden opgeslagen en niet verloren gaan.
- Bij het versturen van het formulier moet er een validatie plaats vinden van de invoervelden.

Technische beperking

De volgende beperkingen zijn opgelegd door het ICT-beleid van de opdrachtgever.

- Het systeem moet gebaseerd zijn op ASP.NET.
- Het systeem moet gebruik maken van een SQL Database.

Niet-functionele requirements
Naast het bepalen van de functionele requirements van het systeem zijn er tijdens de interview ook niet-functionele requirements naar voren gekomen. De niet-functionele requirements zijn de kwaliteitseisen (gebaseerd op ISO 9126[2]) van het systeem.

Onderstaand een opsomming van de niet-functionele eisen:

- **Gebruiksvriendelijkheid**: Het systeem dient eenvoudig in gebruik te zijn.
- **Beveiligbaarheid**: Het systeem dient een goede rol-toegang model te worden geïmplementeerd.
- **Portabiliteit**: Een generiek formulierensysteem dat platformonafhankelijk is, beschikbaar is op mobiele apparaten en toegankelijk is vanaf iedere computer.
- **Efficiëntie**: De applicatie moet binnen < 1 seconden reageren.
- **Koppelbaarheid**: Het systeem moet makkelijk kunnen integreren met verschillende backoffice systemen van de klant.

3.3. Afbakening van de opdracht
Er wordt gewerkt volgens een Scrum aanpak. Binnen Scrum wordt gewerkt volgens Sprints. Een Sprint is een time-box waar binnen een werkend stukje functionaliteit(en) opgeleverd van het uiteindelijke eindproduct. De opdrachtgever beslist binnen iedere Sprint de volgorde welke functionaliteit(en) moeten worden opgepakt. Het is daarom niet duidelijk welke functionaliteit(en) er zijn opgeleverd aan het einde van opdracht. Aan het einde van de opdracht heeft de opdrachtgever een werkend systeem. Echter de beschikbare tijd bepaald de functionele scope van het systeem. Het kan dus zo zijn dat niet alle requirements zijn ingewilligd.

3.4. Risicofactoren
Hieronder een opsomming van de belangrijkste risico's:

- De opdracht is niet goed geformuleerd of wordt verkeerd geïnterpreteerd. Dit veroorzaakt een slechte start van het project, waardoor een hoop werk overnieuw gedaan moet worden. Om dit te voorkomen worden er (tussentijdse) opleveringen van de applicatie gedaan om dit te ondervangen.

- Gebruik van nieuwe technieken. Dit kan vertragend werken. Inleren heeft effect op de doorlooptijd.

- Incidenten die voortkomen uit de bestaande Software Service Level Agreements van FourICT hebben een hogere prioriteit dan projectactiviteiten. Deze incidenten moeten namelijk binnen een afgesproken tijd worden opgelost en project werkzaamheden worden dan tijdelijk stil gelegd.

- Door inzet op een ander belangrijk project waardoor er minder tijd is om het project af te ronden. Door goede afspraken te maken met mijn opdrachtgever en een globale planning vooraf moet er tijd gereserveerd worden om mijn project af te ronden.

[2] https://nl.wikipedia.org/wiki/ISO_9126

3.5. Kwaliteitsborging

Hieronder een opsomming om de kwaliteit te waarborgen:

- **Sprint review meeting**
  Elke twee weken vindt er een sprint review plaats samen met de producteigenaar. Het is mogelijk andere gebruikers uit te nodigen. Hierdoor kan feedback worden gegeven op het product, dit bevordert de kwaliteit.
- **Coding standaard**
  Er moet worden gehouden aan de code standaarden van FourICT. Hierdoor verbetert de kwaliteit van de software. Hiervoor wordt gebruik gemaakt van een analyseer tool die deze controle uitvoert.
- **Risicoanalyse**
  Om de kwaliteit te waarborgen wordt er in de sprintplanning een risicoanalyse uitgevoerd per user story. Dit om de testaanpak en strategie te bepalen.

3.6. Projectorganisatie

Naam	Functie	Scrum Rol
Mark A. Barto	Opdrachtnemer	Scrum master, Ontwikkelaar
Richard van Leeuwe	Opdrachtgever, Bedrijfsmentor	Producteigenaar
Henk van Ewijk	Opdrachtgever	Stakeholder

3.7. Beschrijving mijlpaalproducten

Hieronder de belangrijkste aan te leveren producten:

1. Plan van Aanpak;
2. Analyse en ontwerpdocumentatie;
3. Test documentatie;
4. Oplevering formulierensysteem in OTAP-omgeving;

3.8. Globale planning

Hieronder de globale planning van de werkzaamheden die worden uitgevoerd.

DATUM	WERKZAAMHEDEN
16-11-2015 T/M 27-11-2015	Plan van Aanpak Opstellen user stories Opstellen detail planning Opstellen analysedocument Opzetten ontwikkelomgeving Opzetten gebruikte technieken Bijwerken afstudeerverslag
30-11-2015 T/M 11-12-2015	Sprint 1
14-12-2015 T/M 25-12-2015	Sprint 2
4-1-2016 T/M 15-1-2016	Sprint 3
18-1-2016 T/M 29-1-2016	Sprint 4
1-2-2016 T/M 12-2-2016	Sprint 5
15-2-2016 T/M 26-2-2016	Sprint 6
29-2-2016 T/M 11-3-2016	Sprint 7
11-3-2016 T/M 29-3-2016	Afronden afstudeerverslag

3.9. Technische tools

Ontwikkelomgeving

Als ontwikkelomgeving (IDE) maak ik gebruik van Visual Studio 2015 Enterprise. Er zijn meerdere types van Visual Studio. Ik heb gekozen voor de Enterprise versie. Deze versie biedt extra tools om automatisch te kunnen testen. Daarnaast biedt het de mogelijkheid om UML-diagrammen te ontwerpen. Er is gekozen voor Visual Studio als ontwikkelomgeving omdat dit binnen FourICT wordt voorgeschreven.

Ontwikkelstraat

Ik maak gebruik van Visual Studio Team Services. Dit biedt een complete ontwikkelstraat aan in de cloud waarmee het mogelijk is om de broncode te beheren, projecten te plannen en traceren via een online portaal. Daarnaast beschikt het over scrum tooling, geautomatiseerde builds en test casemanagement. Er is gekozen voor Visual Studio Team Services omdat dit binnen FourICT wordt voorgeschreven.

UML

Er is gebruik gemaakt van de model techniek UML 2.0 voor het modeleren. Daarbij is o.a. gebruik gemaakt van use case diagrammen, activity diagrammen, sequence diagrammen, class diagrammen. UML wordt gebruik voor het ontwerpen van het systeem.

4. Sprint 0: “Analyse”

Scrum bestaat uit een of meerdere Sprints. Een Sprint is een korte iteratie waarin functionaliteiten die onderdeel is van het eindproduct worden opgeleverd. Er is gebruik gemaakt van Sprint 0 als voorbereidingsfase. Dit is belangrijk aangezien deze sprint ervoor zorgt dat de volgende sprints efficiënter van start kunnen gaan. Als dit niet zou gebeuren kan er onnodige tijd verloren worden in de komende sprints.

Opstellen user stories
Naar aanleiding van de globale requirements die vooraf is bepaald zijn er user stories gemaakt. User stories worden gebruikt als requirementstechniek binnen Agile om de business waarde en behoefte van de gebruiker te bepalen. Een user story bestaat uit een korte en krachtige beschrijving van wat een gebruiker wilt. Dit wordt op de volgende manier als zin beschreven:

Als <rol van de gebruiker> wil ik <doelstelling>, zodat ik <business waarde>

Bespreking user stories
In mijn rol als ontwikkelaar van het team kwam tijdens de bespreking van de user stories met de opdrachtgever (stakeholder en producteigenaar) naar voren dat er wat onduidelijkheid was over de verschillende rollen in het systeem. Er was te weinig onderscheid tussen de gebruiker van het formulier en de ontwerper van het formulier. Daarnaast waren bepaalde user stories voor meerdere uitleg vatbaar. Dit kwam doordat er als rol gebruikt werd gemaakt van gebruiker en medewerker. Dit betekende hetzelfde. Verder kwamen er ook nieuwe user stories naar voren:

- Als geregistreerde formuliergebruiker wil ik foto's kunnen uploaden, zodat ik niet handmatig met een camera dit moet gaan doen.
- Als geregistreerde formulierbeheerder wil ik handtekening kunnen zetten in het formulier, zodat er bewijs is voor akkoord.
- Als geregistreerde formulierbeheerder wil ik het thema van het formulier aanpassen, zodat ik mijn eigen huisstijl kan gebruiken.

Deze user stories zijn opgenomen in de product backlog.

Product backlog prioriteren
Samen met de producteigenaar is de product backlog lijst doorgenomen en gerangschikt van hoogste prioriteit naar laagste prioriteit. De prioritering is belangrijk om te bepalen welke user story als eerste moet worden opgepakt. Als dit niet zou gebeuren wordt niet de belangrijkste user story opgepakt bij het plannen van een Sprint. Deze prioriteit wordt bepaald door de producteigenaar op basis van de waarde die de user story voor de business heeft. De user story met de hoogste prioriteit staat bovenaan. Deze wordt als eerste geïmplementeerd. De producteigenaar zorgt ervoor dat de gehele lijst gerangschikt blijft.

Globale schatting user stories
In mijn rol als ontwikkelaar is er een schatting gegeven van de hoeveelheid ontwikkelcapaciteit die nodig is per user story. De bedoeling is om dit samen met het ontwikkelteam te doen door middel van planningspoker in de sprintplanning. Aangezien er binnen het team maar één ontwikkelaar is wordt dit door mij uitgevoerd. Dit wordt getoond onder de kolom 'Effort' en daarbij is gebruik gemaakt van Story Points met de getallen (1, 2, 3, 5, 8, 13, 21). Dit wordt uiteindelijk gebruikt om de planning van de sprint mee vast te stellen hoeveel werk er in een sprint kan worden opgepakt.

Voor het bepalen van de juiste 'Effort' is gekeken in de product backlog wat het minste werk is om te implementeren. Dit is als 1 ingeschat. Wat het meeste werk is om te implementeren is geschat op 13. Daarna is gekeken in vergelijking met de geschatte product backlog items of deze relatief meer of minder werk is. Als dit meer wordt deze geschat op 21 en anders op 8, 5, 3 of 2. Daarna wordt de volgende ingeschat op hetzelfde principe waar ook gekeken wordt of degene meer of minder werk is.

Belangrijkste user stories

Hieronder zijn de belangrijkste user stories weergegeven, voor een volledige lijst zie Analyserapport in Externe Bijlagen.

#R	Titel	Effort	Sprint
1	Als formulierbeheerder wil ik op mijn computer mijn formulieren ontwerpen, zodat ik overal gebruik kan maken van het systeem.	5	1
2	Als formulierbeheerder wil ik mij registreren in het systeem, zodat ik toegang krijg binnen het systeem.	3	1
5	Als geregistreerde formulierbeheerder wil ik een formulier kunnen invoeren met standaard invoervelden, zodat ik het formulier kan ontwerpen.	5	2
7	Als geregistreerde formulierbeheerder wil ik het resultaat als e-mail ontvangen, zodat ik het formulier digitaal via e-mail ontvangt.	3	3
8	Als geregistreerde formulierbeheerder wil ik gebruikers toekennen aan mijn ontworpen formulier, zodat mijn medewerkers deze kunnen inzien.	5	3
14	Als formuliergebruiker wil ik op mijn mobiele apparaat mijn formulieren inzien, zodat als ik onderweg ben mijn formulieren kan bekijken en versturen.	5	5
17	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat het formulier invullen, zodat ik dit niet met pen en papier moet invullen.	5	6
21	Als geregistreerde formuliergebruiker wil ik offline kunnen werken, zodat als ik onderweg ben het formulier later kan versturen.	8	7

Teststrategie bepalen
Om een kwalitatief systeem op te kunnen leveren moet er eerst goed worden getest. Het is uiteraard niet mogelijk om alles te testen van het systeem en daarom moeten er keuzes worden gemaakt. In samenspraak met de producteigenaar is bepaald wat het risico is. Het risico wordt bepaald aan de hand van de verwachte schade en de faalkans dat het kan optreden. De schade wordt in overleg met de producteigenaar bepaald en de faalkans door het ontwikkelteam. De schade en faalkans wordt gemeten met behulp van een classificatie hoog, medium en laag. Bij faalkans wordt gekeken naar de complexiteit van de geïmplementeerde user story.

Uiteindelijk wordt de risicoklasse bepaald met behulp van onderstaand tabel:

		Faalkans		
		Hoog (H)	Medium (M)	Laag (L)
Schade	Hoog (H)	A	A	B
	Medium (M)	B	B	C
	Laag (L)	B	C	C

Uiteindelijk wordt de risicoklasse gebruikt per kwaliteitsattribuut om de zwaarte van de test te bepalen. Bij zwaarte gemiddeld (Risicoklasse B) worden globale testcases opgesteld. Bij hoog (Risicoklasse A) worden gedetailleerde testcases opgesteld. Bij laag (Risicoklasse C) wordt het getest op exploratieve wijze.

Testen
Om de kwaliteit van de software te waarborgen worden tijdens iedere sprint de volgende testen uitgevoerd indien van toepassing.

Automatisch test
Bij elke check-in van de code in Visual Studio Team Service wordt een nieuwe build gemaakt. Tijdens de build worden automatisch testen uitgevoerd. Als een build niet lukt doordat een automatisch test niet lukt kun je direct actie ondernemen om het probleem op te lossen. Hierdoor kun je ervoor zorgen dat bij een wijziging alles blijft functioneren.

Unit test
Een unit test zorgt ervoor dat een wijziging in de code getest word op correctheid. De ontwikkelaar zorgt ervoor tijdens de implementatie van een user story de code wijziging wordt getest. Uiteindelijk bij een check-in van de wijziging wordt deze automatisch getest bij het maken van een build. Het gebruik van unit tests zorgt ervoor dat je zeker bent dat de implementatie werkt.

Ontwikkelomgeving opzetten
Voordat er daadwerkelijk ontwikkeld kan worden is er een ontwikkelomgeving ingericht met:

- Microsoft Visual Studio 2015 Enterprise geïnstalleerd en geconfigureerd;
- Nieuwe project aangemaakt in Visual Studio Team Services en geconfigureerd waarbij deze gebaseerd is op Scrum;

Bij het inrichten van het project in Visual Studio Team Services is de producteigenaar en de stakeholder toegevoegd. Dit is belangrijk zodat deze de planning in de gaten kunnen houden en de productbacklog kunnen rangschrikken en beheren.

Definition of Done opstellen
Er is een 'definition of done' opgesteld waarbij is aangegeven wanneer een taak daadwerkelijk klaar is. Dit is belangrijk om te bepalen wanneer een user story klaar is. Daarnaast bepaalt de 'definition of done' ook de inschatting van een user story. In Scrum is het alleen de bedoeling om documentatie op te leveren als dit een meerwaarde heeft. Aangezien documentatie een onderdeel is van het afstuderen is dit meegenomen als done criteria.

Hierbij is afgesproken dat de volgende onderdelen hieronder vallen:

- Bijwerken documentatie indien van toepassing:
    - Analyse en ontwerpdocument.
        - Bijwerken requirements indien van toepassing.
        - Bijwerken use cases indien van toepassing.
        - Bijwerken use case diagram indien van toepassing.
        - Bijwerken datamodel incl. RRM/RIM indien van toepassing.
    - Testplan document.
- Unittests/integratietests uitgevoerd en geslaagd.
- Ingecheckt in versiebeheer.

Velocity
Met behulp van een velocity kan bepaald worden hoeveel user stories er in een sprint kunnen worden opgepakt. De velocity wordt bepaald aan de hand van het gemiddelde van het aantal story points van de vorige sprints wat is opgelost. Dit is belangrijk om te kunnen bepalen of de globale planning behaald kan worden met het aantal user stories dat in de product backlog staat. Ik heb al ervaring in het werken in een Scrum-traject en daarbij is bepaald dat mijn velocity gemiddeld 10 is, in twee wekelijkse sprints.

Globale architectuur
Op basis van de globale requirements en de user stories is er nu een globaal idee ontstaan hoe de globale architectuur eruit moeten komen te zien. Het is niet de bedoeling om de gehele architectuur vooraf in detail te bepalen. Er wordt iteratief gewerkt waardoor snel een werkend stuk functionaliteit opgeleverd moet worden is het niet de bedoeling om een complete architectuur uit te bedenken. Tijdens het implementeren van een user story kunnen Architectuur beslissingen genomen worden. Het is wel wenselijk om een basis architectuur te leggen zodat in de volgende Sprints sneller van start kan worden gegaan. Hierdoor wordt het duidelijk wat voor systeem gebouwd moet worden.

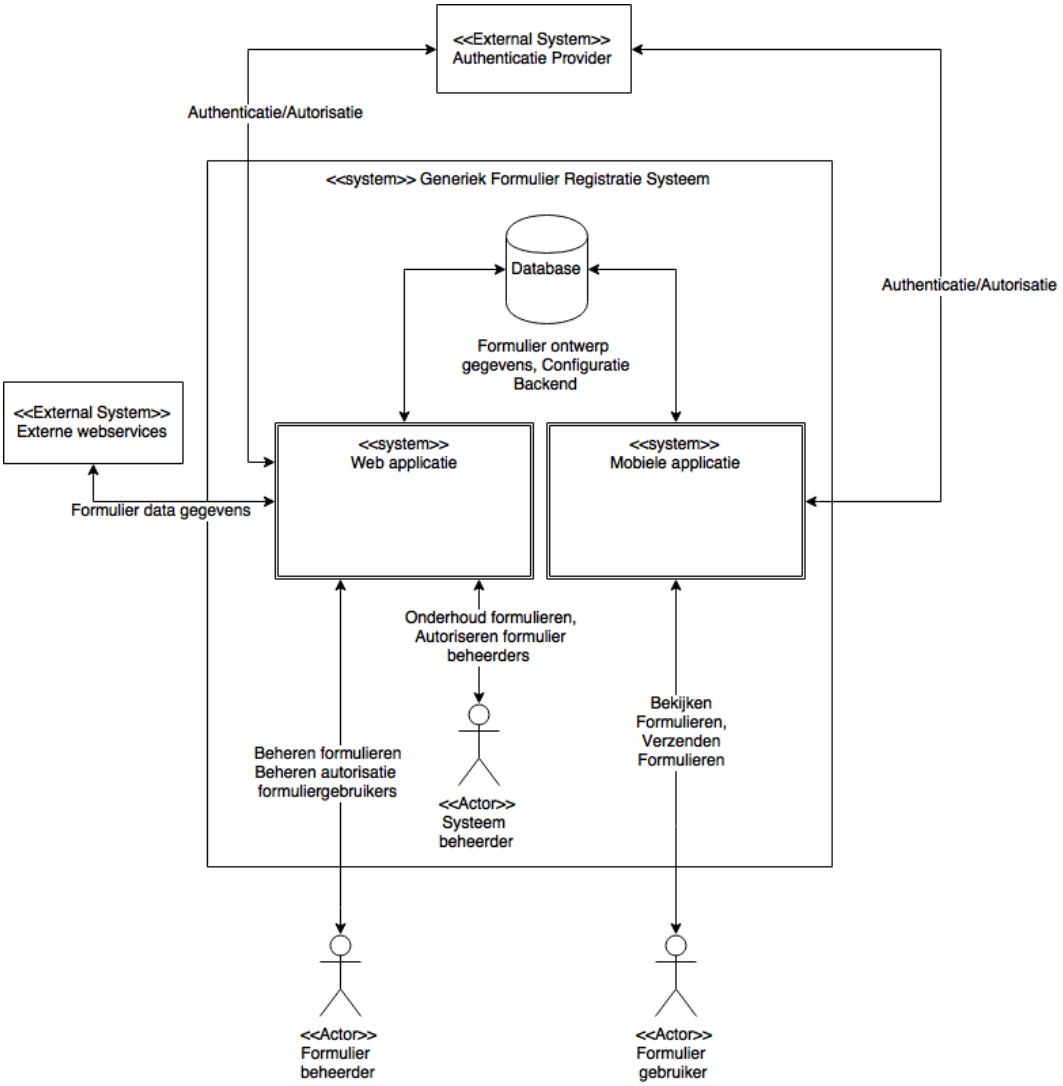
Het is belangrijk om een software-architectuur op te zetten om een kwalitatief hoogwaardig systeem op te kunnen leveren. Om aan de niet-functionele eisen te kunnen voldoen moet er rekening gehouden worden met verschillende aspecten zoals portabiliteit, schaalbaarheid, efficiëntie en koppelbaarheid.

Type applicaties
De keuze van een type applicatie is afhankelijk van de requirements en de technische beperkingen die zijn opgelegd door de opdrachtgever. De voor- en nadelen zijn uitgewerkt in Bijlage 2: Vergelijking mobiel, Windows en webapplicatie

Webapplicatie
Er is aangegeven dat er een systeem ontwikkeld moet worden dat platformonafhankelijk is en toegankelijk is vanaf iedere computer. Er is gekozen voor een webapplicatie omdat deze voldoet aan al deze criteria.

Mobiele applicatie
Er is daarnaast aangegeven dat de gebruiker van het formulier veel onderweg is en geen toegang heeft tot het internet. Ook aangegeven dat er in de toekomst gebruik moet worden gemaakt van de functionaliteiten van het mobiele apparaat. Verder is er aangegeven dat het formulier offline beschikbaar moet zijn en beschikbaar moet zijn op mobiele apparaten. Er wordt daarom ook een mobiele applicatie ontwikkeld.

Systeem context
In onderstaand context diagram wordt dit weergegeven. Op deze manier wordt een beeld gecreëerd waarbij globaal de onderdelen worden weergegeven van het systeem met daarnaast de externe entiteiten waar vanuit het systeem interactie mee heeft.

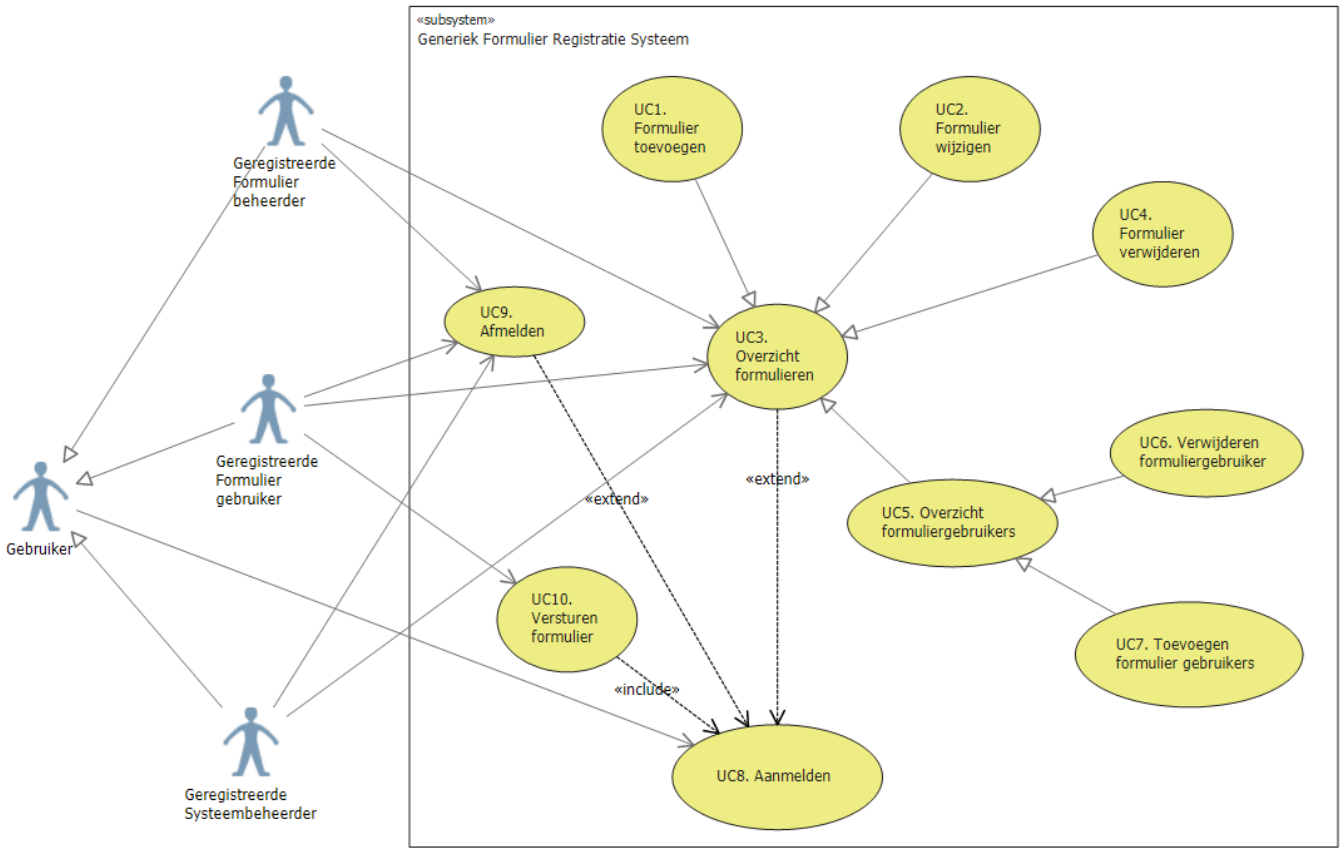


Actoren
De formulier beheerder (actor) kan via een webapplicatie verbinding maken met het systeem en van daaruit beheer uitvoeren op zijn ontworpen formulieren en formulier gebruikers toegang geven.

Daarnaast kan de formulier gebruiker (actor) via een mobiele applicatie op zijn of haar device (bijvoorbeeld mobiel, tablet) verbinding maken met het systeem en van daaruit het formulier bekijken en verzenden.

Het systeem maakt gebruik van een externe partij die de gebruikers toegang geeft voor het gebruik van het systeem. Verder zorgt het systeem door middel van externe Web Services (bijvoorbeeld e-mailservices, backoffice services) dat de data van de formulieren in de achterliggende systemen terecht komen.

Use case diagram
Er is aan de hand van de globale requirements een use case diagram opgezet. Dit geeft een goed beeld van de actoren en de onderlinge relaties tussen de verschillende use cases.



5. Sprint 1: “Initiële opzet”

5.1. Planning

Op 30 november is er een sprint planning geweest met de producteigenaar en stakeholder waarbij gekeken is naar de product backlog. Uit de product backlog is een selectie gemaakt van de eerste user stories. De selectie is gebaseerd op basis van mijn velocity. Mijn velocity is 10 waarbij ervan uit is gegaan dat ik binnen twee weken 10 storypoints kan afronden.

Daarbij zijn wij door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld. Hieronder een overzicht van de user stories die opgepakt zijn in deze sprint.

#PBI	#R[3]	#UC[4]	Titel	Effort[5]
317	1	-	Als formulierbeheerder wil ik op mijn computer mijn formulieren ontwerpen, zodat ik overal gebruik kan maken van het systeem.	5
318	2	UC8	Als formulierbeheerder wil ik mij registreren in het systeem, zodat ik toegang heb binnen het systeem.	3
319	3	UC8	Als formulierbeheerder wil ik mij aanmelden in het systeem, zodat ik toegang krijg tot het systeem.	2

Er is beslist samen met de producteigenaar om #PBI 317 als eerste op te pakken. Dit aangezien deze user story ervoor moet zorgen dat er een basissysteem wordt geïmplementeerd. Het basissysteem bevat de eerste opzet van de projectstructuur en de detaillering van de architectuur. Anderzijds als eerst begonnen wordt met #PBI 318 kan deze niet geïmplementeerd worden omdat er nog geen basis opzet is.

Taken vaststellen

Nadat de user stories zijn geselecteerd moeten er taken worden vastgesteld. De taken worden vastgesteld om de stappen te definiëren die nodig zijn om de user story te implementeren. De user story bevat niet genoeg detailinformatie om te bepalen wat er geïmplementeerd moet worden. Om te bepalen wat er geïmplementeerd moet worden moet de user story besproken worden met de producteigenaar. Tijdens de bespreking met de producteigenaar wordt duidelijk aan welke criteria er voldaan moet worden. Dit wordt ook wel de acceptatiecriteria genoemd.

De acceptatiecriteria is belangrijk om te bepalen wanneer een user story klaar is zodat het voldoet aan de ‘definition of done’. Ik heb gekozen de acceptatiecriteria uit te werken in een use case. Dit heb ik gedaan omdat de acceptatiecriteria te weinig diepgang heeft. Daarnaast gebruik ik de use case als testbasis om te testen. Een use case geeft beter weer hoe inhoudelijk de interactie loopt tussen de gebruiker en het systeem.

[3] Nummer van de requirement.
[4] Nummer van de use case, zie analyse document voor uitwerking van de use case.
[5] Moeilijkheidsgraad uitgedrukt in storepoints.

Nadat de use case is opgezet heb ik de taken bepaald die nodig zijn om de user story te implementeren. Een taak wordt gekoppeld aan een user story en daar wordt een hoeveelheid werk in tijd afgegeven. Voor het bepalen van de tijd van een taak wordt gebruik gemaakt van 4, 8, 12 of 16 uur. Volgens Scrum wordt 8 uur geadviseerd. Dit aangezien een taak tijdens de Daily Scrum klaar is of niet. Daarnaast tijdens ziekte is de kans groter dat de taak klaar is en zal er niet te veel achterstand op worden gelopen. Als dit langer zou duren kan de taak niet worden afgerond. Ik heb gekozen om hiervan af te stappen aangezien ik ten eerste de enige ontwikkelaar ben in het team. Er wordt geen dagelijkse Scrum gedaan. De taken moeten daarnaast voldoen aan de 'definition of done' dus daarom is het belangrijk om taken ook hierin mee te nemen. Tijdens de Sprint planning is er door gesprekken met de producteigenaar een globaal idee ontstaan van de werkzaamheden die verricht moeten worden om een user story te implementeren.

Hier is een voorbeeld van de taken die toegekend zijn aan product backlog item #317:

#T[6]	#PBI	Titel	Tijd[7]
356	317	Architectuur bepalen	8
357	317	Projectstructuur opzetten	8
358	317	Documenteren	16

Risicoklasse

Na het vaststellen van de taken en het opzetten van de use case word de risicoklasse ingeschat. De risicoklasse bepaalt uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICOKLASSE	ZWAARTE
UC8	**Functionaliteit**	H	L	B	●●
	Beveiliging	H	L	B	●●

● Er worden geen testcases opgesteld maar er wordt getest op exploratieve wijze.
●● Er worden globale test cases opgesteld.
●●● Er worden gedetailleerde test cases opgesteld.

De schade is vastgesteld in overleg met de producteigenaar. De faalkans wordt daarentegen door de ontwikkelaar bepaald en is vastgesteld bij kenmerk functionaliteit op Laag. Dit aangezien er gebruik wordt gemaakt van een externe authenticatie provider waar de logica in zit. Hierdoor is de faalkans minder aangezien er minder functionaliteit wordt geraakt. Voor beveiliging is de faalkans ook laag omdat de authenticatie provider aan hoge beveiligingseisen voldoet. De testen zijn uitgewerkt in hoofdstuk 5.3

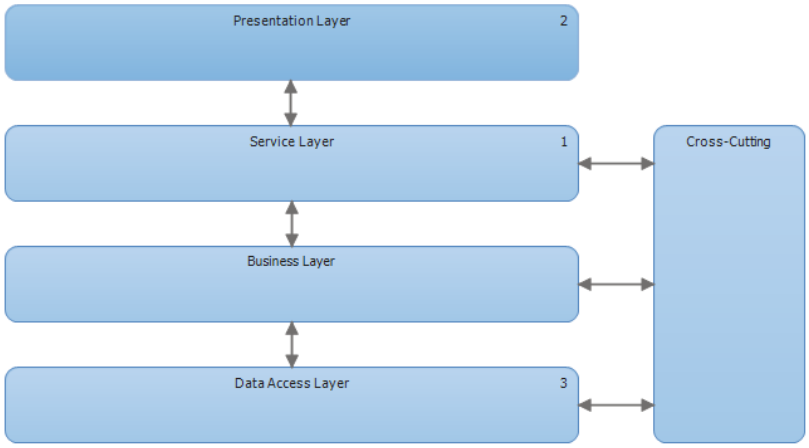
De sprint goal is: Opzet architectuur.

[6] Nummer van de taak gekoppeld aan de product backlog item.
[7] Hoeveelheid werk van de taak uitgedrukt in aantal uren.

5.2. Uitvoering

Na het bepalen van de globale architectuur is het nu duidelijk dat er als systeem een webapplicatie ontwikkeld moet worden. Een webapplicatie is gebaseerd op een client-server model waarbij op de cliënt een webapplicatie draait in de webbrowser. Een webapplicatie bestaat uit een gebruikersinterface, business logica en data.

Logische architectuur
Er is aangegeven dat de applicatie schaalbaar en eenvoudig uitbreidbaar moet zijn. De applicatie wordt daarom ontworpen op basis van een lagen structuur. Dit is om de kwaliteit van het systeem te waarborgen. Hierbij heeft elke laag zijn eigen verantwoordelijkheid en communiceert alleen met de bovenliggende of onderliggende laag. Dit heeft als grote voordeel dat de implementatie van een laag eenvoudig kan worden vervangen.

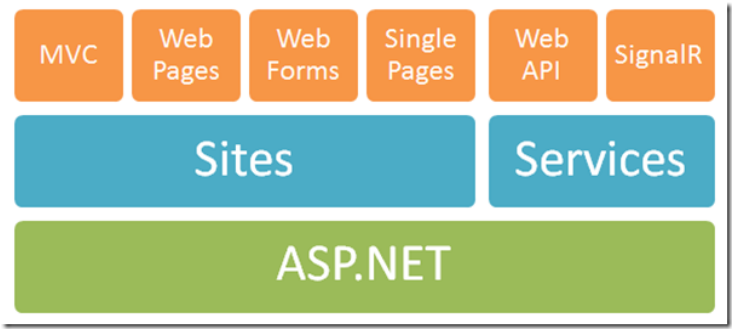


Figuur 5.1 Lagen structuur

De volgende lagen worden gebruikt:

- **De presentatie laag:** Dit is de laag die beschrijft hoe een applicatie communiceert met de gebruiker. Meestal gaat dit via een grafische userinterface (GUI).
- **De service laag:** Dit is de laag die verantwoordelijk is voor de communicatie tussen de presentatie laag.
- **De business laag:** Bevat logica die verantwoordelijk is voor de business denk daarbij aan validatieregels.
- **De data access laag:** Dit is de laag die verantwoordelijk is voor de communicatie met de databronnen (databases of Web Services).
- **Cross-cutting:** Dit is een aspect die verantwoordelijk is voor gemeenschappelijke functionaliteit tussen de verschillende lagen denk bijvoorbeeld aan logging.

Keuze UI Framework
De keuze van het Framework is gebaseerd op de technische beperking en de niet-functionele eisen. Het ASP.NET Framework is als technische beperking opgegeven. Het ASP.NET Framework bestaat uit de volgende onderdelen.



Figuur 5.2 ASP.NET Architectuur

Voor het opzetten van een webapplicatie kan er gebruik worden gemaakt van de volgende Frameworks: MVC (Model, View en Controller), Web Pages, Web Forms en Single Pages (SPA).

Er is aangegeven dat het systeem makkelijk uitbreidbaar en onderhoudbaar moet zijn. Verder is er aangegeven dat er offline moet worden kunnen gewerkt. Daarnaast moet de applicatie binnen een seconde reageren. Verder is gebruiksvriendelijkheid een belangrijk punt. Daarnaast moet in de toekomst gebruik kunnen worden gemaakt van de functionaliteiten van het mobiele apparaat.

Bij MVC, Web Pages en Web Forms wordt de HTML gegenereerd op de server. Dit betekent extra laadtijden naar de Server toe. Daardoor zijn de laadtijden een stuk langzamer en is er ook afhankelijkheid met de server. Een SPA is niet afhankelijk van de server alleen voor de initiële load om de bestanden (HTML, CSS, JS) op te halen. Dit betekent snelle laadtijden, afgezien van de initiële load, en minder round-trips naar de server. Voor offline gebruik is het daarnaast moeilijk implementeerbaar bij het gebruik van MVC, Web Pages en Web Forms. Omdat een SPA niet afhankelijk is van de server, behalve bij de initiële load, worden de pagina's direct vanuit de browser cache geladen. Verder is gebruiksvriendelijkheid een belangrijk punt. Een SPA heeft als doel om een vlotte gebruikservaring te bieden waarbij het aanvoelt alsof de applicatie op de eigen computer draait. Het nadeel is de zoekmachine optimalisatie aangezien een SPA maar uit een pagina bestaat.

Gebaseerd op de eisen van het systeem is daarom gekozen voor een SPA.

Keuze SPA Framework
Er zijn verschillende SPA Frameworks beschikbaar onder anderen DurandalJS en AngularJS. Het grote verschil tussen beide frameworks is dat DurandalJS gebruik maakt van Knockout voor de databinding. AngularJS maakt gebruik van zijn eigen databinding. In praktijk heb ik ervaren dat het gebruik van databinding via Knockout niet prettig ervaren. Om databinding binnen Knockout toe te passen moet er gebruik worden gemaakt van observables. Observables zorgen ervoor dat automatisch gedetecteerd word als het model wijzigt waardoor de UI ook wordt aangepast. Bij Knockout moet per variabele aangegeven worden of iets in de UI wijzigt door expliciet aan te geven dat het een Observable is. Dit resulteert in meer ontwikkeltijd. Bij het gebruik van AngularJS is dit niet nodig. AngularJS biedt zijn eigen databinding. Hierdoor is het gebruik eenvoudiger waardoor er minder kans is op fouten. Ik heb daarom gekozen voor AngularJS

AngularJS
Met AngularJS worden SPA-applicaties ontwikkeld die gebruik maken van de model-view-controller architectuur. AngularJS is gedeeltelijk onafhankelijk, behalve bij de initiële load, van een server en door middel van een service wordt de applicatie van data voorzien. Verder biedt het voorzieningen voor het gebruik van HTML-templates, filtering en sortering.

AngularJS bestaat uit verschillende modules die op zichzelf staan (loosely coupled) en die met behulp van dependency injection gebruikt kunnen worden in de applicatie. Hierdoor is het mogelijk om deze modules eenvoudig te testen en maakt het erg onderhoudsvriendelijk. Daarnaast biedt het de mogelijkheid om door middel van declaratieve wijze Single Page Applications te bouwen.

Keuze Service
Om een SPA-applicatie van data te voorzien moet er gebruik worden gemaakt van een service. Een service moet ervoor zorgen dat data uit een database die op een centrale server staat wordt opgehaald. Binnen het ASP.NET Framework is er keuze uit SignalR en Web API. SignalR wordt gebruik om realtime date te zenden naar een cliënt. Een Web API is een RESTful[8] http-service die zorgt dat de data word opgehaald en ondersteund verschillende cliënts zoals browsers en mobiele apparaten. Er wordt daarom gebruik gemaakt van een WebAPI.

Keuze template voor ontwerp
Er is aangegeven dat een gebruiksvriendelijk systeem erg belangrijk is. De producteigenaar heeft daarnaast aangegeven dat de grafisch vormgeving erg belangrijk is van een webapplicatie voor de eerste indruk. Aangezien ik geen ervaring heb in de grafische vormgeving van een webapplicatie heb ik onderzocht welke opties er zijn. Het is mogelijk om een grafische vormgever in te huren. Een andere optie is het kopen van een template. Het voordeel van een grafische vormgever is dat het ontwerp uniek is en exact voldoet aan de eisen van de opdrachtgever. Een groot nadeel zijn de kosten en de implementatie tijd. Bij het kopen van een template is het grote voordeel de lage kosten en de korte implementatie tijd en de compleetheid van de template. Het nadeel is dat de template niet uniek is en niet exact voldoet aan de eisen van de opdrachtgever. De opdrachtgever heeft gekozen om een template te kopen.

De template is gebaseerd op Bootstrap en moderne web technologie (CSS3 & HTML5). Bootstrap is een CSS-framework die het mogelijk maakt om eenvoudig en snel responsive websites mee op te zetten die werkt op verschillende desktops en mobiele apparaten. Responsive websites zijn websites die zich automatisch aanpassen aan de grootte van het scherm. Dit heeft als grote voordeel dat het niet voor elk apparaat noodzakelijk is om een aparte site te ontwikkelen met een andere vormgeving.

Het gebruik van deze template zorgt ervoor dat de ontwikkelaar zich volledig kan focussen op het ontwikkelen van de applicatie en zich niet bezig hoeft te houden met de vormgeving.

[8] https://en.wikipedia.org/wiki/Representational_state_transfer

Authenticatie en autorisatie
Tijdens de sprint planning bij het bepalen van de user story is aangegeven door de producteigenaar dat beveiliging een hoge prioriteit heeft. Door de nieuwe meldplicht van datalekken[9] per 1 Januari 2016 is beveiliging extra belangrijk. Dit bepaald dat een ernstige data lek gemeld moet worden bij de autoriteiten. Niet voldoen hieraan kan een boete opleveren. Daarnaast een mogelijk moet zijn om in de toekomst via verschillende manieren zoals bedrijfsnetwerken of sociale netwerken aan te melden. Dit heeft mij ertoe besloten om te onderzoeken welke andere mogelijkheden er zijn.

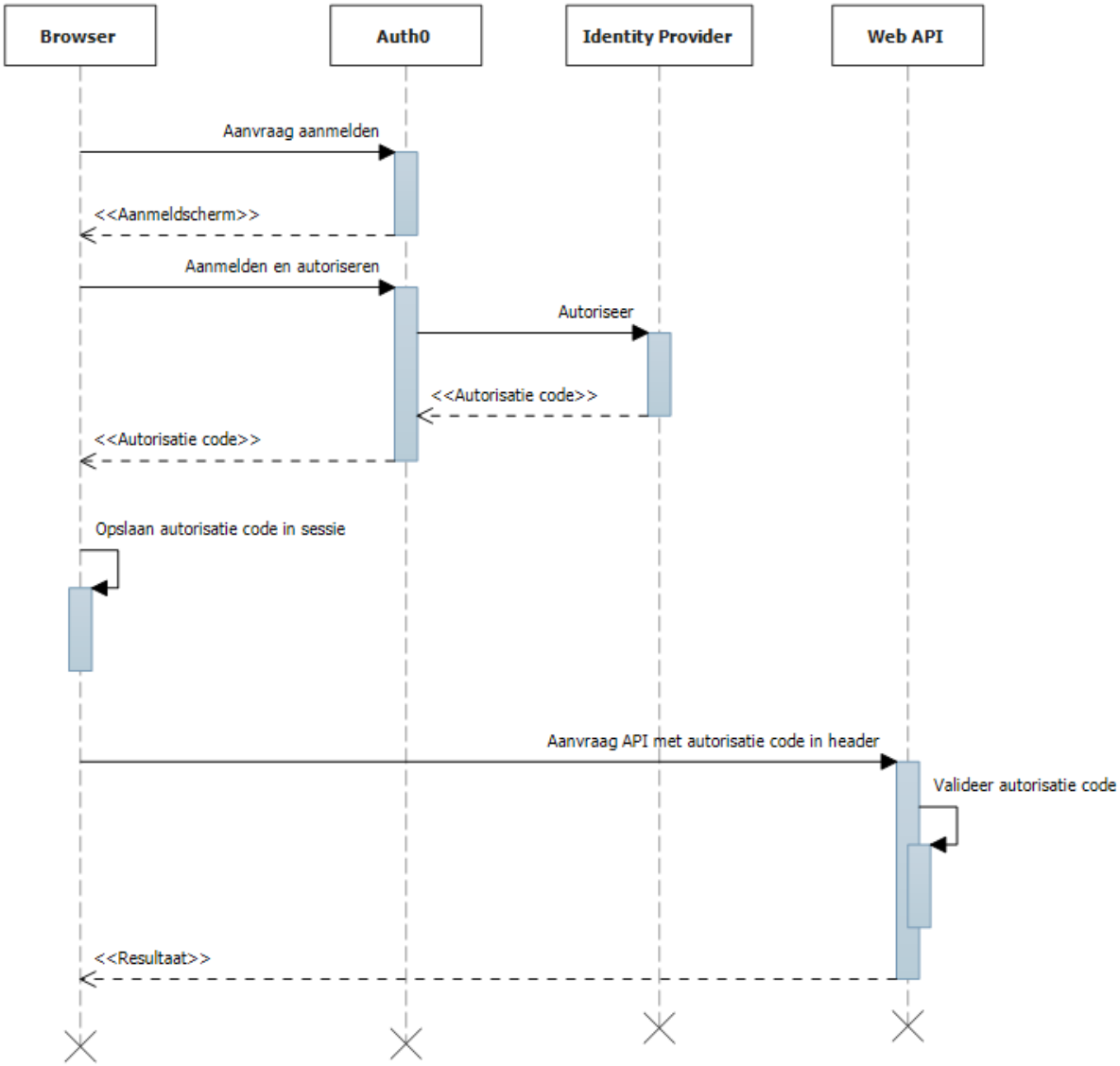
Voor het autoriseren van gebruikers zijn er meerdere manieren om authenticatie mogelijk te maken. Het is mogelijk om zelf een implementatie te doen of gebruik te maken van een externe partij. Het voordeel van een eigen implementatie is dat er geen afhankelijkheid is met een externe partij. Daarnaast blijven de gebruikersgegevens privé en ben je er zeker van dat de gegevens niet ongeoorloofd worden gedeeld. Het nadeel is de extra ontwikkeltijd voor het implementeren van het authenticatie proces. Verder het risico op een beveiligingsrisico. Het implementeren van two-factor authenticatie is niet eenvoudig.

Er is daarom onderzoek gedaan naar externe authenticatie providers (Auth0 en Authrocket). Hierbij is gekeken naar de volgende functionaliteiten: beveiliging, integratie met verschillende platformen, prijs, uitbreidbaarheid, gebruik van open standaarden. Zie de uitwerking van de voor- en nadelen in Bijlage 4: Vergelijking Auth0 en AuthRocket

Auth0
Er is gekozen voor een externe partij om zodoende het gehele authenticatie en autorisatie proces te versimpelen. Dit betekent minder complexiteit bij het bouwen en hierdoor risico's te beperken om beveiligingslekken te vermijden. Uit Bijlage 4: Vergelijking Auth0 en AuthRocket is aftelezen dat Auth0 voldoet aan de verschillende eisen die zijn opgesteld. Auth0 biedt de mogelijkheid om eenvoudig door middel van een SDK te autoriseren en te authentiseren. Verder is het gebaseerd op openstandaarden zodat er geen afhankelijkheid is met de leverancier om vendor lock-in[10] tegen te gaan. Integratie met vele verschillende platformen. Open source. Grote community. Het nadeel van het gebruik van Auth0 is de beschikbaarheid, prijs en dat de gebruikersdata niet in eigen beheer is.

Authentiseren en toegang geven van een formulierbeheerder
De Identity Hub biedt een API (Application Programming Interface) aan die het mogelijk maakt voor ontwikkelaars om administratieve taken uit te kunnen voeren. Er moet hierbij gedacht worden aan het toegang verschaffen van gebruikers. Het is belangrijk om te weten hoe de authenticatie in zijn werk gaat en daarvoor is onderstaand sequentiediagram opgesteld:

[9] https://autoriteitpersoonsgegevens.nl/nl/melden/meldplicht-datalekken
[10] https://nl.wikipedia.org/wiki/Vendor_lock-in



1. Bij het starten van de applicatie in de browser vraagt hij via de Identity Hub (Auth0) het aanmeldscherm.
2. Na het opgeven van de authenticatie gegevens gaat de aanvraag naar de Identity Hub (Auth0).
3. De Identity Hub controleert welk Identity Provider er is geselecteerd en valideert dit en geeft een autorisatie code (JWT) terug. Dit gaat via een Identity Provider als er bijvoorbeeld gebruik wordt gemaakt van Social logins zoals Twitter/Google etc.
4. De autorisatie code (JWT) wordt uiteindelijk op de cliënt opgeslagen om zodoende niet telkens opnieuw geautoriseerd te worden.
5. Bij het opvragen van gegevens via de eigen Web API wordt deze autorisatie code (JWT) in de header meegestuurd.
6. De autorisatie code (JWT) wordt gevalideerd om er zeker te zijn van de toegang tot het gebruik van de Web API.
7. Uiteindelijk wordt er een resultaat teruggestuurd vanuit de Web API.

Authenticatie-methode
Binnen de autorisatie-provider zijn er verschillende authenticatie mogelijkheden:

- Het is mogelijk om gebruikers te autoriseren door gebruik te maken van de interne database.
- Het is mogelijk om gebruikers te autoriseren via sociale netwerken zoals Facebook, Twitter, Google, of elk andere Identity Provider die OAuth ondersteunt.
- Het is mogelijk om gebruikers te autoriseren via bedrijfsnetwerken zoals LDAP, Office 365, Active Directory, Google Apps, etc.
- Het is mogelijk om gebruikers te autoriseren zonder wachtwoord met behulp van een vingerafdruk of een code per sms.

Uit onderzoek van Splashdata [11] waarbij twee miljoen wachtwoorden zijn geanalyseerd blijkt dat het meest gebruikte wachtwoord 123456 van 2015 nog steeds bovenaan staat in de lijst. Hieruit blijkt dat een wachtwoord niet veilig genoeg is en makkelijk te kraken is. Daarnaast blijkt uit een ander onderzoek dat door World's Biggest Data Breaches [12] is uitgevoerd dat het nog geregeld voorkomt dat data met wachtwoorden worden gestolen. Er is daarom gekozen om gebruikers te autoriseren zonder wachtwoord met behulp van een code per e-mail.

Onafhankelijk van Auth0
Het is niet wenselijk om afhankelijk te zijn van een bepaalde gekozen externe partij zoals Auth0. Er is daarom een autorisatie service ontwikkeld. Het grote voordeel van het gebruik van services is dat ze binnen de applicatie geïnjecteerd kunnen worden, daardoor makkelijk kan testen en dat het singletons zijn. Als er een andere externe partij wordt gekozen in de toekomst is het voldoende om alleen de logica in de service aan te passen.

Controle autorisatie na herladen pagina
Binnen een SPA-architectuur draait alles na een initieel request in de browser. Daarbij moet rekening worden gehouden dat bij het herladen van de browser of bij het laden van een nieuwe pagina opnieuw geautoriseerd moet worden. Dit is uiteraard niet gebruiksvriendelijk. Het autorisatie token dat is opgeslagen bij het initieel request moet daarom opnieuw gevalideerd worden. Dit moet gedaan worden door de Identity Hub zodoende dat niet opnieuw geautoriseerd moet worden.

Token meesturen in header
Om de autorisatie token door te geven in de header bij een aanroep naar een API wordt er gebruik gemaakt van een interceptor. Dit is een generieke manier om binnen een http-aanvraag extra informatie mee te versturen. Hierdoor is het mogelijk om bij een aanvraag van de Web API de autorisatie token mee te geven die lokaal is opgeslagen. De Web API ontvangt het autorisatie token en kan valideren of de gebruiker toegang heeft tot de aanvraag.

[11] https://www.teamsid.com/worst-passwords-2015/
[12] http://www.informationisbeautiful.NET/visualizations/worlds-biggest-data-breaches-hacks/

5.3. Testen

Functionaliteit
Voor het controleren van het kenmerk kwaliteitsattribuut functionaliteit zijn er globale test cases (TC1 t/m TC6) opgezet. Onderstaand een voorbeeld van de globale testcases. Voor een volledige lijst zie Testrapport in Externe bijlagen.

ID #	Beschrijving	Sprint	Use case
TC1	Aanmelden zonder e-mail adres.	1	UC8
TC2	Aanmelden met onjuist e-mail adres.	1	UC8
TC5	Aanmelden met onjuist verificatie code .	1	UC8
TC6	Aanmelden met juiste gegevens.	1	UC8

Testnummer	Data		Verwacht resultaat	Werkelijk resultaat
	email	Verificatiecode		
TC1	<leeg>	nvt	Er kan geen leeg e-mail adres ingevuld worden.	V
TC2	test@	nvt	Foutmelding onjuist e-mail adres	V
TC5	mark@fourict.nl	12345	Foutmelding onjuiste verificatie code ingevuld	V
TC6	info@fourict.nl	QWRTYDDT	Autorisatie gelukt	V

Beveiliging
Bij beveiliging is gecontroleerd of de gegevens veilig via een versleutelde verbinding worden gestuurd. Daarnaast is gecontroleerd of de gegevens van de autorisatietoken met encryptie is opgeslagen.

5.4. Sprint review

Op 11 december 2015 is er een sprint review geweest waarbij de authenticatie en de opzet van de webapplicatie is gedemonstreerd aan de producteigenaar en stakeholder. Tijdens de demonstratie bleek er nog een probleem te zitten bij het herladen van de pagina waardoor opnieuw aangemeld moet worden. Er is daarvoor een type bug aangemaakt in Visual Studio Team Services zodat dit kan worden meegenomen in de product backlog.

Daarnaast was de vraag om het aanmelden voor elke gebruiker open te stellen zodat iedereen zich kan aanmelden als formulierbeheerder. Er zijn verschillende rollen waaronder een formuliergebruiker, formulierbeheerder en beheerder. Nu was ook de bedoeling dat de formuliergebruiker ook formulierbeheerder is. Een gebruiker kan daarbij beheerder zijn van zijn eigen ontworpen formulieren. Daarnaast kan de gebruiker ook zijn formulieren inzien waar hij toegang toe heeft gekregen van andere formulierbeheerders. Dit punt is opgenomen in de product backlog.

5.5. Retrospective

Wat ging goed?
- Communicatie met de producteigenaar, snel antwoord op vragen.
- Tevreden stakeholder en producteigenaar over het snel tonen van resultaat.

Wat ging niet goed?
- Tijdens de sprint review bleek dat elke gebruiker zich kan aanmelden als formulierbeheerder. Dit had tijdens de sprint planning eruit moet komen. Aangezien er gebruik wordt gemaakt van korte sprints kon snel ingespeeld worden op deze wijziging. Dit is het voordeel van korte sprints.

Welke acties moeten er komen om dit te voorkomen?
- Tijdens de sprint planning de vragen beter formuleren zodat dit tijdens de sprint planning eruit was gekomen.

6. Sprint 2: “Formulier ontwerpen”

6.1. Planning

Op 11 december 2015 is na de demonstratie een nieuwe sprint planning geweest met de producteigenaar en stakeholder waarbij gekeken is naar de globale planning. Er is geen wijziging in prioritering geweest dus de globale planning is hetzelfde gebleven. Uit de product backlog is een nieuwe selectie gemaakt op basis van 10 storypoints. Daarbij zijn wij door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld.

#PBI	#R	#UC	Titel	Effort
321	4	UC3	Als geregistreerde formuliergebruiker wil ik een overzicht van mijn formulieren, zodat ik het juiste formulier kan bewerken.	2
322	5	UC1	Als geregistreerde formuliergebruiker wil ik een formulier kunnen invoeren met standaard invoervelden, zodat ik het formulier kan ontwerpen.	5
232	6	UC1	Als geregistreerde formuliergebruiker wil ik restricties kunnen opgeven in het formulier, zodat er minder fouten ontstaan.	3

Risicoklasse

Na het opzetten van de use case wordt de risicoklasse ingeschat. De risicoklasse bepaalt uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICOKLASSE	ZWAARTE
UC1	Functionaliteit	H	H	A	●●●
	Gebruiksvriendelijkheid	M	H	B	●●
	Continuïteit	H	L	B	●●
UC3	Functionaliteit	H	M	A	●●●
	Performance	L	L	C	●
	Continuïteit	H	L	B	●●
	Gebruiksvriendelijkheid	M	L	C	●
	Portabiliteit	H	L	B	●●

De testen zijn uitgewerkt in hoofdstuk 6.3

De sprint goal is:

- Het ontwerpen van een standaardformulier met validatie voor een formulierbeheerder.

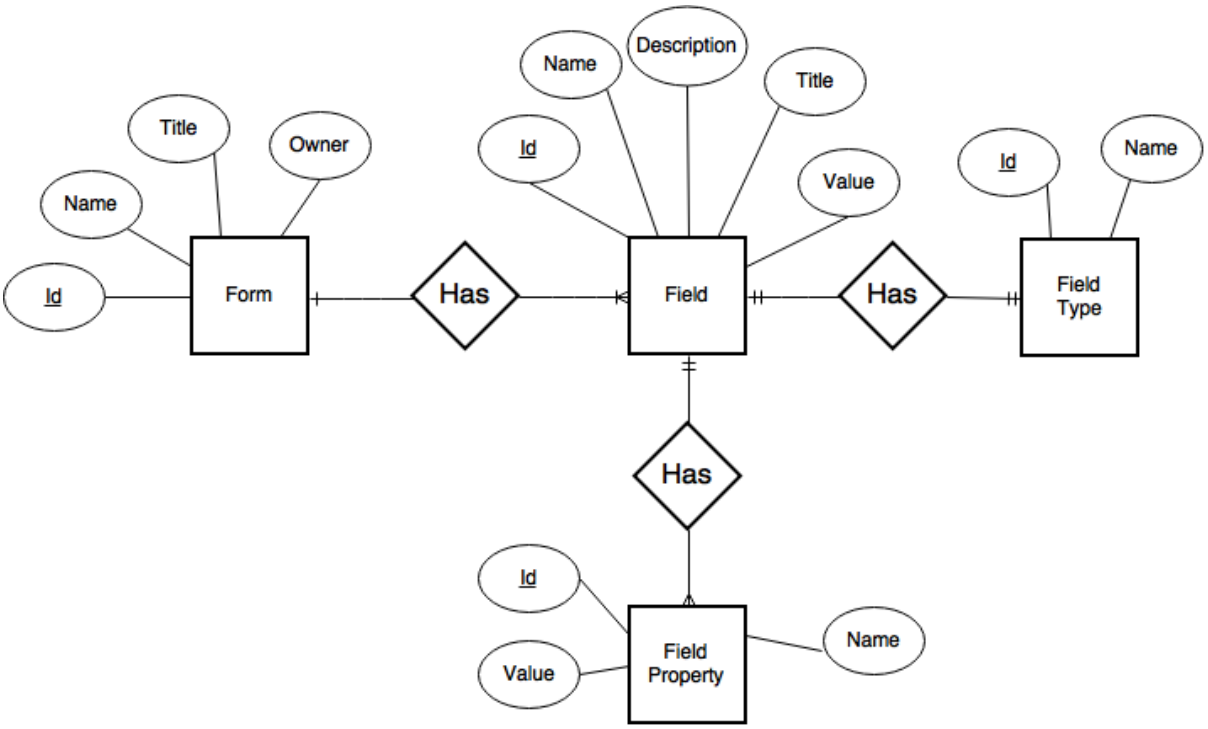
6.2. Uitvoering

Database ontwerpen
Voordat er een database kan worden ontworpen moet er eerste geanalyseerd worden welke gegevens er nodig zijn. Tijdens het bespreken van de user stories met de producteigenaar is duidelijk geworden welke informatie nodig is. Deze heb ik vertaald naar een entity-relationship diagram (E-R diagram). Een E-R diagram geeft een visuele weergave van de verschillende entiteiten, bijbehorende attributen en de relatie onderling weer. Bij het opstellen heb ik rekening gehouden met de eisen onderhoudbaarheid en uitbreidbaarheid. Daarnaast met de requirements.

Bij het ontwerpen van onderstaand model is rekening gehouden met alle user stories in deze Sprint. Het ontwerpen, tonen en valideren van formulieren. Deze zijn meegenomen bij het ontwerp van onderstaand E-R diagram.

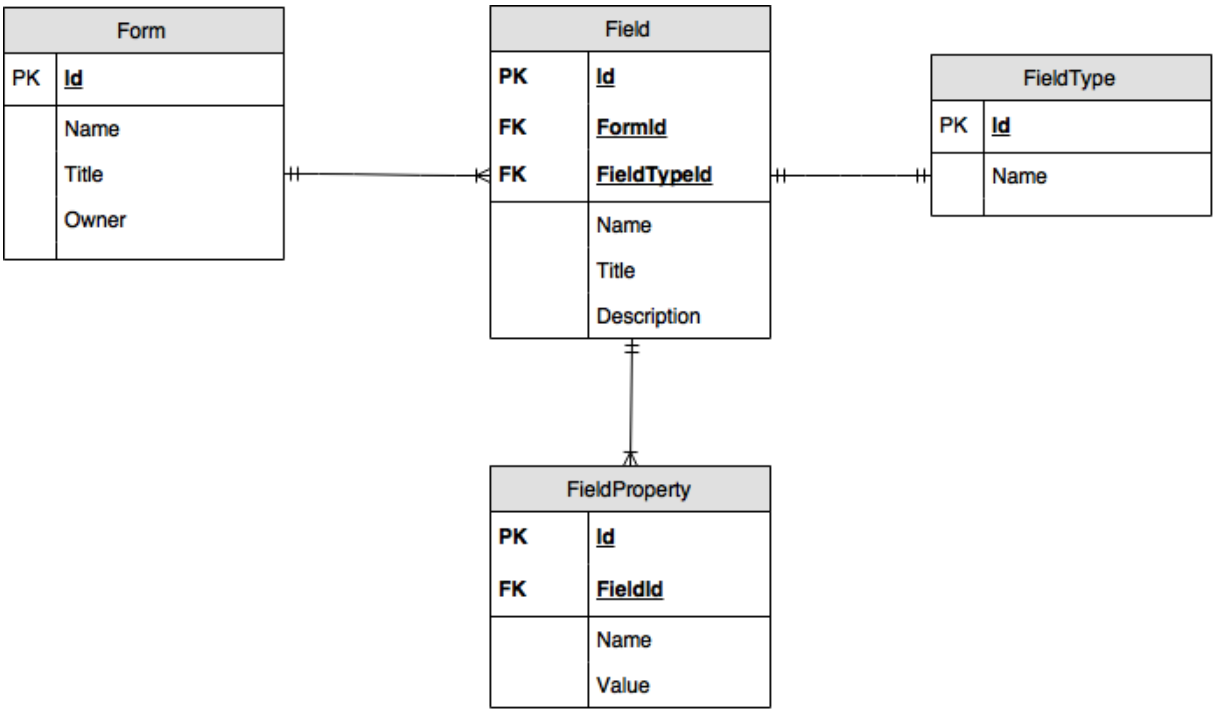
De volgende keuzes zijn gemaakt:

- Een formulier wordt opgezet door een gebruiker. Om te bepalen welk formulier door welke gebruiker is gemaakt is er een veld ‘owner’ toegevoegd. Dit bevat de unieke identificatie van de gebruiker binnen de autorisatie provider.
- Er is aangegeven dat er rekening moet worden gehouden met nieuwe type velden. Daarom is een aparte entiteit FieldType gedefinieerd. Deze bevat het type veld (Tekstveld, Selectbox). Deze keuze is gemaakt zodat een nieuw type veld eenvoudig kan worden toegevoegd om aan de uitbreidbaarheid en onderhoudbaarheid te voldoen.
- Er is een aparte tabel FieldProperty. Deze bevat de specifieke eigenschappen van een veld. Een type veld heeft namelijk niet overal dezelfde eigenschappen.



Een formulier (Form) heeft een naam, titel en een eigenaar. Een formulier bevat meerdere velden. Een veld (Field) heeft een naam, titel en omschrijving en een standaardwaarde. Een veld is van een bepaald veldtype. Een veldtype (FieldType) heeft een naam zoals een tekstveld, selectie vak, etc. Een veld kan bestaan bijbehorende eigenschappen. Een veldeigenschap (FieldProperty) heeft een naam en waarde. Een naam kan bijvoorbeeld ‘placeholder’ zijn en als waarde kan deze ‘Vul hier uw naam in’ hebben.

Na het opstellen van de E-R diagram is deze omgezet naar een logisch datamodel.



Bij het omzetten naar een fysiek datamodel is er een datadictionary opgemaakt. De datadictionary geeft de beschrijving weer van de gebruikte entiteiten inclusief de attributen. Zie analyserapport in Externe Bijlagen, hoofdstuk database voor een uitwerking van de datadictionary.

Naam	Type	Inhoud	Null	Lengte	Beschrijving	Opmerking
Id	uniqueidentifier	5BA851CF-B8BA-E511-9BEE-54271EAC3402	N		Unieke identificatie	PK
FormId	uniqueidentifier	D19FFB23-A2BA-E511-9BEE-54271EAC3402	N		Unieke identificatie naar formulier	FK naar Form
FieldTypeId	uniqueidentifier	75585D25-11BA-E511-9BEE-54271EAC3402	N		Unieke identificatie naar veldtype	FK naar FieldType
Name	varchar	Naam	N	50	Naam die gebruikt word	

					bij het versturen van formulier	
Title	varchar	Uw naam	Y	200	Titel van het veld die als label dient in het formulier	
Description	varchar	Geef hier uw naam op	Y	max	Omschrijving van veld	

Uiteindelijk is er een relationeel representatie model (RRM) model ontworpen. Een voorbeeld uit het RRM (Zie analyserapport in Externe Bijlagen, hoofdstuk database).

Field (Id, *FormId*, *FieldTypeId*, Name, Title, Description)

- FormId is een vreemde sleutel en verwijst naar Id in Form, FormId mag niet NULL zijn.
- FieldTypeId is een vreemde sleutel en verwijst naar Id in FieldType, FieldTypeId mag niet NULL zijn.
- Als een formulier (Form) verwijderd wordt moeten de velden (Field) ook met dezelfde FormId verwijderd worden, constraint op tabel niveau.

Daarna is de relationeel implementatie model (RIM) model ontworpen. Een voorbeeld uit het RIM (Zie analyserapport in Externe Bijlagen, Hoofdstuk database).

```
CREATE TABLE Field (
        Id UNIQUEIDENTIFIER NOT NULL DEFAULT NEWID(),
        FormId UNIQUEIDENTIFIER NOT NULL,
        FieldTypeId UNIQUEIDENTIFIER NOT NULL,
        Name VARCHAR(50) NOT NULL,
        Title VARCHAR(200) NULL,
        Description VARCHAR(MAX) NULL,
        PRIMARY KEY (Id),
        CONSTRAINT FK_Form FOREIGN KEY(FormId) REFERENCES Form(Id)
                ON UPDATE CASCADE
                ON DELETE CASCADE,
        CONSTRAINT FK_Field FOREIGN KEY(FieldTypeId) REFERENCES FieldType(Id)
                ON UPDATE CASCADE
                ON DELETE NO ACTION
)
```

Service ontwikkelen
In de vorige Sprint is er naar voren gekomen dat er gebruik wordt gemaakt van een WebAPI. Een WebAPI zorgt voor ervoor dat de SPA-applicatie van data wordt voorzien vanuit een database. Om te werken met relationele data vanuit een database wordt er gebruik gemaakt van een object-relational mapper (ORM). Dit zorgt ervoor dat gegevens vanuit een relationele database omgezet worden naar entiteiten (objecten). De ontwikkelaar heeft op deze manier eenvoudig toegang tot het ophalen en opslaan van gegevens in de database zonder zich druk te maken om de data laag. Er wordt gebruik gemaakt van het Entity Framework van Microsoft. Het Entity Framework integreert eenvoudig in het Microsoft platform. Er zijn ook andere ORM-oplossing beschikbaar zoals NHibernate. Er moet gebruik worden gemaakt van configuratie bestanden om de mapping te definiëren. Dit kost extra ontwikkeltijd. Er is wel een tool[13] beschikbaar maar deze is niet gratis. EF ondersteund standaard door middel van een GUI om een nieuwe of bestaande database te maken. De mapping wordt hiervoor automatisch gemaakt.

Het Entity Framework biedt een oplossing in drie verschillende scenario's:

1. **Code First**
   Vanuit code is het mogelijk de domein-entiteiten te definiëren en hiervan wordt door het entity framework automatisch een database gegenereerd.
2. **Database first**
   Als er gebruik word gemaakt van een bestaande database of van een ontworpen database is het mogelijk om code te generen met de domein-entiteiten.
3. **Model first**
   Verder biedt het Entity Framework een designer tool aan waarmee het mogelijk is om een database te ontwerpen en vanuit hier wordt een database gegenereerd met de domein-entiteiten.

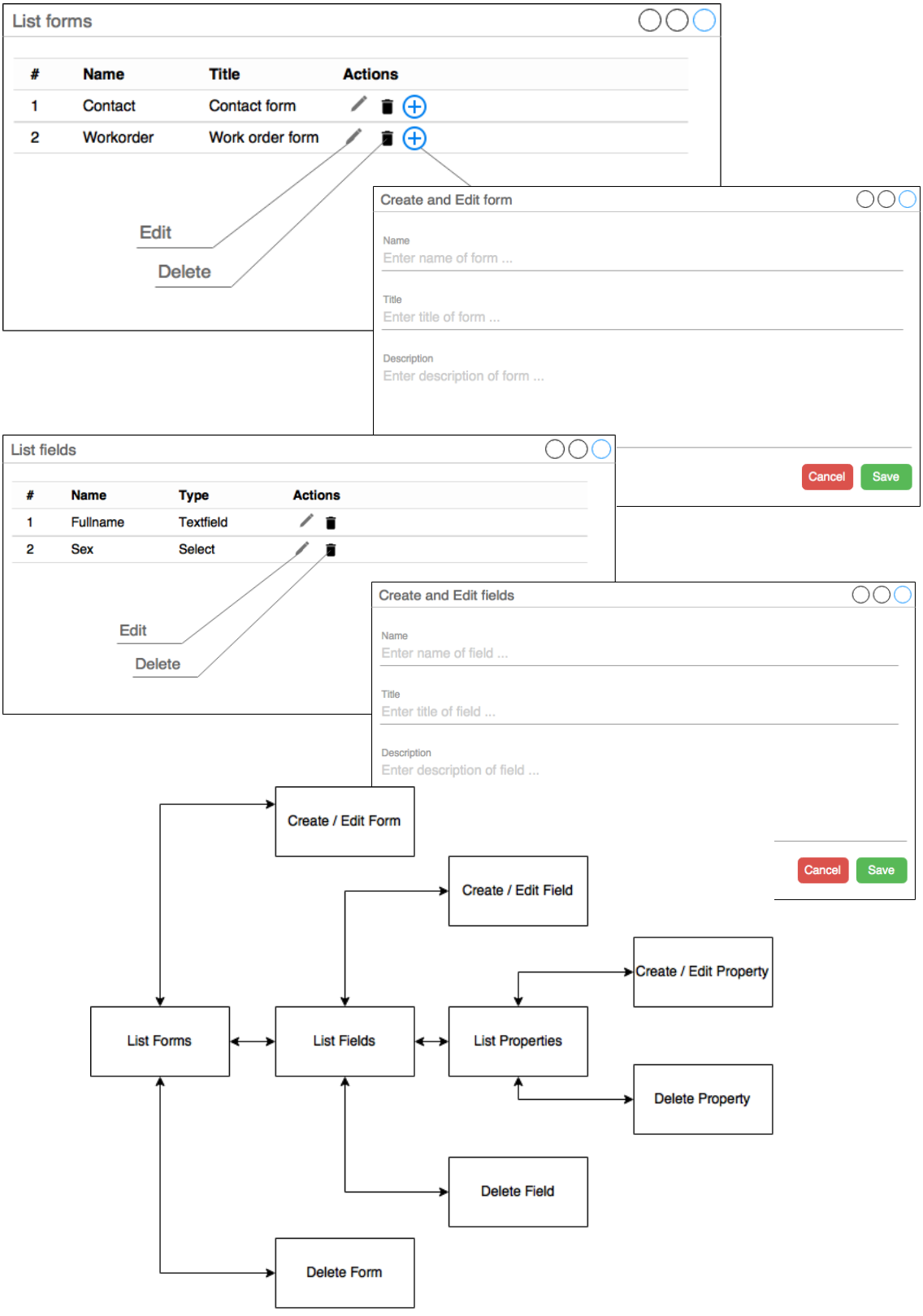
Er is gekozen voor de Database First aanpak aangezien er vooraf een database is ontworpen en gemaakt. Bij Code First wordt de database opgebouwd aan de hand van domein-entiteiten. In tegenstelling tot Code First word de database voor je gegenereerd. Hierdoor bij wijzigingen aan de code wordt er een database script gegenereerd.

Het domein project bevat de implementatie van de domein-entiteiten en het dataproject bevat de database context van het Entity Framework en zorgt voor de afhandeling van CRUD (create, read, update, delete) operaties naar de database toe. Dit betekend dat er geen SQL-query's geschreven hoeft te worden.

[13] https://www.devart.com/

Ontwerpen schermen

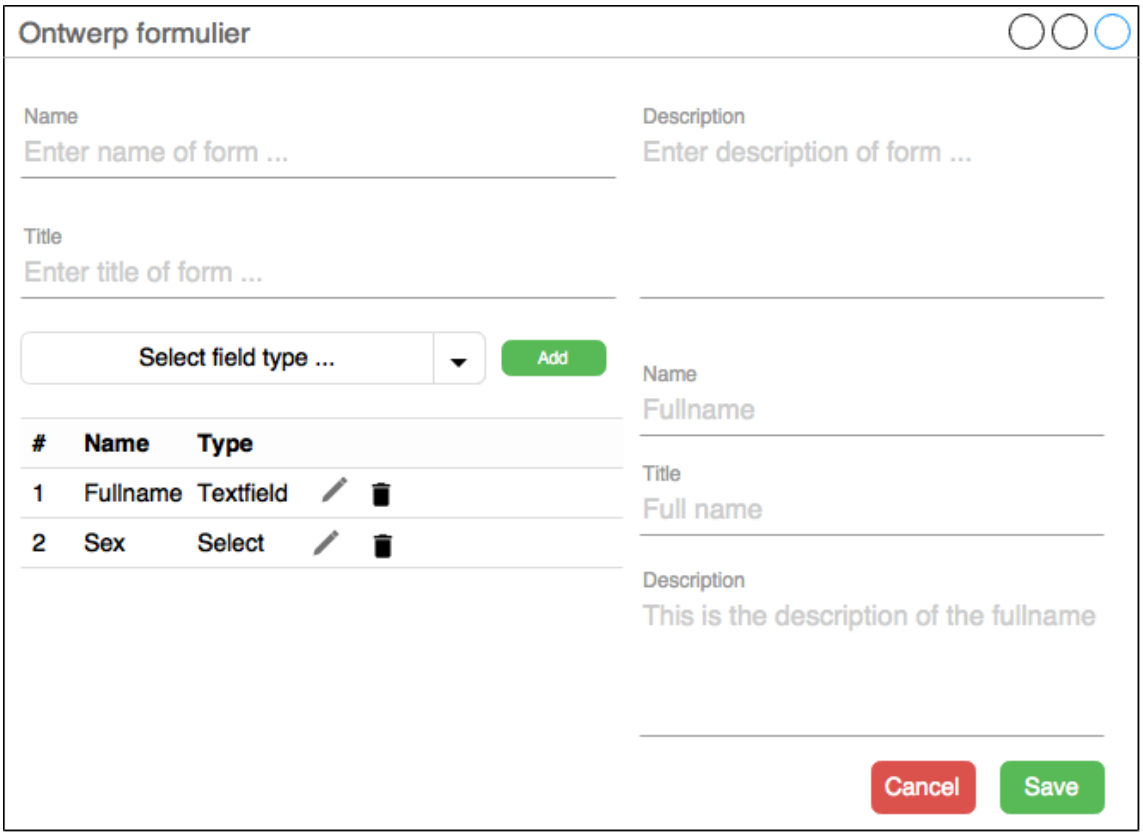
Er is aangegeven dat gebruiksvriendelijkheid een belangrijke eis is voor het systeem. Daarom is er vooraf een mock-up gemaakt. Een mock-up geeft snel een visueel beeld hoe het ontwerp eruit komt te zien. Hierdoor kan snel feedback worden gegeven door de producteigenaar zodat het voldoet aan de eisen. Als dit in een later stadium gebeurd kost dit veel extra tijd om aan te passen.



Na de eerste opzet van de schermen heb ik dit aan de producteigenaar voorgelegd. Tijdens de bespreking van de schermen kwam het volgende naar voren:

- Er zijn te veel verschillende acties nodig zijn om bij het gewenste scherm te komen. Bijvoorbeeld om bij het wijzig scherm te komen van de veldeigenschap moeten er te veel stappen doorlopen worden (List Forms -> List Fields -> List Properties -> Create / Edit Property).
- Bij het wijzigen van een Form (Create / Edit Form) moet er terug naar het overzicht van de formulieren (List Forms) om de velden te kunnen zien (List Fields) van het formulier. De wens is om tijdens het wijzigen van het formulier ook de velden direct te kunnen zien.

Uiteindelijk is een nieuw concept bedacht om alle functionaliteiten van het ontwerpen van een formulier op een scherm te tonen.



Implementatie ontwerp formulier
Doordat nu alle acties op een pagina gebeurd brengt dit wel extra complexiteit met zich mee.

- Het formulier moet dynamisch opgebouwd worden.
- Naast de naam, titel en omschrijving van het formulier moeten nu alle gerelateerde velden opgehaald worden.
- Bij het wijzigen van een veld moeten naast naam, titel en omschrijving ook alle gerelateerde veld eigenschappen opgehaald worden.
- Daarnaast moet het formulier het gehele formulier opslaan met bijbehorende velden en veld eigenschappen.
- Er moet ook rekening worden gehouden met de uitbreidbaarheid van nieuwe type velden.

WebAPI beschikbaar stelen
De WebAPI is een service die op de server staat en de SPA-applicatie staat op de Client. Er moet daarom een connectie vanuit de SPA-applicatie opgezet worden naar de WebAPI.

Binnen de SPA-applicatie moet formulier data kunnen worden toegevoegd, verwijderd, gewijzigd en opgehaald kunnen worden. Dit wordt ook wel afgekort als CRUD (create, read, update en delete) operaties. De SPA-applicatie communiceert via het http-protocol met de WebAPI. Er zijn vier http-methodes die corresponderen met deze CRUD-operaties.

HTTP-methode	CRUD-operatie	Omschrijving
POST	CREATE	Creëren van entiteiten.
READ	READ	Lezen van entiteiten.
PUT	UPDATE	Wijzigen van entiteiten.
DELETE	DELETE	Verwijderen van entiteiten.

Dit betekend dat voor elke entiteit een aparte methode moet worden geschreven die correspondeert met de http-methode. Voor het toevoegen, wijzigen en verwijderen en ophalen van een formulier moeten dus vier aparte methodes worden gemaakt. Voor alle entiteiten betekent dit dus zestien methodes. Dit is natuurlijk niet onderhoudsvriendelijk. Want bij het toevoegen van entiteiten moet er ook apart de attributen worden meegegeven. Bij het ophalen van formulieren moet er ook worden aangegeven welke attributen als resultaat terug moet worden verzonden. Wat nu als er een nieuw attribuut bij de entiteit bijkomt? Er moeten dan verschillende methodes worden aangepast. Daarnaast moet er natuurlijk rekening mee worden gehouden dat een mobiele applicatie andere data verwacht dan de webapplicatie. Er is daarom onderzocht of dit niet op een uniforme manier opgelost kan worden. Na onderzoek bleek hier OData geschrikt voor te zijn.

> **OData**
> OData [14]is een protocol om informatie uit te wisselen op een eenvoudige en gestandaardiseerde manier. Hierbij is niet dat de server bepaald welke informatie gedeeld wordt maar de cliënt. Het probleem was bij ‘klassieke’ Web Services dat er meerdere methodes geïmplementeerd moeten worden met alle mogelijke selecties. OData biedt een uniforme oplossing waardoor het voldoende is om de context van de domein entiteit te ontsluiten naar de cliënt toe. Dit biedt mogelijkheden voor de cliënt om te kunnen sorteren, te kunnen filteren en te selecteren.

Data beschikbaar stellen aan de cliënt
Om de entiteiten (Forms, Fields, FieldTypes en FieldProperties) van de database aan de cliënt beschikbaar te stellen wordt er gebruik gemaakt van BreezeJS[15]. Er is gebruik gemaakt van BreezeJS aangezien deze ondersteuning biedt voor OData. Daarnaast worden alle wijzigingen bijgehouden van de entiteiten. Hierdoor worden alle wijzigingen in 1 keer opgeslagen. Tenslotte het cachen van de data zodat bij offline gebruik de data niet via een API-aanroep moet worden opgehaald. Ook biedt het ondersteuning voor mobiele apparaten.

[14] http://www.odata.org/
[15] http://breeze.github.io/doc-main/

Dynamisch opbouwen formulier
Bij het ontwerpen van een formulier bestaat deze uit een of meerdere verschillende type velden. Elk type veld heeft een andere set van eigenschappen. Bij een selectie box moeten er opties mee gegeven worden en bij een tekstveld niet. De uitdaging is om dit visueel weer te geven in de browser.

Dit kan worden opgelost door gebruik te maken van directives binnen AngularJS. Directives[16] zijn markeringen op een DOM-element (zoals een attribuut, elementnaam, commentaar of CSS-klasse) die ervoor zorgen dat de HTML-compiler van Angular een bepaald gedrag aan dat DOM-element koppelt. Met andere woorden een directive is een herbruikbare module waarin userinterface en gedrag zijn geïmplementeerd.

Field Directive
Er is een field directive gemaakt die het veld mee krijgt vanuit het model. Daarbij wordt de type uitgelezen (select, input) en die een specifieke html template uitleest. In deze html template staat de definitie van het veld. Angular zorgt dat dit wordt gecompileerd en in de DOM word geplaatst.

Als er in de toekomst een nieuw veldtype toegevoegd wordt moet er alleen een nieuwe html pagina aangemaakt worden met de naam van het type en een definitie van de eigenschappen.

Bij het toevoegen van een type aan een formulier wordt er uit een selectie box een keuze gemaakt. Na het selecteren van het veld wordt het model aangepast naar het geselecteerde veld.

Options Directive
Als er gekozen wordt voor een selectie box als veldtype moeten er verschillende opties ingevuld kunnen worden. Hierbij is de uitdaging hoe de gegevens hiervan moeten worden opgeslagen in de database. Het is mogelijk om elke waarde afzonderlijk op te slaan in de fieldproperty tabel maar hoe worden deze gegevens dan weer uitgelezen? Daarom is er besloten om de waarde als komma gescheiden veld op te slaan met als naam options. Op deze manier wordt de waarde options direct meegegeven aan de options directive.

[16] https://nl.wikipedia.org/wiki/AngularJS#Belangrijke_directives

Om dit te realiseren is er een nieuwe directive options ontwikkeld die binnen de options html template kan worden gebruikt. De directive zorgt ervoor dat de property options eerst wordt uitgelezen en als deze niet bestaat wordt aangemaakt. Als de property bestaat wordt de waarde uitgelezen en worden de waardes getoond en het model options gezet. Bij het toevoegen van een nieuwe waarde word de waarde van de property weer als komma gescheiden waarde opgeslagen in het model.

Property Directive

Het moet ook mogelijk zijn om andere specifieke veld eigenschappen op te slaan. Dit moet makkelijk uitbreidbaar zodat het niet noodzakelijk is om bij elk veld eigenschap de tabel aan te passen. Daarom is er een property directive ontwikkeld die ervoor zorgt dat binnen een template automatisch de naam van het veld in het model wordt aangemaakt. Als er bijvoorbeeld als veld eigenschap een ‘placeholder’ wordt meegeven wordt dit als naam van het model gebruikt.

Deze directive kan worden gebruikt als attribuut binnen een bestaand HTML-element. Hiervoor is gekozen omdat het mogelijk maakt om deze aan andere HTML-elementen te koppelen.

Gegevens ophalen aan de hand van gebruikersidentificatie

De formuliergebruiker moet een overzicht krijgen van zijn ontworpen formulieren. Deze gebruiker mag niet de andere formulieren inzien. Nadat de gebruiker is geautoriseerd in het systeem is er binnen het systeem een unieke identificatie van de gebruiker bekend. Deze unieke identificatie wordt gebruikt om de formulieren van de gebruiker op te halen. De unieke identificatie wordt in het model Forms opgeslagen onder het attribuut Owner. Dit voorkomt dat de gebruiker alle formulieren ziet.

Gegevens ophalen aan de hand van een rol

Binnen het systeem wordt er gebruik gemaakt van verschillende rollen. Denk hierbij aan een formulierbeheerder, systeembeheerder of formuliergebruiker. De systeembeheerder moet een overzicht krijgen van alle formulieren en de formuliergebruiker van zijn eigen formulieren.

Er zijn twee opties om dit probleem op te lossen.

1. Een nieuwe tabel maken waarin de rol en de unieke identificatie van de gebruiker wordt opgeslagen. Dit wordt ook wel ‘Role based’ security genoemd.
2. De rol meesturen als claim, waardoor deze met de autorisatie token wordt meegestuurd. Dit wordt ook wel ‘Claim based’ security genoemd.

De eerste optie is niet onderhoudsvriendelijk. Dit brengt extra complexiteit met zich mee aangezien eerst de gebruikers authenticatie moet worden opgehaald en daarna een query op de database moet worden uitgevoerd om de gebruikersrol te bepalen.

Er is gekozen voor de tweede optie aangezien dit een generieke manier is om extra gegevens met geautoriseerde gebruikers mee te sturen. Dit brengt geen extra complexiteit met zich mee. Daarnaast ligt de verantwoordelijkheid bij de Identity Hub en niet binnen het eigen systeem. Binnen de Identity Hub kan er een extra claim worden meegenomen door gebruik te maken van metadata en kan worden meegenomen met de autorisatie token.

6.3. Testen

Functionaliteit
Voor het controleren van het kenmerk kwaliteitsattribuut functionaliteit zijn er gedetailleerde test cases (TC7 t/m TC27) opgezet . Onderstaand een voorbeeld van de gedetailleerde testcases. Voor een volledige lijst zie Testrapport in Externe bijlagen

ID #	Beschrijving	Sprint	Use case
TC7	Toevoegen formulier zonder alle velden ingevuld	2	UC1
TC25	Formulier wordt getoond in de lijst na toevoegen van formulier	2	UC3

ID#	Actie	Verwacht resultaat	Werkelijk resultaat
TC7	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Klik op de button ‘Opslaan’	Foutmelding formulier kan niet worden toegevoegd	X
TC25	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het ‘input’ veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Na toevoegen van formulier wordt de naam van het formulier getoond in de lijst.	V

Gebruiksvriendelijkheid
Ik heb een willekeurige gebruiker gevraagd om het systeem op gebruiksvriendelijkheid te testen. Daarbij was de opdracht om zonder enige kennis een formulier te ontwerpen.

Uit deze test is het volgende gekomen:

- De selectie van de velden is onduidelijk:
  - Wat wordt er bedoeld met 'input' => tekstveld?
- Bij het verkleinen van het scherm bleek het toevoegen van een veld te verdwijnen.
- Een tooltip bij de velden bij het wijzigen en verwijderen van een veld. Er is niet duidelijk wat hiermee bedoeld wordt.
- Er wordt een foutmelding rechtsonder getoond maar deze verdwijnt te snel.

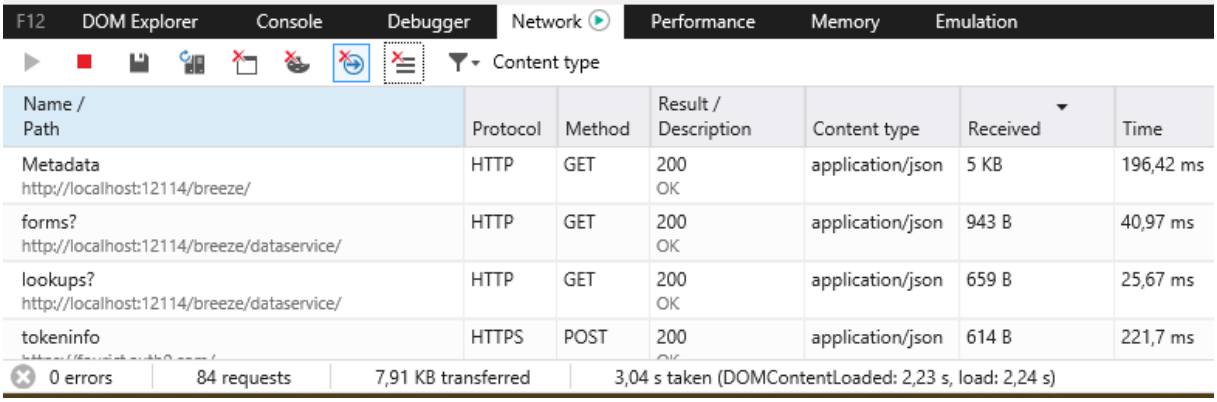
Bovenstaande testresultaten zijn opgenomen in de productbacklog.

Portabiliteit
De verschillende gedetailleerde testcases zijn uitgevoerd op de drie meest gebruikte browsers volgens StatCounter[17] (Google Chrome, Internet Explorer en FireFox) om de portabiliteit te testen.

Er zijn geen problemen gevonden onder de verschillende browsers.

Continuïteit
Het systeem is nog niet in productie waardoor de continuïteit niet getest kan worden. De continuïteit wordt bij een systeemtest uitgevoerd.

Performance
Er is op exploratieve wijze getest door te controleren via de browser hoe snel de pagina laadt. Uit de test is gekomen dat bij het opnieuw laden van de gehele SPA-applicatie het laden 3,04s duurde.



Als de SPA-applicatie geladen was duurde het laden van alleen de pagina 3,04 s.
Er is aangegeven dat de gehele SPA-applicatie binnen een seconde gestart moet zijn. Er is daarom een item in de productbacklog geplaatst om de SPA-applicatie te optimaliseren.

[17] http://gs.statcounter.com/#browser-ww-monthly-201502-201602

6.4. Sprint review

Aangezien 25 december 2015 op eerste kerstdag is hebben wij de sprint review op 24 december 2015 gezet. Tijdens de sprint review is er gekeken naar de volgende aspecten:

- Het authenticatie proces;
- Overzicht tonen van formulier(en);
- Het ontwerpen van het formulier;
- Het instellen van validatieregels van het formulier.

Bij het authenticatie proces is het aanmeldproces gedemonstreerd waarbij iedereen zich als formulierbeheerder kan aanmelden ook al is de gebruiker niet geregistreerd. Daarnaast zijn verschillende overzichten getoond op basis van de aanmelding. Voor het ontwerpen van een formulier wordt er een keuze gegeven van verschillende standaard type velden met daarbij hun eigen parameters. Er is ook gekeken naar de uitbreidbaarheid met nieuwe veldtypes en parameters.

De demonstratie verliep goed afgezien van wat kleine tekstuele aanpassingen waarbij “Forms” in de navigatie menu gewijzigd moet worden naar “My Forms”. Daarnaast moet in het overzicht van de formulieren de titel gewijzigd worden naar “Forms I created”. Verder kwam eruit dat er een mogelijkheid moet zijn om een preview te zien van het ontworpen formulier. De product backlog is bijgewerkt met de nieuwe mogelijkheid om een preview te krijgen maar is geen onderdeel van het afstudeerproject.

6.5. Retrospective

Wat ging goed?
- Bug authenticatie direct meegenomen van de vorige sprint aangezien ik hier last kreeg bij het testen.
- De user story dat iedereen zich kan aanmelden als formulierbeheerder.
- Ondanks feestdag toch op tijd een demonstratie kunnen geven.
- Goede feedback van de producteigenaar en stakeholder.

Wat ging niet goed?
- Planning aangezien er een dag minder was.
- Ik had de bug en feature niet mee mogen nemen omdat deze niet ingeplant stond.

Welke acties moeten er komen om dit te voorkomen?
- De afhankelijkheden van de verschillende user stories beter in de gaten houden.

7. Sprint 3: “Toekennen gebruikers en afhandelen"

7.1. Planning

Op 24 december 2015 is er een sprint planning geweest met de producteigenaar en stakeholder waarbij gekeken is naar de globale planning, zie forecast in hoofdstuk sprint 0. Uit de product backlog is een selectie gemaakt op basis van 10 storypoints. Daarbij zijn wij door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld.

#PBI	#R	#UC	Titel	Effort
324	8	UC6	Als geregistreerde formulierbeheerder wil ik een formuliergebruiker toekennen aan mijn ontworpen formulier, zodat mijn medewerkers deze kunnen inzien.	5
325	7	-	Als geregistreerd formulierbeheerder wil ik het resultaat als e-mail ontvangen, zodat ik het resultaat via e-mail ontvangt.	3
326	11	UC2	Als geregistreerde formulierbeheerder wil ik een formulier kunnen wijzigen, zodat ik het formulier kan beheren.	2

Risicoklasse

Na het opzetten van de use case word de risicoklasse ingeschat. De risicoklasse bepaald uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICO-KLASSE	ZWAARTE
UC2	**Functionaliteit**	M	L	B	●●
UC6	**Functionaliteit**	L	M	C	●

De testen zijn uitgewerkt in hoofdstuk 7.3

De sprint goal is:

- Het toekennen van gebruikers aan een formulier en het afhandelen hiervan.

7.2. Uitvoering

Toekennen gebruikers
Voor het toekennen van gebruikers van de applicatie is de uitdaging geweest om te kijken op welke manier de relatie gelegd moet worden tussen het formulier en de gebruiker. Er zijn hiervoor twee oplossingen:

1. Aanmaken van een tabel waarin de id van het formulier en het id van de gebruiker wordt opgeslagen;
2. Opslaan in de metadata van de gebruiker binnen de authenticatie provider. Door gebruik te maken van een claim.

Uiteindelijk is gekozen om voor optie twee te gaan. Dit is gekozen omdat de gebruiker binnen de authenticatie provider verwijderd kan worden. Hierdoor blijft deze relatie bestaan in de tabel. Nu kan er bij het toekennen van een gebruiker in dezelfde aanvraag ook de metadata gezet worden.

Het toekennen van een gebruiker is in het sequence diagram weergegeven, zie Bijlage 5: Toekenning gebruiker.

1. Bij de aanvraag naar de API vanuit de browser wordt er eerst gevalideerd aan de hand van de JWT-autorisatie token. Dit is niet opgenomen in de sequence diagram aangezien dit voor elke aanvraag terugkomt;

2. De Web API verstuurt de aanvraag naar de Identity Hub waarbij gecontroleerd word of het mail adres van de gebruiker bestaat en retourneert hiervan het resultaat terug naar de API met de gebruikersgegevens zoals de metadata.

3. Als de gebruiker bestaat wordt er een controle uitgevoerd of de gebruiker die toegekend wordt niet de formulier beheerder is. Daarnaast wordt er een controle uitgevoerd of de gebruiker niet al is toegekend aan het formulier. De metadata van de gebruiker wordt uiteindelijk aangepast met het formulier en wordt verstuurd. Uiteindelijk word het resultaat teruggestuurd naar de browser.

4. Als de gebruiker niet bestaat wordt de gebruiker toegevoegd in Auth0 waarbij ook de metadata wordt meegegeven. Daarnaast wordt er ook een verificatie mail door de Identity Hub aangemaakt zodat de gebruiker eerst zijn e-mailadres moet verifiëren voordat gebruik kan worden gemaakt van de applicatie.

De metadata van een gebruiker wordt als een JSON Object opgeslagen waarbij het unieke nummer van het formulier in een array met de naam “forms” wordt opgeslagen.

Aanpassing architectuur
Voor het versturen van een e-mail met als resultaat de waardes van het formulier naar de gebruiker zijn er meerdere oplossingen mogelijk.

Er kan o.a. gebruik worden gemaakt van:

1. Een eigen Mail Server om de mail te versturen;
2. Een SaaS-oplossing om de mail te versturen;
3. Een ESB-oplossing om de mail te versturen;

Aangezien de webapplicatie en de mobiele applicatie op de cliënt staat en niet op de server moet bij de eerste oplossing een API beschikbaar worden gesteld. Het is niet mogelijk om vanuit de cliënt een mail te sturen. De API staat op een server en kan connectie maken met de mailserver om een e-mail te versturen. Vanuit de cliënt is het mogelijk om een aanvraag te doen naar de API zodat de e-mail verstuurd kan worden. Het voordeel hiervan is het eigen beheer van de mailserver. Bij wijziging van mail provider is deze makkelijk aanpasbaar op de server en is het niet noodzakelijk om de cliënts aan te passen. Het nadeel is het onderhoud, configureren en beheren van de mailserver. Verder is er geen garantie dat de mail is verstuurd. Alles moet zelf afgehandeld worden.

In tegenstelling tot het gebruik van een SaaS (Software-as-a-Service) oplossing zoals MailChimp[18] is er wel een garantie dat de mail verstuurd word en worden statistieken bijgehouden. Het nadeel is de afhankelijkheid aangezien de cliënts aangepast moeten worden naar de API van de afnemer van de dienst. Anderzijds is dit op te lossen door het te versturen naar een eigen API maar dit kan betekenen dat er op twee plekken iets fout kan gaan. Uiteraard is het mogelijk om dit allemaal via de cliënt op te lossen door het resultaat voor de zekerheid lokaal op te slaan maar dit vraagt allemaal extra aanpassingen. Daarnaast is er nog een belangrijkere vraag dat als de klant het niet via de mail wilt hebben maar via een andere oplossing zoals zijn backoffice. Wat als er besloten wordt binnen FourICT om het resultaat van de formulieren in de database op te slaan?

Een ESB-oplossing kan hier uitkomst in bieden. Een ESB oftewel een Enterprise Service Bus[19] is een design pattern waarmee de communicatie tussen de afnemers van diensten en aanbieders hiervan, vereenvoudigd wordt.

Het grote voordeel van het gebruik van een ESB-oplossing is het compleet loskoppelen (loosely coupled) van serviceaanbieders en serviceaanvragers. Daarnaast het versimpelen en standaardiseren van interfaces tussen de aanbieders en aanvragers. Verder kunnen op een centrale en generieke manier gebruik worden gemaakt van monitoring. Ook is het mogelijk om snel in te springen op veranderingen. Het nadeel is dat dit wel extra complexiteit met zich meebrengt in de oplossing.

Om tot een generieke oplossing te komen voor toekomstige ontwikkelingen is er daarom besloten om de architectuur aan te passen en gebruik te maken te maken van een Enterprise Service Bus (ESB) architectuur.

[18] Dienst die via internet beschikbaar is om e-mail berichten te versturen.
[19] https://nl.wikipedia.org/wiki/Enterprise_service_bus

Keuze middleware product
Er zijn verschillende middleware producten beschikbaar voor het implementeren van een ESB-oplossing. Er wordt binnen FourICT gebruik gemaakt van Microsoft BizTalk en nServiceBus. Bij de keuze is rekening gehouden met verschillende criteria waarbij gekeken is naar de functionaliteit, kosten, integratie, ondersteuning, onderhoudbaarheid, gebruiksvriendelijkheid. De criteria is in samenspraak met de producteigenaar vastgesteld. Zie Bijlage 3: Vergelijking BizTalk en nServiceBus voor de uitwerking.

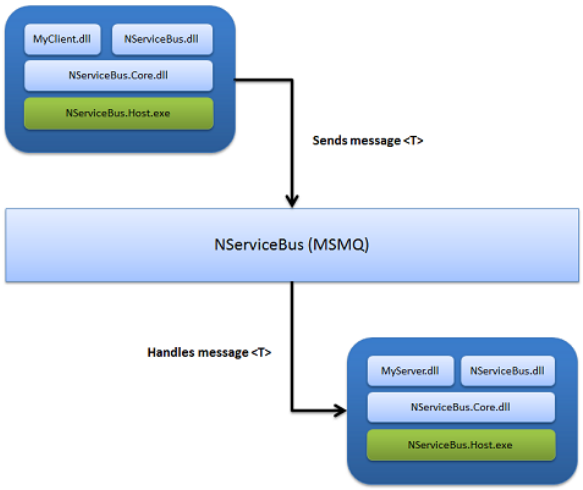
nServiceBus
De keuze is bepaald door het budget. Daarnaast biedt nServiceBus voldoende functionaliteit om loskgekoppelde applicaties te ontwikkelen. BizTalk daarentegen is een complete server oplossing die te uitgebreide functionaliteit bevat die niet nodig is.

nServiceBus[20] is een open source .NET Framework gebaseerd op het ESB (Enterprise Service Bus) pattern om op eenvoudige manier losgekoppelde applicaties te ontwikkelen die betrouwbaar, schaalbaar en uitbreidbaar zijn.

Het biedt de volgende eigenschappen:

- Hoge prestaties.
- Makkelijk schaalbaar.
- Betrouwbaar door middel van een automatisch mechanisme om het nog eens te proberen als er iets fout gaat.
- Loosely-coupled door pub / sub principe.
- Eenvoudig uitbreidbaar.
- Biedt verschillende mogelijkheden om berichten te transporteren via MSMQ, RabbitMQ, SQL Service, Windows Azure Queues, etc.

Globaal is dit op onderstaande tekening weergegeven.



1. Een cliënt (webapplicatie, mobiele applicatie) verstuurt een bepaald type bericht op de service bus.

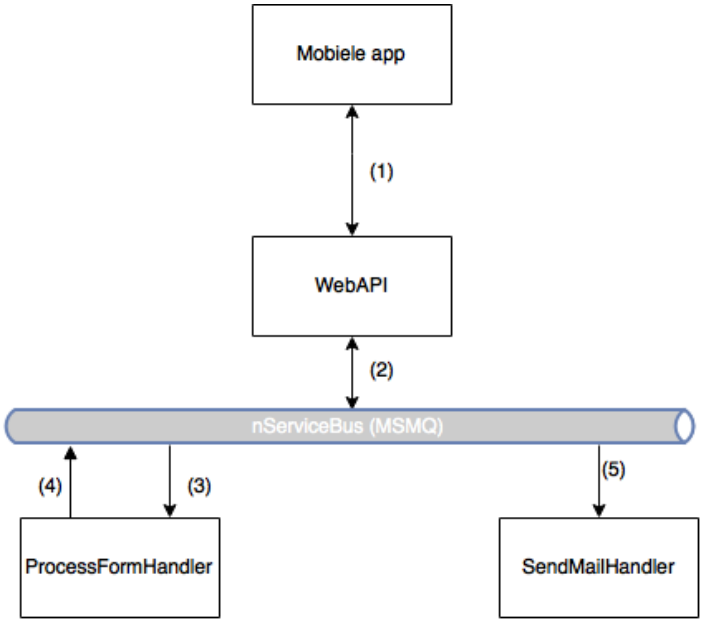
2. Het bericht wordt door nServiceBus opgeslagen in de messagequeue (MSMQ).

3. Een andere cliënt pakt dit bericht op en gaat met het type bericht aan de slag. Dit wordt ook wel een handler genoemd;

Figuur 7.1 Architectuur nServiceBus

[20] http://particular.NET/nServiceBus

Afhandeling formulier gegevens
Voor het afhandelen van de formulier gegevens wordt er gebruik gemaakt van een ESB-oplossing. In onderstaande afbeelding wordt er schematisch weergegeven hoe de afhandeling van de formulier gegevens wordt verwerkt.



Figuur 7.2 Afhandeling formulier gegevens met nServiceBus

1. De mobiele app verstuurt de formulier gegevens naar de Web API;

2. De Web API maakt een bericht ‘ProcessForm’ met de authenticatie gegevens en de formulier gegevens en stuurt het bericht naar nServiceBus die het in de MSMQ plaatst; De mobiele applicatie krijgt direct een response terug dat het bericht is verstuurd naar de nServiceBus.

3. De nServiceBus handler ‘ProcessFormHandler’ pakt het berichttype ‘ProcessForm’ op en zorgt voor de afhandeling van de formulier gegevens.

4. Er wordt een nieuw bericht aangemaakt ‘FormProcessed’ en verstuurd naar nServiceBus die het in de MSMQ plaatst;

5. De nServiceBus handler ‘SendMailHandler’ pakt het berichttype ‘FormProcessed’ op en verstuurt de mail naar de formulierbeheerder.

Elke handler heeft een eigen verantwoordelijkheid waarbij er geen afhankelijkheden zijn. Dit maakt het eenvoudig om een nieuwe handler te ontwikkelen die ervoor zorgt dat het bericht in een extern systeem geïmplementeerd kan worden. Dit maakt het zeer eenvoudig om de implementatie van de SendMailHandler te vervangen door een ander.

Wanneer de mailserver niet bereikbaar is of er een exceptie optreedt in de handler, dan zorgt nServiceBus voor het afhandelen van deze fouten. Er zit een retry mechanisme in waardoor er beslist kan worden hoe hier mee om te gaan. Het bericht kan getraceerd worden waardoor je gegarandeerd bent dat het bericht is aangekomen.

7.3. Testen

Functionaliteit
Er is exploratief getest voor het wijzigen van het formulier (UC2). Bij het exploratief testen is het wijzigen van het formulier getest in de browser.

Hierbij zijn geen fouten ontdekt.

Daarnaast zijn er globale testcases opgezet voor het toekennen van gebruikers aan een formulier (UC6). Voor een volledige lijst zie Testrapport in Externe bijlagen.

Hierbij zijn geen fouten ontdekt.

7.4. Sprint review

Op vrijdag 15 Januari is er een sprint review geweest over de derde sprint. Hierbij zijn de volgende aspecten gedemonstreerd:

- Het toekennen van gebruikers aan een formulier.
- De architectuuraanpassing en de opzet van het afhandelen van formulier gegevens.

Helaas bleek tijdens de demonstratie dat het toekennen van gebruikers aan een formulier niet goed ging. Na onderzoek bleek dit te komen doordat de autorisatie token was verlopen. Na het opnieuw aanmelden bleek het weer te werken. Daarnaast voor het afhandelen van de mail is het concept van de architectuuraanpassing gedemonstreerd. Aangezien het formulier nog niet wordt opgestuurd omdat dit een onderdeel van de mobiele applicatie was dit niet demonstreerbaar. Aangezien er een afhankelijkheid is tussen de webapplicatie en de mobiele applicatie, is er een extra taak bijgevoegd dat bij de implementatie van de user story R18 dit gedemonstreerd wordt.

Uit de sprint review sessie waren de volgende onderdelen gekomen:

- Aanpassen van “Assign” naar “Assign users to form”.
- Mogelijkheid om niet alleen specifiek voor een formulier een gebruiker toe te kennen maar via een overzicht van alle gebruikers met daarbij alle formulier waar ze toegang toe hebben te tonen.

Dit is opgenomen als extra user story in de product backlog.

7.5. Retrospective

Wat ging goed?
- Aanpassing gemaakt aan de architectuur voor het verzenden van de mail. Dit voldoet nu aan de niet-functionele requirement om in de toekomst te voldoen aan andere mogelijkheden dan mail.

Wat ging niet goed?
- De demonstratie. Het was lastig om de architectuur aanpassing te demonsteren. Dit kwam doordat de user story voor het versturen van het formulier geen onderdeel was van deze sprint.

Welke acties moeten er komen om dit te voorkomen?
- De user stories in de sprint planning beter scheiden van afhankelijkheid.

8. Sprint 4: “Webapplicatie afronden”

8.1. Planning

Op 15 januari 2016 is er een sprint planning geweest na de sprint review en is met de producteigenaar en stakeholder gekeken naar de product backlog. Door de producteigenaar is besloten om de webapplicatie eerst af te ronden alvorens door te gaan met de mobiele applicatie en daardoor is de prioriteit gewijzigd van de #PBI 331, 332 en 329. Daarbij zijn wij door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld. Daarnaast zijn er wat bugs opgepakt die in

#PBI	#R	#UC	Titel	Effort
331	12	UC4	Als geregistreerde formulierbeheerder wil ik een formulier kunnen verwijderen, zodat ik formulier kan beheren.	2
332	10	UC9	Als geregistreerde formulierbeheerder wil ik mij af kunnen melden, zodat ik niet meer ben aangemeld.	2
329	13	UC7	Als geregistreerde formulierbeheerder wil ik formuliergebruikers verwijderen, zodat ik formuliergebruikers kan beheren.	3

Risicoklasse

Na het opzetten van de use case wordt de risicoklasse ingeschat. De risicoklasse bepaalt uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICO-KLASSE	ZWAARTE
UC4	**Functionaliteit**	L	L	C	●
UC7	**Functionaliteit**	L	H	B	●●
UC9	**Functionaliteit**	L	L	C	●

De testen zijn uitgewerkt in hoofdstuk 8.3

De sprint goal is:

- Het afronden van de webapplicatie.

8.2. Uitvoering

Verwijderen formulier
Bij het verwijderen moet er rekening mee worden gehouden met twee dingen.

1. Het formulier moet uit de metadata van de gebruikers verwijderd worden.
2. Het formulier moet inclusief de gerelateerde velden met veldeigenschappen uit de database verwijderd worden.

Afmelden gebruiker
Voor het afmelden van de gebruiker is de autorisatie service uitgebreid zodat de gebruiker zich kan afmelden van het systeem. Ten eerste moet bij het afmelden eerst de autorisatie token worden verwijderd uit de lokale opslag. De autorisatie token wordt automatisch meegestuurd aan elke aanvraag naar de API. De API weet wie je bent en of je geautoriseerd bent. Ten tweede moet de profiel informatie die aan de gebruiker is gekoppeld verwijderd worden uit de lokale opslag.

Verwijderen toekenning gebruiker
Voor het verwijderen van een toekenning van een gebruiker is het noodzakelijk om de metadata aan te passen.

Het UML sequence diagram is uitgewerkt in Bijlage 6: Verwijderen toekenning gebruiker.

1. Bij de aanvraag naar de API vanuit de browser wordt er eerst gevalideerd aan de hand van de JWT-autorisatie token. Zie hoofdstuk 0. Dit is niet opgenomen in de SD aangezien dit voor elke aanvraag terugkomt;

2. De Web API verstuurt de aanvraag naar de Identity Hub waarbij gecontroleerd wordt of gebruiker bestaat en retourneert hiervan het resultaat terug naar de API met de gebruikersgegevens zoals de metadata.

3. Eerst worden alle formulieren opgehaald uit de metadata en wordt er gecontroleerd of het formulier toegekend is aan de gebruiker. Als het formulier niet gevonden wordt in de metadata van de gebruiker wordt er een foutmelding weergegeven.

4. Als het formulier is toegekend aan de gebruiker wordt het formulier uit de metadata verwijderd. De aanvraag wordt daarna doorgestuurd aan Auth0 waarbij de metadata wordt opgeslagen. Uiteindelijk wordt een melding teruggestuurd naar de cliënt met de nieuwe gebruikersdata.

8.3. Testen

Functionaliteit
Het verwijderen van het formulier (UC4) is exploratief getest. Hierbij is in de browser de test uitgevoerd waarbij de functionaliteit is getest. Daarnaast getest onder andere als systeembeheerder gecontroleerd of het formulier mag worden verwijderd. Er is ook gecontroleerd of een formulier door een andere gebruiker verwijderd mag worden.

> *Hierbij is een fout ontdekt en direct opgelost waardoor als systeembeheerder het formulier niet verwijderd kon worden.*

Voor het testen van het verwijderen van een formuliergebruiker (UC7) zijn globale testcases opgesteld. Voor een volledige lijst zie Testrapport in Externe bijlagen.

Er is 1 fout geconstateerd dat de gebruiker niet door een systeembeheerder kan worden verwijderd. Dit is opgelost.

Het afmelden van een formuliergebruiker (UC9) is exploratief getest. Hierbij is in de browser de test uitgevoerd waarbij de functionaliteit is getest. Er is gecontroleerd of de gebruiker is afgemeld en/of de token is verwijderd uit de lokale opslag.

> *Hierbij zijn geen fouten ontdekt.*

8.4. Sprint review
Op vrijdag 29 Januari 2016 is er een sprint review geweest over de vierde sprint. Hierbij zijn de volgende aspecten gedemonstreerd:

- Het verwijderen van een formulier.
- Het afmelden van de gebruiker.
- Het verwijderen van een toegekend formulier aan een gebruiker.

De demonstratie verliep zonder enige problemen en het afronden van de webapplicatie in de eerste fase was een feit. De stakeholder en producteigenaar waren erg tevreden.

Uit de sprint review sessie waren de volgende onderdelen gekomen:

- Mogelijkheid om niet fysiek het formulier te verwijderen uit de database zodat de wijziging teruggedraaid kan worden.
- Confirmatie alvorens het formulier daadwerkelijk wordt verwijderd.

Deze punten zijn opgenomen in de product backlog.

8.5. Retrospective

Wat ging goed?
- Demonstratie verliep zonder problemen.
- De stakeholder en producteigenaar waren tevreden met het snelle resultaat.

9. Sprint 5: “Mobiele applicatie”

9.1. Planning

Op 29 januari 2016 is er een sprint planning geweest na de sprint review en is met de producteigenaar en stakeholder gekeken naar de product backlog. Aangezien de architectuur opzet niet getest kon worden fysiek op een iOS is ervoor gekozen om deze weer op te nemen. Uit de product backlog is een selectie gemaakt op basis van 10 storypoints. Daarbij zijn wij door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld.

#PBI	#R	#UC	Titel	Effort
343	14	-	Als formuliergebruiker wil ik op mijn mobiele apparaat mijn formulieren inzien, zodat als ik onderweg mijn formulieren kan bekijken en versturen.	5
344	15	UC8	Als formuliergebruiker wil ik met mijn mobiele apparaat kunnen aanmelden, zodat ik toegang krijg tot het systeem.	3
346	16	UC3	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat een overzicht van mijn formulieren, zodat ik een formulier kan selecteren.	2

Risicoklasse

Na het opzetten van de use case wordt de risicoklasse ingeschat. De risicoklasse bepaalt uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICO-KLASSE	ZWAARTE
UC3	**Functionaliteit**	H	L	B	●●
	Performance	L	L	C	●
	Continuïteit	H	L	B	●●
	Gebruiksvriendelijkheid	M	L	C	●
	Portabiliteit	H	L	B	●●
UC8	**Functionaliteit**	H	L	B	●●
	Beveiliging	H	L	B	●●

De testen zijn uitgewerkt in hoofdstuk 9.3

De sprint goal is:

- Het aanmelden en selecteren van het formulier via de mobiele applicatie.

9.2. Uitvoering

Opzet architectuur mobiele applicatie

Er zijn vele verschillende mogelijkheden om een mobiele applicatie te ontwikkelen. Deze zijn onder te verdelen in drie verschillende types:

1. Web app
2. Native app
3. Hybride app

De voor- en nadelen zijn schematisch weergegeven in Bijlage 1: Vergelijking webapp, native app en hybride app.

Er is gekozen voor een hybride app als mobiele applicatie aangezien er gebruik kan worden gemaakt van beschikbare functionaliteiten zoals de camera, scannen van een barcode, lokale opslag van het apparaat. Aangezien via het formulier in de toekomst deze functionaliteiten moet bevatten is hier al op geanticipeerd. De hybride app kan ook functioneren zonder dat er internetverbinding noodzakelijk is. Verder is er geen kennis nodig van specifieke gebruikte talen van het ontwikkelplatform omdat er gebruik wordt gemaakt van bestaande web technologieën.

Omdat het gebaseerd is op bestaande web technologieën betekent dit ook dat de applicatie vrij generiek kan worden gehouden en daardoor onderhoudsvriendelijk. Het is daarom niet noodzakelijk om voor elk platform (iOS, Android en Windows Phone) broncode te schrijven.

De hybride app is daarnaast ontwikkeld voor alleen het weergeven van het formulier en niet voor het ontwerpen van het formulier. Deze keuze is gemaakt aangezien het ontwerpen van het formulier veel handelingen vereist op een smartphone en/of tablet. Het formulier wordt maar een keer ontworpen en daarom zal het niet vaak voorkomen dat het continu wordt aangepast.

Aangezien er rekening mee moet worden met de niet-functionele eis dat het op verschillende mobiele apparaten en op verschillende platformen draait is dat hiermee afgedekt.

Keuze hybride platform

Voor het ontwikkelen van hybride applicaties zijn er verschillende Frameworks beschikbaar. Er is onderzoek gedaan naar Ionic[21] en Framework 7[22]. Beide Frameworks bieden de mogelijkheid om hybride apps te ontwikkelen met HTML5, CSS en JS. Er is gekozen voor Ionic aangezien dit Framework gebaseerd is op Angular in tegenstelling tot Framework 7. Hierdoor kan dezelfde opzet als de webapplicatie gebruikt worden waardoor er sneller ontwikkeld kan worden. Het Ionic Framework. Ionic is een UI framework dat is geoptimaliseerd voor mobiele apparaten om met behulp van HTML5, CSS en Javascript hybride mobiele applicaties te ontwikkelen. Dit is een goede keuze aangezien het Ionic framework gebaseerd is op AngularJS. Hierdoor is de leercurve laag om gebruik te maken van het framework. Verder kan dezelfde project opzet voor de webapplicatie en mobiele applicatie gebruikt worden. Daarnaast biedt het extra ondersteuning door middel van een

[21] http://ionicframework.com/
[22] http://framework7.io/

CLI-tool (Command Line Interface) om snel aan de slag te gaan. Verder is het gebaseerd op open source en is er een grote community. Ook biedt het door middel van Cordova[23] om specifieke onderdelen van het mobiele apparaat zoals de camera, gps te kunnen raadplegen op verschillende mobiele platformen zoals Android en IOS. Cordova zorgt ervoor dat de mobiele applicatie gereed wordt gemaakt om te gebruiken als native app.

Wijziging routing
Bij het implementeren van dezelfde opzet als de webapplicatie voor het gebruik met Ionic bleek de routing niet te werken. De routing zorgt ervoor dat de juiste controller en view wordt geladen. Ionic baseerde de routing op de ui-router en de webapplicatie op de ngRoute implementatie. Na wat onderzoek[24] bleek het grote verschil te zitten dat het een oplossing bied voor geneste views. Daarnaast zal de wijziging in de URL geen effect hebben dat links binnen de applicatie moeten worden aangepast. Om de routing te laten werken binnen Ionic is ervoor gekozen om de bestaande implementatie te wijzigen. Aangezien bij het ontwikkelen hier al rekening mee is gehouden door een helper implementatie te maken van de routing functionaliteit bij de webapplicatie heb ik dit kunnen wijzigen (loosely coupled) naar de nieuwe implementatie.

Aanmelden gebruiker
Het autorisatie proces is al eerder opgezet door middel van een authenticatie service binnen de webapplicatie. Aangezien Ionic gebaseerd is op AngularJS kan dezelfde implementatie worden gebruikt.

Service ontwikkelen
Service ontwikkelen In de vorige Sprint is er naar voren gekomen dat er gebruik wordt gemaakt van een WebAPI. Een WebAPI zorgt voor ervoor dat de SPA-applicatie van data wordt voorzien vanuit een database. Aangezien Ionic gebaseerd is op AngularJS kan dezelfde implementatie gebruikt worden. De enige aanpassing is in de gebruikersinterface waarbij gebruik wordt gemaakt van de styling van Ionic. Door het gebruik van deze styling wordt de lijst van formulieren correct weergegeven op de verschillende platformen.

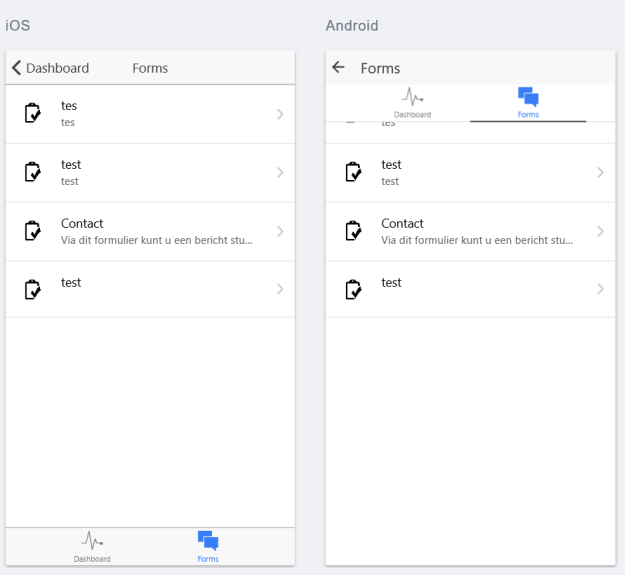
[23] https://cordova.apache.org/
[24] http://findnerd.com/list/view/AngularJS-difference-between-routeprovider-and-stateprovider/8338/

9.3. Testen

Functionaliteit
Voor het aanmelden van de gebruiker (UC8) en het tonen van een overzicht (UC3) zijn globale testcases opgesteld. Hierbij zijn de globale testcases (TC1 t/m TC6) uitgevoerd die zijn opgezet in Sprint 1 op de mobiele applicatie uitgevoerd. Daarnaast voor het tonen van het formulier zijn de globale testcases (TC25 t/m TC27) uitgevoerd. Voor een volledige lijst zie Testrapport in Externe bijlagen

Er zijn geen fouten gevonden

Gebruiksvriendelijkheid
Er is door een willekeurige gebruiker een usability test uitgevoerd. Het was niet mogelijk om dit op het fysieke mobiele apparaat te testen aangezien de API publiekelijk beschikbaar moet zijn. Er is daarom gekozen om dit via de simulator te doen van Android en iOS. Er is door de opdrachtgever aangegeven dat het moet draaien op een iOS en Android. Daarnaast is het belangrijk om deze test nogmaals uit te voeren op een fysiek mobiel apparaat. Dit is belangrijk aangezien dit een andere gebruikerservaring heeft. Denk bijvoorbeeld aan de navigatie met de vingers in plaats van de muis.



Uit de test zijn de volgende resultaten gekomen:

- Tabblad wordt op iOS onderaan getoond en op Android bovenaan.
- Het icoon bij Forms is niet duidelijk dat het om formulieren gaat.
- Het icoon naast de naam van het formulier is ook niet duidelijk.
- Er kan niet worden gezocht op formulieren.
- Bij het aanmelden miste een icoon.

Deze punten zijn opgenomen in de product backlog.

Performance
De performance van de mobiele applicatie kan niet goed getest worden. Dit moet gebeuren op het fysieke apparaat. De performance van de mobiele applicatie via de simulator verliep goed.

Continuïteit
De continuïteit kan niet getest worden aangezien dit pas kan bij het publiceren van de mobiele applicatie in de store.

Portabiliteit
Uiteraard is het niet mogelijk om de mobiele applicatie op alle apparaten te testen. Er is aangegeven door de opdrachtgever dat het moet werken op iOS en Android. Daarbij zijn verschillende simulators gebruikt op een tablet en smartphone. De portabiliteit van de mobiele applicatie is uitgevoerd in een simulator op verschillende apparaten.

9.4. Sprint review

Op vrijdag 12 Februari 2016 is er een sprint review geweest over de vijfde sprint.

Hierbij zijn de volgende aspecten gedemonstreerd:

- Het aanmelden van de gebruiker op het mobiele apparaat.
- Het tonen van de lijst van formulieren.

Het demonstreren verliep wat moeizaam aangezien het niet eenvoudig is om het op een mobiel apparaat te tonen. Om het te kunnen tonen op een mobiel apparaat moet dit gedeployed worden op het apparaat en anderzijds via een simulator. Het tonen op een iOS kun je niet simuleren aangezien je daarvoor een iMac nodig hebt. Ionic bied hiervoor een omgeving voor om de applicatie te testen via de webbrowser en op deze manier is het gedemonstreerd aan de stakeholder en producteigenaar. Fysiek is het nog niet te testen aangezien het dan beschikbaar moet zijn via de Apple store en/of Play Store.

Uit de sprint review sessie was het volgende gekomen:

- Een oplossing waarbij dezelfde implementatie via de mobiele applicatie en de webapplicatie gebruikt kan worden.

Dit is niet direct meegenomen bij het opzetten van de mobiele applicatie aangezien dit extra tijd met zich meebrengt en er goed onderzocht moet worden welke opties er mogelijk zijn.

Dit punt is opgenomen in de product backlog.

9.5. Retrospective

Wat ging goed?
- Aanvragen van een developer account bij Apple.
Wat ging niet goed?
- Het testen van de applicatie op een fysiek apparaat.
- Het proces voordat acceptatie plaats duurde erg lang wat resulteerde dat ik de opzet niet kon testen op iOS.
Welke acties moeten er komen om dit te voorkomen?
- Aanschaf van een iMac computer
- iMac beschikbaar stellen zodat het op werk getest kan worden.

10. Sprint 6: “Versturen formulier”

10.1. Planning

Op 12 februari 2016 is er een sprint planning geweest na de sprint review en is met de producteigenaar en stakeholder gekeken naar de product backlog. Door de producteigenaar is aangegeven dat hij op vakantie gaat van 19 t/m 26 februari 2016. Zijn taken worden waargenomen door de heer van Ewijk als er vragen en of onduidelijkheden zijn. Er is daarom besloten om de sprint review op te schuiven naar maandag 29 Februari 2016. Wij zijn door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld.

#PBI	#R	#UC		Effort
347	17	UC10	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat het formulier invullen, zodat ik dit niet met pen en papier moet invullen.	3
348	18	UC10	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat het formulier versturen, zodat de formulierbeheerder het formulier digitaal aangeleverd krijgt.	5
349	19	UC10	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat een validatie melding, zodat ik mijn fouten kan herstellen in het formulier.	2

Risicoklasse
Na het opzetten van de use case wordt de risicoklasse ingeschat. De risicoklasse bepaalt uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICO-KLASSE	ZWAARTE
UC10	**Functionaliteit**	H	H	A	●●●
	Performance	M	L	C	●
	Inpasbaarheid	M	M	B	●●
	Gebruiksvriendelijkheid	M	H	B	●●
	Portabiliteit	H	H	A	●●●

De testen zijn uitgewerkt in hoofdstuk 10.3
De sprint goal is:

- Het versturen van het formulier vanuit de mobiele applicatie naar de server.

10.2. Uitvoering

Formulier invullen
Voordat een formulier ingevuld kan worden moeten de volgende stappen doorlopen worden:

1. Ophalen formulier data van de API.
2. Tonen formulier met velden inclusief veld eigenschappen.
3. Validatie melding tonen.

AngularJS biedt standaard een groot verscheidenheid aan verschillende standaard directives zoals ng-app, ng-bind, ng-repeat, etc. Directives[25] zijn markeringen op een DOM-element (zoals een attribuut, elementnaam, commentaar of CSS-klasse) die ervoor zorgen dat de HTML-compiler van Angular een bepaald gedrag aan dat DOM-element koppelt. Daarnaast zijn ook directives beschikbaar die derden hebben ontwikkeld.

Formulier generatie
Er wordt gebruik gemaakt van de Angular-Formly[26] directive. Angular-Formly is een directive die het mogelijk om dynamisch aan de hand van JSON Data een formulier te genereren. Angular-Formly biedt de volgende functionaliteit:

- Templates, om formulier te tonen op basis van platform type.
- Expressie parameters, om velden afhankelijk te maken van andere velden.
- Uitbreiden met eigen velden, om nieuwe veldtypes te definiëren.
- Controller, per veldtype is het mogelijk om functionaliteit toe te voegen zodat het mogelijk wordt om lookup data van een externe API op te halen.
- Validatie, biedt mogelijkheid om validatie melding per veld af te handelen.

Het gebruik van deze directive zorgt ervoor dat het formulier eenvoudig uitbreidbaar is naar de toekomst toe met nieuwe velden. Daarnaast ook de mogelijkheden om de layout te definiëren. Aangezien de JSON direct ook de configuratie bevat van wat aan de directive word meegegeven kun je dit makkelijk uitbreiden. Het grote voordeel hiervan is dat in de toekomst niet de mobiele applicatie opnieuw gedeployed moet worden om de nieuwe veldtypes te gebruiken. Dit wordt gewoon als configuratie in de JSON Data meegegeven.

Ophalen formulier data van de API
Voor het ophalen van de data van het formulier is er voor gekozen om aan de API kant de formulier gegevens om te zetten naar een structuur dat Angular-Formly begrijpt. Als dit namelijk aan de cliënt kant zou gebeuren ben je afhankelijk van Angular-Formly. Aan de server kant heb je meer mogelijkheden om ook specifieke configuratie mee te nemen van velden. Bij een wijziging is het niet noodzakelijk om de gehele mobiele applicatie opnieuw te deployen.

Tonen formulier met velden inclusief veld eigenschappen
De API retourneert nu rechtstreeks de data die de directive Angular-Formly verwacht zodat dit eenvoudig implementeerbaar is. De verantwoordelijkheid van het opbouwen van het formulier wordt nu door Angular-Formly gedaan.

[25] https://nl.wikipedia.org/wiki/AngularJS#Belangrijke_directives
[26] http://angular-formly.com

10.3. Testen

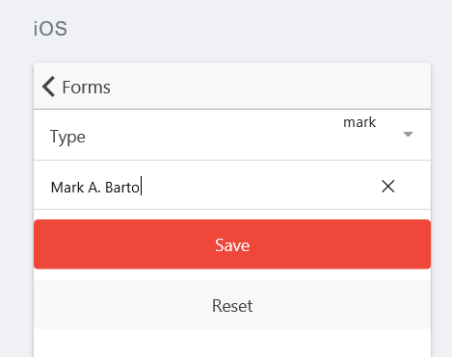
Functionaliteit
Voor het versturen van het formulier (UC10) zijn gedetailleerde testcases opgesteld. Hierbij zijn de gedetailleerde testcases uitgevoerd. Voor een volledige lijst zie Testrapport in Externe bijlagen

Er zijn geen fouten gevonden

Portabiliteit
De portabiliteit van de mobiele applicatie is uitgevoerd in een simulator op verschillende apparaten. Hierbij zijn de functionele tests uitgevoerd op deze apparaten.

Er zijn geen fouten gevonden

Gebruiksvriendelijkheid



Er is door een willekeurige gebruiker een usability test uitgevoerd. Daarbij zijn dezelfde functionele tests uitgevoerd. Dit is wederom via een simulator gedaan voor de verschillende mobiele apparaten. De volgende resultaten zijn hieruit gekomen:

- Het is niet duidelijk wanneer een veld verplicht is. Aangeven met een *
- De verschillende opties van het veld ‘select’ is niet duidelijk weergeven.
- De button is rood voor het versturen maar dit moet groen zijn.
- De tekst ‘Save’ is niet duidelijk, iets meer in de trend van ‘Send Form’.

Deze punten zijn meegenomen in de productbacklog.

Performance
De performance test kan pas worden uitgevoerd als de mobiele applicatie in productie wordt genomen. Aangezien er nu gebruik wordt gemaakt van een simulator levert dit geen passende test op.

10.4. Sprint review

Op maandag 29 Februari 2016 is er een sprint review geweest over de zesde sprint.

Hierbij zijn de volgende aspecten gedemonstreerd:

- Het versturen van een ontworpen formulier via de mobiele applicatie.

Aangezien er nog geen beschikking is over een fysiek apparaat is het wederom getest via een simulator. Daarbij is een formulier ontworpen met velden en verstuurd via een mobiel apparaat. In Sprint 3 was aangegeven dat het versturen nog niet getest kon worden. In deze Sprint is dit direct meegenomen. De demonstratie verliep zonder problemen.

10.5. Retrospective

Wat ging goed?
- De demonstratie van het versturen van het formulier via e-mail.

Wat ging niet goed?
- Het testen van de applicatie op een fysiek apparaat.
- Er is nog geen iMac beschikbaar zodoende ik verplicht ben om het thuis te compileren en te testen

Welke acties moeten er komen om dit te voorkomen?
- Aanschaf van een iMac computer
- iMac beschikbaar stellen zodat het op werk getest kan worden.

11. Sprint 7: “Offline werken”

11.1. Planning

Op 29 februari 2016 is er een sprint planning geweest na de sprint review en is met de producteigenaar en stakeholder gekeken naar de product backlog. Wij zijn door de belangrijkste user stories in de product backlog gegaan en zijn daar taken aan gekoppeld.

#PBI	#UC	#R	Titel	Effort
532	UC10	21	Als geregistreerde formuliergebruiker wil ik offline kunnen werken, zodat als ik onderweg ben het formulier later kan versturen.	8

Risicoklasse
Na het opzetten van de use case wordt de risicoklasse ingeschat. De risicoklasse bepaalt uiteindelijk de zwaarte van het testen.

#	KENMERK	SCHADE	FAALKANS	RISICO-KLASSE	ZWAARTE
UC10	**Functionaliteit**	H	H	A	●●●
	Performance	M	L	C	●
	Inpasbaarheid	M	M	B	●●
	Gebruiksvriendelijkheid	M	H	B	●●
	Portabiliteit	H	H	A	●●●

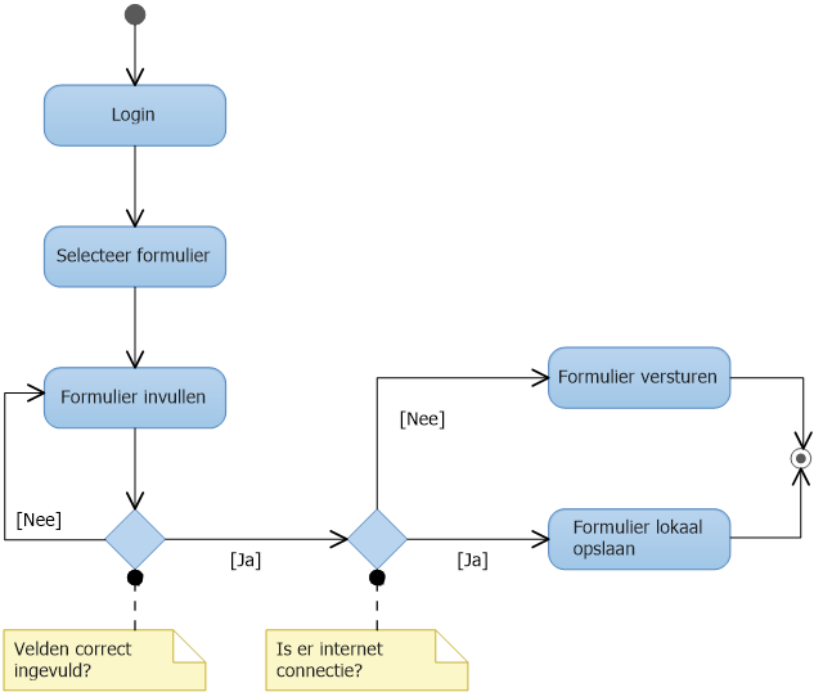
De testen zijn uitgewerkt in hoofdstuk 11.3

De sprint goal is:

- Het offline werken van de mobiele applicatie.

11.2. Uitvoering

Als een ingevuld formulier verstuurd wordt moet een internetverbinding zijn die de gegevens naar de API stuurt. Het probleem is dat als er geen internet connectie is dat het formulier niet verstuurd kan worden. De gebruiker krijgt hierbij een melding. Er is een internet connectie vereist om het formulier te versturen. De gebruiker moet het dan nog een keer proberen als er wel internet is. Dit is natuurlijk geen gewenste situatie. In praktijk komt het voor dat een gebruiker geen internet heeft tijdens het verzenden van het formulier. Daarom moeten de gegevens op het mobiele apparaat tijdelijk worden opgeslagen zodat het formulier opnieuw verstuurd kan worden als er wel internet is.



Voor het tijdelijk opslaan van het formulier moeten de volgende stappen doorlopen worden:

1. Gebruiker meldt zich aan.
2. Selecteert het formulier.
3. Het formulier wordt ingevuld en na het invullen klikt hij op verzenden.
4. Systeem controleert op validatie van de velden.
5. Als validatie mislukt moet het formulier worden aangepast.
6. Als validatie lukt wordt de verbinding gecontroleerd.
7. Als er verbinding is wordt het formulier naar de API verstuurd.
8. Als er geen verbinding is wordt het formulier lokaal opgeslagen.

Voor het controleren op een verbinding met internet kan er gebruik worden gemaakt van een cordova network plugin[27]. Cordova maakt het mogelijk om te kunnen communiceren met het fysieke apparaat. Voor het opslaan van de data op het apparaat word er teven gebruik gemaakt van een cordova file plugin[28] . Het is ook mogelijk om gebruik te maken van de HTML5 lokale opslag maar de limiet is dat er totaal maar 5MB kan worden opgeslagen. Voor de implementatie is gebruik gemaakt van de strategy en factory pattern. Er zijn drie services gemaakt. De eerste service wordt gebruikt voor het lokaal opslaan van het formulier. De tweede service wordt gebruikt voor het sturen van het formulier naar de API. Er is een derde service gemaakt waar business logica in zit waar de strategie wordt bepaald. De strategie wordt bepaald aan de hand van de internetverbinding. Als er internetverbinding is wordt er gebruik gemaakt van de ene service waar het formulier naar de API wordt verstuurd en anders van de andere service waar de informatie lokaal wordt opgeslagen.

[27] http://ngcordova.com/docs/plugins/network/
[28] http://ngcordova.com/docs/plugins/file/

11.3. Testen

Aangezien er alleen iets functioneels is gewijzigd bij het versturen van het formulier is alleen op de kwaliteitsattribuut functionaliteit getest. Verder is er een regressietest uitgevoerd om te controleren of de werking nog klopt.

Functionaliteit
Er is dit keer niet zwaar getest maar gemiddeld en daarbij is een testcase opgezet om te controleren of het formulier nu lokaal wordt opgeslagen. Voor een volledige lijst zie Testrapport in Externe bijlagen.

11.4. Sprint review

Op vrijdag 11 Maart 2016 is er een sprint review geweest over de laatste sprint.

Hierbij zijn de volgende aspecten gedemonstreerd:

- Het offline opslaan van een formulier via de mobiele applicatie.

Het offline testen is wederom getest via een simulator. Het internet is uitgezet voordat het formulier verstuurd wordt. Daarbij wordt het formulier lokaal opgeslagen.

Er was helaas onvoldoende tijd om de weergave te maken dat het offline formulier opnieuw verstuurd kan worden.

11.5. Retrospective

Wat ging niet goed?
- Er is al meerdere keren aangegeven dat er een iMac en verschillende mobiele apparaten beschikbaar moeten komen. Dit om goede testen uit te kunnen op fysieke apparaten. Aangezien eerst de producteigenaar en daarna de officemanager op vakantie was is dit uitgesteld.
- Te weinig tijd en dit kwam enerzijds dat de sprint planning verschoven was in verband van vakantie. Daarnaast door een paar dagen ziekte waardoor ik niet alles af heb kunnen maken.

Welke acties moeten er komen om dit te voorkomen?
- Aan het begin van de eerste sprint opdracht geven wat er benodigd is.

12. Evaluatie

12.1. Productevaluatie

Er wordt teruggekeken aan de hand van de globale eisen en wensen van het systeem of het voldoet aan de verwachtingen.

Nieuwe technologie

Het systeem is opgezet volgens de laatste nieuwe technologieën. Hierbij is gebruik gemaakt van ASP.NET WebAPI, nServiceBus, AngularJS, Ionic. Daarnaast is gebruik gemaakt van de laatste coding standaarden binnen FourICT zodat het eenvoudig onderhoudbaar is.

Schaalbaarheid en koppelbaarheid

Bij het maken van architectuur keuzes zoals de ESB is rekening gehouden met de schaalbaarheid en koppelbaarheid van het geselecteerde product. ESB biedt mogelijkheden om verschillende leveranciers te koppelen. Verder biedt het mogelijkheden voor verticaal schalen om het in een Cloud omgeving te deployen. Daarnaast kan het horizontaal schalen door middel van een distributor de load te verdelen. Auth0 biedt verschillende abonnementsvormen waarbij geschaald kan worden naar de load van het aantal gebruikers. Daarnaast het koppelen met andere identity providers of mogelijkheden om de gebruikers anders te bewaren dan via de interne database.

Gebruiksvriendelijk systeem

Er zijn verschillende usability tests binnen het systeem uitgevoerd door een andere gebruiker. Daar zijn resultaten uitgekomen waarbij gekeken is naar de gebruiksvriendelijkheid van het systeem. Denk daarbij aan de gebruikte kleuren en/of bepaald gedrag van het systeem. De usability tests voor de mobiele applicatie zijn uitgevoerd in een simulator. Dit had op een fysiek apparaat moeten gebeuren want dit geeft toch net een andere ervaring. Denk aan het selecteren van een veld met je vinger of met de muis. Het tikken van tekst in een te klein veld. Het controleren van de velden bij het omdraaien van het scherm. Dit is een andere gebruikerservaring en daar kunnen andere resultaten uitkomen dan in een simulator.

Beschikbaar op mobiele apparaten

Het formulier is beschikbaar op verschillende mobiele apparaten zoals iOS en Android. Er zijn testen uitgevoerd via een simulator. Er was enerzijds geen mogelijkheid om het te testen op een fysiek apparaat aangezien er geen toestel beschikbaar is gesteld. Doordat er geen Apple computer beschikbaar is kon het niet via een fysiek apparaat worden getest. Het Ionic platform biedt mogelijkheden om het voor andere mobiele platformen beschikbaar te maken.

Flexibiliteit

Het systeem is flexibel genoeg om aan te passen aan veranderde omstandigheden. Auth0 is gebaseerd op openstandaarden waardoor eenvoudig andere leveranciers gebruikt kunnen worden. ESB is gekozen om afhankelijkheden tussen verschillende leveranciers te vermijden.

Nieuwe type velden

Het systeem ondersteund een generieke manier om nieuwe type velden te definiëren. Door middel van een html template kan de eigenschappen van velden worden opgegeven. Op deze manier wordt het mogelijk gemaakt om eenvoudig velden toe te voegen. Voor de implementatie van het scannen van een barcode en maken van foto's moet gebruik worden gemaakt van de functionaliteit die Cordova aanbiedt.

De klant moet zich kunnen aanmelden voor het gebruik van het systeem, zodat er alleen toegang is tot zijn eigen ontworpen formulieren en niet van anderen.

Het aanmeldt proces blijkt niet zo eenvoudig te zijn om te implementeren in de client. De documentatie bleek niet altijd up-to-date te zijn en was vrij technisch van aard. In het begin was het veel uitzoeken om de werking te doorgronden. Uiteindelijk ben ik erg tevreden met het gebruik van Auth0.

Verder het risico om beveiligingslekken met Auth0 geheel uit te sluiten klopt gedeeltelijk. Het autoriseren aan de Auth0 is wel beveiligd maar aan de API kant moet de token of de gebruiker geautoriseerd is wel worden gecontroleerd. Daarnaast moet de API ook beveiligd worden zodat alleen verzoeken vanuit de web- en/of mobiele applicatie gerechtigd zijn (CORS). Als dit vergeten wordt kan iedereen gebruik maken van de API zonder autorisatie token. Verder moet de API ook via een beveiligde verbinding opgezet worden.

Verder is nog steeds de vraag wat Auth0 met de gebruikersgegevens doet. Het risico dat een hack via Auth0 plaats vindt is vele malen groter dan op het huidige systeem. Aangezien het hier niet gaat om risicovolle privacygevoelige informatie is de impact kleiner.

De klant kan zijn eigen digitale formulier ontwerpen, invoervelden definiëren en restricties opgeven voor het valideren van invoervelden.

Er kan een standaardformulier worden ontworpen. Met het gebruiksgemak om nieuwe velden te introduceren en eigenschappen van velden toe te voegen ben ik erg tevreden. Door gebruik te maken van een Field Directive kun je snel een nieuw veld definiëren. Daarnaast voor het uitbreiden van een veld maak je gebruik van de Property Directive. De directives zijn herbruikbare componenten die je in de .html template kunt gebruiken. Dit biedt erg veel flexibiliteit waardoor de directive er uiteindelijk voor zorgt dat de eigenschap in de database wordt weggeschreven. Het gebruik van de Angular-Formly directive geeft nog meer flexibiliteit aangezien die ervoor zorgt om het formulier dynamisch op te bouwen op de mobiele applicatie. Dit geeft nog meer mogelijkheden naar de toekomst toe zoals de vormgeving bepalen, afhankelijkheid met andere velden, etc.

De mogelijkheid dat de klant binnen het systeem gebruikers toegang kan geven tot het digitale formulier.

De implementatie van Auth0 met de SDK om gebruikers toegang te geven was niet zo eenvoudig zoals in eerste instantie gedacht was. Bij het aanmaken van gebruikers heb ik geen gebruik gemaakt van een e-mail en wachtwoord. Dit kwam door een tekortkoming binnen Auth0. Bij het aanmaken van een gebruiker moet een wachtwoord worden gegenereerd. Het gegeneerde wachtwoord wordt niet automatisch naar de gebruiker gestuurd na het verifiëren van het e-mailadres. Om dit wel mogelijk te maken moet hier een eigen implementatie worden ontwikkeld. Dit brengt weer extra beveiligingsrisico's met zich mee. Er is uiteindelijk in overleg besloten om afgezien van het wachtwoord gebruik te maken van een e-mail met verificatie code.

De klant moet kunnen bepalen op welke manier het digitale formulier wordt afgehandeld. Standaard via de mail.

Het gebruik van een ESB-oplossing met behulp van nServiceBus biedt mogelijkheden om onafhankelijk van de leverancier de formulier gegevens door te sturen. Nu alleen via de e-mail maar in de toekomst naar achterliggende systemen van de klant of via een maatwerkoplossing. Ik ben nog niet tevreden met de manier waarop de gegevens worden opgeslagen in de queue. Er is een limiet van 2MB.

De medewerker van de klant kan zich aanmelden om het digitale formulier te bekijken.

Nadat een formuliergebruiker is aangemeld op de mobiele applicatie kan het digitale formulier bekeken worden. De lijst wordt samengesteld op basis van de toegekende formulieren en de formulieren die de formuliergebruiker zelf heeft gemaakt.

Het formulier moet offline beschikbaar zijn in de app zodat als er geen verbinding tot stand kan worden gebracht zodat de gegevens lokaal worden opgeslagen en niet verloren gaan.

Er is voldaan aan deze eis. Voor het "offline werken" ben ik niet helemaal tevreden. Het formulier moet nadat het lokaal is opgeslagen automatisch naar de API worden doorgestuurd als er weer verbinding is. Dit zou een betere oplossing zijn dan om een overzicht te tonen van opgeslagen formulieren. Aangezien anders vergeten wordt door de gebruiker om het formulier nogmaals te versturen.

Bij het versturen van het formulier moet er een validatie plaats vinden van de invoervelden.

Er vindt validatie plaats van de verschillende invoervelden. Dit is gebaseerd op eenvoudige validatie waarbij gecontroleerd wordt of het veld verplicht is. Uiteindelijk is de bedoeling om de validatie uit te breiden. Dit door middel van data types zoals telefoonnummer, e-mailadres. Etc. Daarnaast afhankelijkheden met andere velden.

Conclusie
Er is voldaan aan alle globale eisen. Ik ben tevreden met het uiteindelijke resultaat en de opgeleverde kwaliteit tot nu toe. Het product biedt nog wel te veel basis functionaliteit waardoor het nog geen volledig product is. Er mist nog extra functionaliteit als velden voor het maken van foto's, barcode scannen, handtekening zetten. Daarnaast het automatische synchronisatie proces bij het offline werken. Verder is het doorsturen via e-mail van het formulier een standaard functionaliteit. Het resultaat moet van het formulier eerder worden doorgestuurd naar een achterliggend bedrijfssysteem van de gebruiker.

De opdrachtgever heeft aangegeven dat aan de verwachtingen is voldaan en dat hij tevreden is met het eindproduct. Daarnaast wordt het toegevoegd aan het productportfolio van FourICT.

12.2. Procesevaluatie

Ik ben tevreden over het verloop van het proces. Er is gebruik gemaakt van een aangepaste versie van een scrum aanpak. Dit aangezien er niet in team verband wordt gewerkt maar individueel. Er is gebruik gemaakt van korte sprints, twee weken, waardoor er snel feedback wordt gegeven van het opgeleverde product. Hierdoor kon er snel ingesprongen worden op gewijzigde requirements. Zoals het openstellen van de registratie/aanmelden van iedere gebruiker in de eerste sprint. Tijdens de sprintplanning zijn er user stories geselecteerd in samenspraak met de producteigenaar. Voor de detaillering van een user story is er een use case gemaakt. Het gebruik van use cases is een goede manier om te dienen als testbasis. Ik merkte wel op dat niet altijd een use case kon worden opgezet zoals bij een niet-functionele eis. Daarnaast bij gewijzigde requirements bleek het steeds lastiger te worden om use cases bij te houden. Aangezien testen ook gebaseerd zijn op use cases moeten deze ook opnieuw worden uitgevoerd. Het gebruik van een product risicoanalyse (PRA) werkt erg goed om te bepalen wat de faal en risico is. Hierdoor kon de zwaarte van de test worden bepaald. Tijdens de sprint heb ik een retrospective bijgehouden om het proces te evalueren.

De communicatie met de opdrachtgever verliep goed en door de korte lijnen zijn vragen snel beantwoord. Helaas bleek er op het moment van testen van de mobiele applicatie geen middelen beschikbaar te zijn zoals deze in het PvA zijn aangegeven. Zoals een Apple Computer om te kunnen testen voor iOS. Daarnaast verschillende mobiele apparaten om de test te kunnen uitvoeren. Ondanks verzoeken tijdens de retrospective kwam dit door vakantieplanning van de opdrachtgever en officemanager. Er is daarom gekozen om de test uit te voeren via een simulator. Aan het begin van de eerste Sprint had ik dit al moeten aangeven zodat dit al ging had kunnen worden gezet.

Daarnaast merkte ik dat door het plannen van alle sprints geen rekening is gehouden met de feestdagen en/of ingeplande vakantie. Aan het einde van Sprint 2 is de Retrospective en Sprint planning gepland een dag eerder aangezien vrijdag op een feestdag viel. Hier had ik geen rekening mee gehouden waardoor een dag eerder de werkzaamheden klaar hadden moeten zijn. Uiteindelijk heeft dit geen consequentie gehad op de planning. Door ziekte van een paar dagen in de laatste Sprint is het helaas niet gelukt om alle activiteiten uit te voeren.

Aangezien het project individueel word uitgevoerd ben ik van mening dat testen door iemand anders had moet gebeuren. Als ontwikkelaar ben je snel geneigd om niet out-of-the-box te denken en ben je te veel gefocust om de test te laten slagen.

Ik merkte dat door het opzetten van globale testcases en/of gedetailleerde testcases niet altijd alles is gedekt. In Sprint 1 bleek tijdens de retrospective nog een probleem te zitten. Tijdens het herladen van de pagina moet opnieuw aangemeld worden. Hier was geen testcase voor opgezet. Een retrospective dient daarnaast als een whitebox test waardoor ook nog eens tijdens de demonstratie wordt getest.

12.3. Beroepstaken

12.3.1. Uitvoeren analyse door definitie van requirements (niveau 3)

Vooraf heb ik interviews gehouden met de stakeholders om de eisen en wensen te achterhalen. Deze eisen en wensen zijn vastgelegd in het analyse document. Hierin is eerst een omschrijving van het systeem opgesteld en de positionering van het systeem bepaald. Hierdoor wordt duidelijk welke onderdelen voorkomen in het systeem en de externe entiteiten waar vanuit het systeem interactie mee heeft. Daarna zijn de eisen en wensen onderverdeeld in functionele en niet-functionele eisen. Daarnaast zijn er user stories opgesteld en vastgelegd in de product backlog. Deze zijn besproken met de producteigenaar waarbij de prioriteit is vastgesteld. Tijdens de sprint planning zijn naar aanleiding van de gekozen user stories use case beschrijvingen opgesteld. Deze use cases zijn in samenspraak met de producteigenaar opgesteld. Na goedkeuring zijn taken aan de user story opgesteld. Verder is er een use case diagram opgesteld. Daarnaast voor traceerbaarheid is er een opsomming gemaakt van alle eisen en wensen.

12.3.2. Opstellen gegevensmodel voor een database (niveau 3)

Bij het ontwerpen van een formulier is er een gegevensmodel opgesteld, zie hoofdstuk Sprint 2: “Formulier ontwerpen”. Dit is uitgewerkt in een E-R diagram hierin zijn 1-op-1 relaties en 1-op-veel relaties gebruikt. Verder is er een logisch datamodel opgezet. Daarnaast is er gebruik gemaakt van JSON Data om de veld eigenschappen zo dynamisch mogelijk te houden.

12.3.3. Ontwerpen, bouwen en bevragen van een database (niveau 3)

Er is na het opstellen van het gegevensmodel een database ontworpen. Hiervoor is in het analyse document een Relationeel Representatie Model (RRM) en een Relationeel Implementatie Model (RIM) ontworpen. Daarnaast is een modeldictionary ontworpen waarin de verschillende entiteit eigenschappen zijn toegelicht. Het opzetten en raadplegen van relationele data gebeurt met behulp van het Entity Framework van Microsoft.

12.3.4. Ontwerpen systeemdeel (niveau 3)

Bij het ontwerpen van het systeem is rekening gehouden met de verschillende eisen en wensen van de opdrachtgever. Voor het ontwerpen van het systeem is gebruik gemaakt van verschillende architectuurbeslissingen, zie hoofdstuk Sprint 0: “Analyse”. Er is daarbij rekening gehouden met de verschillende kwaliteitseisen zoals efficiëntie, koppelbaarheid, portabiliteit en uitbreidbaarheid. Er is gebruik gemaakt van de cliënt-server architectuur. Voor de cliënt is gekozen voor een SPA-architectuur. Dit zorgt ervoor dat de verantwoordelijkheid van het MVC pattern naar de cliënt is verplaatst. Als architectuurstijl is gebruik gemaakt van het drie-lagen model. Hierdoor worden de verschillende lagen presentatie, business en data laag van elkaar gescheiden. Dit is zo opgezet dat elke laag zijn eigen verantwoordelijkheid heeft. Daarnaast is een keuze gemaakt voor een enterprise service bus (ESB) oplossing, zie hoofdstuk Sprint 6: “Versturen formulier”.

12.3.5. Bouwen applicatie (niveau 3)

Er zijn twee verschillende applicaties gebouwd. Een webapplicatie en een mobiele applicatie. De webapplicatie is volgens de client-server architectuur opgezet. Zie uitwerking in Sprint 1: “Initiële opzet”. Daarbij is gebruik gemaakt van het AngularJS SPA framework. Voor de service laag is gebruik gemaakt van het ASP.NET Web API framework om te communiceren met de webapplicatie en de mobiele applicatie. Voor de business laag is gebruik gemaakt van de ESB-oplossing nServiceBus. Voor de data laag is gebruik gemaakt van het Entity Framework om te communiceren met de database. De mobiele applicatie (hybride) is ontwikkeld met het Ionic Framework en het AngularJS SPA framework.

12.3.6. Uitvoeren van en rapporteren over het testproces (niveau 3)

- Tijdens elke sprint planning is de risicoklasse bepaald. De risicoklasse is bepaald aan de hand van de faalkans en de schade. Aan de hand van de risicoklasse kon de zwaarte van de test bepaald worden. Dit bepaalt uiteindelijk de teststrategie. Zie

Testplan. De testen zijn vastgelegd als testcases in Visual Studio Team Services. Voor een volledige lijst zie Testrapport in Externe bijlagen. Er is gebruik gemaakt van automatische testen zoals unittests en end-2-end test. Verder is er gebruik gemaakt van een OTAP-omgeving. De continu integratie binnen Visual Studio Team Services zorgt ervoor dat automatisch een nieuwe release wordt gemaakt waarbij automatisch deze testen worden uitgevoerd.

Afkortingenlijst

Afkorting	Omschrijving
Producteigenaar	De verantwoordelijke voor de product backlog, waarmee de prioriteiten van de belanghebbenden naar voren komen.
Ontwikkelteam	een groep van specialisten verantwoordelijk voor het ontwikkelwerk en de realisatie van een in principe kant en klaar product(uitbreiding).
Storypoint	Worden gebruikt in planning om de omvang van een klus (story) aan te geven. Door een bekende korte klus 1 of 2 punten te geven kunnen grote klussen ingeschat worden als bijvoorbeeld driemaal zo lang of tienmaal zo lang.
Epic	Een epic heeft dezelfde vorm als een User story, alleen omvat deze een grotere scope.
Use case	Een use case beschrijft omvat alle manieren waarop het systeem gebruikt kan worden om een bepaald doel voor een bepaalde gebruiker te behalen.
Retrospective	Een evaluatie van de sprint om de kwaliteit van de volgende sprint te waarborgen.
Requirements	De eisen en wensen van de opdrachtgever voor het te ontwikkelde software.
Daily Scrum	Een korte meeting van minstens een kwartier met het team. In deze meeting wordt per teamgenoot antwoord gegeven op: wat ga je doen? Wat heb je gedaan? Zijn er nog problemen?
Definition of Done	De Definition of Done is een lijst waar het aan moet voldoen om de kwaliteit te waarborgen voordat het product backlog item klaar is.
JWT-Token	Een JWT staat voor JSON Web token en word gebruikt om tussen twee verschillende partijen veilig te kunnen communiceren. Deze wordt gebruikt voor autorisatie en bevat speciale eigenschappen om de identiteit vast te leggen.
Cross-origin resource sharing (CORS)	Dit is een beveiligingsoverweging om ongeautoriseerde aanvragen te voorkomen waarbij siteX geen toegang heeft tot de resources van SiteZ.
GFRS	Generiek Formulier Registratie Systeem, naam van het systeem

VSTS	Visual Studio Team Services[29], cloud-oplossing van Microsoft voor versiebeheer, release management, agile planning tools.
VS	Visual Studio, ontwikkelplatform van Microsoft
Mobiele applicaties	Dit is een applicatie die speciaal ontworpen is om te draaien op verschillende soorten mobiele apparaten zoals een tablet of smartphone.
Windows Applicaties	Een Windows applicatie is een applicatie die op de computer lokaal staat geïnstalleerd.
Web Applicaties	Een webapplicatie is een applicatie die op het internet staat en die via elke browser benaderbaar is op mobiele apparaten zoals een tablet of smartphone.

[29] https://azure.microsoft.com/nl-nl/services/visual-studio-team-services/

Geraadpleegde literatuur

Microsoft Application Architecture Guide, 2nd Edition. (2009). Opgehaald van https://msdn.microsoft.com/en-us/library/ff650706.aspx

Rozanski, N., & Woods, E. (2011). *Software Systems Architecture, 2nd Edition.* Pearson Education (Us).

Swart, N. d. (2010). *Handboek requirements: brug tussen business en ict.* Eburon Business.

Tre, G. D. (2007). *Principes van databases.* Pearson Benelux B.V.

Warmer, J., & Kleppe, A. (2011). *Praktisch UML, Vijfde editie.* Pearson Benelux B.V.

Bijlage 1: Vergelijking webapp, native app en hybride app

		Voordelen	Nadelen
Type app	**Webapp**	• Generiek platform • Geen goedkeuring store • Altijd de laatste versie • Lage ontwikkelkosten	• Altijd verbinding met internet • Niet alle functionaliteit apparaat zijn te gebruiken. • Afhankelijk HTML5
	Native app	• Gebruik van beschikbare functionaliteiten op het apparaat (Sensors, GPS, Camera's). • Geen internetverbinding (offline) • Snelheid	• Er moet specifieke kennis zijn van de taal van de verschillende ontwikkelplatformen (Objective-C/Swift, Java en C#). • Goedkeuring plaatsing app • Hoge ontwikkelkosten
	Hybride app	• Gebruik van beschikbare functionaliteiten op het apparaat (Sensors, GPS, Camera's). • Geen internetverbinding (offline) • Snelheid • Gebaseerd op bestaande technieken	• Minder grafische mogelijkheden • Niet optimaal gebruik maken van unieke gebruiksfuncties van het platform • Laadtijden

Bijlage 2: Vergelijking mobiel, Windows en webapplicatie

Type applicatie		Voordelen	Nadelen
	Mobiel	• Ondersteuning voor mobiele apparaten • Ondersteuning offline gebruik • Altijd en overal onderweg toegankelijk	• Beperkte schermgrootte
	Windows	• Snelheid • Meer grafische mogelijkheden • Veiligheid	• Alleen lokaal toegankelijk op de computer • Installatie vereist
	Web	• Het is platformonafhankelijk (Mac, Windows, Unix) en draait op mobiele apparaten. • Er is geen installatie vereist om het te installeren op de computer. • Altijd toegang mits er een verbinding is met internet. • Aanpassingen aan de software zijn eenvoudig te distribueren oftewel makkelijk onderhoudbaar is.	• Internetverbinding is noodzakelijk. • Afhankelijkheid van de leverancier.

Bijlage 3: Vergelijking BizTalk en nServiceBus

	Microsoft BizTalk	nServiceBus[30]
Gebruiksvriendelijk **(**Makkelijk in gebruik?)	Nee, complexe installatie, veel kennis van opzetten.	Ja, geen complexe installatie noodzakelijk.
Functionaliteit (Ondersteuning REST?)	Ja[31], uitgebreide mogelijkheden via BAM	Ja, beperkte mogelijkheden.
Onderhoudbaarheid (Monitoring en beheer)	Ja, uitgebreid toolbeheer	Ja, via ServicePulse, ServiceInsight en ServiceControl
Integratie Ondersteuning verschillende adapters?	MSMQ, WCF, SharePoint, RFID, EDI, SFTP	MSMQ, SQL Service, RabbitMQ, Azure Service Bus, Azure Storage Queues
Interoperabiliteit (Open Source?)	Nee	Ja
Beschikbaarheid (Gekeken naar de schaalbaarheid en beschikbaarheid van de service)	On Premises	Load balancer via distributor, Beschikbaar via Azure, eigen omgeving.
Ondersteuning (SLA?)	Ja, betaalde ondersteuning beschikbaar of via forums.	Ja, betaalde ondersteuning of via forums.
Prijs (Licentiekosten, hardware)	+50k	+-3k
Community Is er een grote community? Gebaseerd op de trends van Google.	Ja, via forums	Ja, via forums

[30] http://particular.net/nservicebus

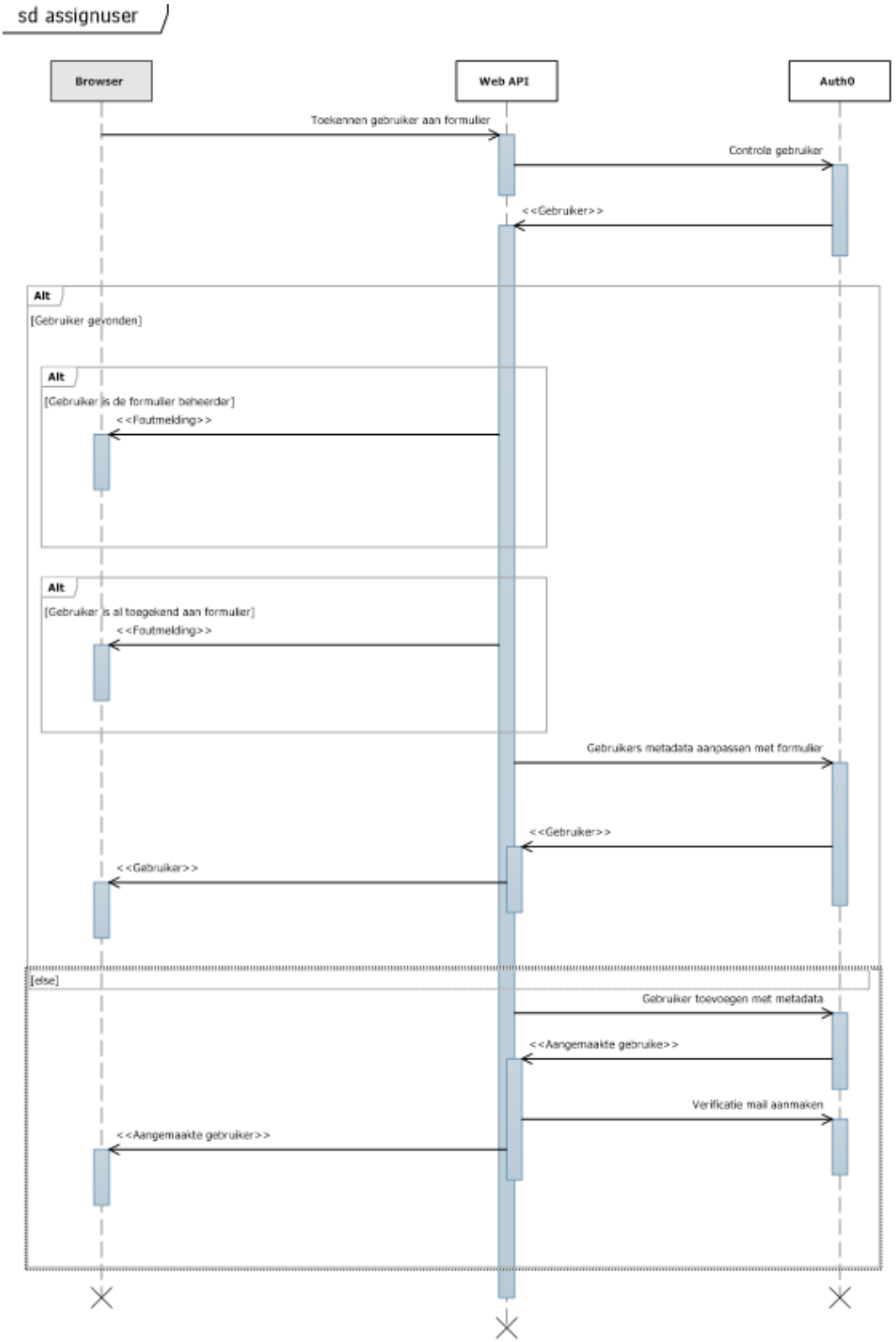
[31] https://msdn.microsoft.com/en-us/library/jj248701.aspx

Bijlage 4: Vergelijking Auth0 en AuthRocket

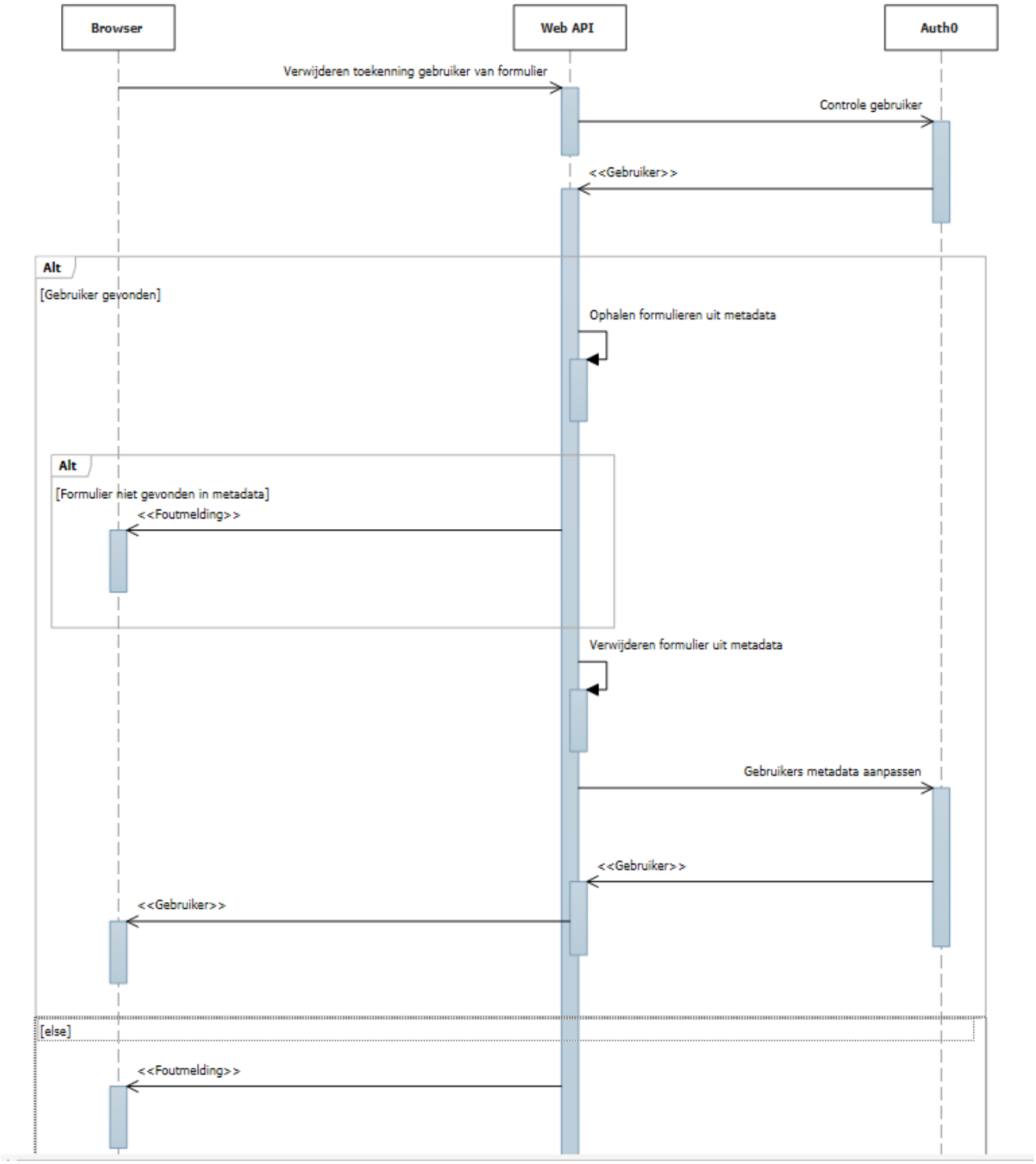
		Auth0[32]	AuthRocket[33]
Beveiliging (Voldoet aan beveiligingsnormen)	**Multifactor authenticatie**	Ja, via SMS	Nee
	Voldoet aan beveiligingsnormen	Ja, SOC2, Safeharbor	x
SDK (Ondersteuning voor verschillende ontwikkelplatformen)		Ja, 15+	Ja, 2
Integratie Hoe is de integratie met verschillende netwerken (identity providers)	**Sociale netwerken** (LinkedIn, Twitter, Facebook, etc.)	Ja, 30[34]+	Ja, 5
	Bedrijfsnetwerken (Azure, AD, WS-federation)	Ja, 10+	?
Interoperabiliteit (Ondersteund open standaarden)		JWT,OAuth, OpenID, SAML	JWT
Beschikbaarheid (Gekeken naar de schaalbaarheid en beschikbaarheid van de service)		Azure & Amazon[35]	Onbekend
Prijs		$12 per maand	$49 per maand
Community Is er een grote community? Gebaseerd op de trends van Google.		Ja	Nee

[32] https://auth0.com/why-auth0
[33] https://authrocket.com/
[34] https://auth0.com/docs/identityproviders
[35] https://auth0.com/availability-trust

Bijlage 5: Toekenning gebruiker



Bijlage 6: Verwijderen toekenning gebruiker



Externe bijlagen

- Plan van Aanpak
- Analyserapport
- Testplan
- Testrapport
- Studieplan

Plan van Aanpak

Ontwikkeling van een generiek formulierenregistratiesysteem bij FourICT B.V.

Opleiding: Informatica deeltijd

Begeleidend examinator: dhr. E.M. van Doorn

Tweede examinator: mevr. A.M.J.J Lousberg

Datum: 24-03-2016

Plaats: Zoetermeer

Versie: 1.0

Geschreven door: Mark Barto (12092452)

Versiebeheer

Versie	Datum	Omschrijving
0.1	20-11-2015	Initiale versie.
0.2	25-11-2015	Planning aangepast.
0.3	28-11-2015	Randvoorwaarden, Wijze van rapporteren bijgewerkt en planning aangepast.
1.0	24-03-2016	Laatste aanpassingen.

Inhoudsopgave

1. Opdrachtomschrijving

1.1. Organisatie

FourICT B.V. is een ICT-bedrijf dat in 2008 is opgericht en zich heeft gespecialiseerd in het ontwikkelen en integreren van complexe ICT-toepassingen in de Telecom, Media en Healthcare sector. FourICT bestrijkt het gehele traject van analyse, ontwerp, ontwikkeling, test en implementatie tot en met projectmanagement en beheer.

Het bedrijf bestaat uit een team van 10 vaste medewerkers en 3 zelfstandige ondernemers welke ondersteund worden door een project secretariaat. FourICT richt zich in het bijzonder op proces automatisering (BPMS AuraPortal), Enterprise Service Bus oplossingen, Microsoft development en mobiele app ontwikkeling.

1.2. Opdrachtgever

Voor dit project is de opdrachtgever Richard van Leeuwe van FourICT.

1.3. Probleemstelling

Er zijn nog veel bedrijven die gebruik maken van handmatige ingevulde (papieren) formulieren, waarvan de inhoud vervolgens weer handmatig moet worden overgezet naar één of meerdere backoffice systemen. Een kostbare activiteit, die niet alleen veel tijd kost maar ook fouten in de hand werkt en impact heeft op de uiteindelijke kwaliteit van de werkzaamheden.

In het verleden is bij FourICT een maatwerk systeem ontwikkeld voor de klant waarbij formulieren beheerd kunnen worden. Aangezien het huidige systeem gebaseerd is op achterhaalde technologie is het steeds moeilijker om het systeem te onderhouden. Daarnaast voldoet het huidige systeem niet meer aan de eisen van klant. Hierbij moet gedacht worden aan het offline kunnen werken zonder internetverbinding en de integratie met bedrijfssystemen. Verder voldoet het huidige systeem niet aan de gebruikte standaarden waarbij nagedacht is over de verschillende kwaliteitseisen als uitbreidbaarheid en koppelbaarheid. Omdat het uitbreiden van het huidige systeem een kostbare gelegenheid wordt is er gekozen om een nieuw systeem te ontwikkelen. Aangezien ook andere klanten bij FourICT interesse hebben in het systeem is uiteindelijk de vraag ontstaan om een generiek systeem te ontwikkelen gebaseerd op huidige nieuwe technologie.

De kracht van de applicatie zal een gebruiksvriendelijk systeem zijn dat eenvoudig te onderhouden is. Tevens heeft de applicatie een grote flexibiliteit m.b.t. het integreren van data van het formulier via e-mail of tot een maatwerk interface met het backoffice systeem van de klant.

1.4. Doelstelling

Het ontwikkelen van een generiek systeem waarbij klanten digitale formulieren kunnen ontwerpen en beheren. Zonder enige IT-kennis. En waarbij medewerkers in het veld de formulieren kunnen invullen.

2. Ontwikkelmethode en -omgeving

Binnen FourICT wordt scrum als standaard gebruikt als software methodiek. Aangezien scrum geheel zelfstandig door mij in verschillende rollen van scrum wordt uitgevoerd is gekozen om een aantal onderdelen aan te passen. Hierbij is gekozen om de daily scrum te laten vervallen. Dit aangezien alle rollen door mij zelfstandig wordt uitgevoerd en mijn opdrachtgever direct aanspreekbaar is als dit noodzakelijk is. Omdat Scrum een Agile methode is waarbij het niet noodzakelijk is om vooraf alles te documenteren is er gekozen om documentatie een onderdeel te maken van het inschatten van de taken van een userstory. Verder wordt er op de laatste dag van de sprint een review sprint gehouden waarin een demonstratie wordt gegeven van de opgeleverde user stories van de sprint. Na deze sprint review wordt direct de sprint planning gehouden waarin de planning voor de volgende sprint wordt bekeken en eventueel aangepast. De sprint review uitkomsten en de restrospective worden opgenomen in het afstudeerverslag.

Daarnaast wordt binnen FourICT Visual Studio Team Services als standaard gebruikt als repository voor de broncode. Verder biedt VSTS de Agile Scrum methodiek aan waarin de productbacklog met de user stories kan worden vastgelegd samen met de prioritering. Ook biedt het een scrumboard waarin precies kan worden bijgehouden hoe de vorderingen verloopt.

3. Afbakening opdracht

De opdrachtgever heeft de nadruk gelegd op een gebruiksvriendelijk systeem dat eenvoudig te onderhouden is. Daarnaast de grote flexibiliteit m.b.t. de integratie van data van het formulier rechtstreeks in de backoffice systeem van de klant. Voor de integratie van de data van het formulier zal dit alleen via e-mail beschikbaar zijn en niet via een maatwerk interface.

De volgende functionaliteiten moeten gerealiseerd worden voor het systeem:

- De klant moet zich kunnen aanmelden voor het gebruik van het systeem, zodat er alleen toegang is tot zijn eigen ontworpen formulieren en niet van anderen.
- De klant kan zijn eigen digitale formulier ontwerpen, invoervelden definiëren en restricties opgeven voor het valideren van invoervelden.
- De mogelijkheid dat de klant binnen het systeem gebruikers toegang kan geven tot het digitale formulier.
- De klant moet kunnen bepalen op welke manier het digitale formulier wordt afgehandeld. Standaard via de mail.
- De medewerker van de klant kan zich aanmelden om het digitale formulier te bekijken.

- Het formulier moet offline beschikbaar zijn in de app zodat als er geen verbinding tot stand kan worden gebracht zodat de gegevens lokaal worden opgeslagen en niet verloren gaan.
- Bij het versturen van het formulier moet er een validatie plaats vinden van de invoervelden.

4. Randvoorwaarden

Om deze opdracht succesvol te kunnen voltooien is het belangrijk dat onze opdrachtgever beschikbaar is. Verder is het belangrijk om tussentijdse bijeenkomsten te plannen om te verzekeren dat er wordt voldaan aan de eisen.

Het project zal worden uitgevoerd in een periode van 17 weken, met als startdatum 16 November en als einddatum 29 Maart 2016.

Dit document moet gelezen en goedgekeurd worden voordat het project daadwerkelijk wordt gestart.

5. Risicofactoren

Hieronder staat een opsomming van mogelijke problemen die tijdens dit project kunnen ontstaan, en hoe we deze problemen kunnen voorkomen of oplossen.

1. Door achterstanden op het gebied van kennis en vaardigheden is er meer tijd nodig om de stof zich eigen te maken. In dit geval moet zo veel mogelijk in eigen tijd verdiept worden in het eigen maken van de stof. Anderzijds in het hulp vragen aan medewerkers.

2. Doordat documenten of broncode beschadigd raken of verloren gaan moet werk opnieuw worden gedaan. De documentatie wordt veilig via de Cloud opgeslagen in Dropbox en de broncode wordt opgeslagen in Visual Studio Team Services.

3. De opdracht is niet goed geformuleerd of wordt verkeerd geïnterpreteerd. Dit veroorzaakt een slechte start van het project, waardoor een hoop werk overnieuw gedaan moet worden. Om dit te voorkomen worden er (tussentijdse) opleveringen van de applicatie gedaan om dit te ondervangen. Aan de andere kant zal de projectgroep de eisen van de opdrachtgever zo concreet mogelijk moeten formuleren, zodat er weinig misverstanden kunnen ontstaan.

4. Door inzet op een ander belangrijk project waardoor er minder tijd is om het project af te ronden. Door goede afspraken te maken met mijn opdrachtgever en een globale planning vooraf moet er tijd gereserveerd worden om mijn project af te ronden.

5. Doordat bij het publiceren van de App in de Google Play Store of Apple App Store geen goedkeuring wordt gegeven of te laat kan er vertraging oplopen op het publiceren van de App.

6. Door niet op tijd een Apple computer met xCode in bezit te hebben kan er vertraging worden opgelopen om de App in de Apple App store te krijgen. Het op tijd beschikbaar stellen van een Apple computer.

6. Kwaliteitsborging

6.1. Sprint review meeting

Elke twee weken vindt er een sprint review plaats samen met de product eigenaar. Het is mogelijk andere gebruikers uit te nodigen. Hierdoor kan feedback worden gegeven op het product zodat daarmee de kwaliteit wordt bevorderd.

6.2. Coding standaard

Er moet worden gehouden aan de code standaarden van FourICT. Hierdoor verbeterd de kwaliteit van de software. Hiervoor wordt gebruik gemaakt van een analyseer tool die deze controle uitvoert.

6.3. Risicoanalyse

Om de kwaliteit te waarborgen wordt er in de sprintplanning een risicoanalyse uitgevoerd per user story. Dit om de testaanpak en strategie te bepalen

7. Projectorganisatie

De projectgroep zal door mij worden uitgevoerd als medewerker in de organisatie. Binnen de projectgroep worden alle functies en verschillende rollen die daarbij komen kijken zelfstandig door mij uitgevoerd. Alleen mijn opdrachtgever is onderdeel van de projectorganisatie.

Naam	**Rol**	**Scrum Rol**
Mark A. Barto	Senior Software Engineer	Scrum master, Ontwikkelaar
Richard van Leeuwe	Opdrachtgever, Bedrijfsmentor	Producteigenaar
Henk van Ewijk	Opdrachtgever	Stakeholder

8. Wijze van rapporteren

Het project zal door middel van twee wekelijkse sprints worden uitgevoerd. Tijdens elke sprint zal het afstudeerverslag worden bijgewerkt met daarin toegelicht de werkzaamheden die zijn verricht, de planning, de problemen die zijn geconstateerd.

De productbacklog zal worden bijgehouden in Visual Studio Team Services.

Alle verslagen worden geschreven in Microsoft Word.

9. Benodigde mensen/ middelen

Natuurlijkerwijs zijn wij in dit project afhankelijk van onze opdrachtgever. Daarnaast zullen wij ook onze docenten van tijd tot tijd moeten raadplegen.

We hebben hierbij de volgende middelen nodig:

1. Laptop voor ontwikkeling voor thuis en op het werk met internet;
2. Dropbox, om documenten te bewaren;
3. Visual Studio Team Services, om broncode op te slaan;
4. Azure abonnement, voor publiceren van Api, SQL Services en beschikbaar stellen van webapplicatie;
5. Microsoft Word, om rapporten te schrijven;
6. Visual Studio Enterprise, om applicaties, UML-diagrammen mee te ontwikkelen;
7. Mac computer met Xcode, om iOS apps te ontwikkelen/publiceren in de app store;
8. Betaalde jaarlijkse abonnement bij Android en Apple om apps te publiceren;
9. Tablet met iOS/Android, om mobiele applicatie te testen;

10. Planning

Aangezien binnen scrum niet van de voren vaststaat welke userstories er worden uitgevoerd in de sprint, is er onderstaand een globale planning aangegeven van de werkzaamheden. De eerste twee weken worden gebruikt om de aanzet te maken naar de eerste sprint alvorens er wordt begonnen. Dit is een globale planning waar in de sprintplanning uiteindelijk bepaald wordt wat er in de volgende sprint aan werkzaamheden worden verricht.

DATUM	WERKZAAMHEDEN
16-11-2015 T/M 27-11-2015	Plan van Aanpak Opstellen user stories Opstellen detail planning Opstellen analysedocument Opzetten ontwikkelomgeving Opzetten gebruikte technieken Bijwerken afstudeerverslag
30-11-2015 T/M 11-12-2015	Sprint 1
14-12-2015 T/M 25-12-2015	Sprint 2
4-1-2016 T/M 15-1-2016	Sprint 3
18-1-2016 T/M 29-1-2016	Sprint 4
1-2-2016 T/M 12-2-2016	Sprint 5
15-2-2016 T/M 26-2-2016	Sprint 6
29-2-2016 T/M 11-3-2016	Sprint 7
11-3-2016 T/M 29-3-2016	Afronden afstudeerverslag

11. Beschrijving mijlpaalproducten

Aan te leveren producten:

1. Plan van aanpak.
2. Analyse document met daarin uitgewerkt de werking van het systeem in een sequence diagram, een class diagram, de procesbeschrijvingen, de functionele en niet-functionele eisen, de use case diagrammen inclusief beschrijvingen en tenslotte een deployment diagram;
3. Database uitgewerkt in een UML-class diagram, relationeel representatiemodel en relationeel implementatiemodel;
4. Een testplan met de uitwerking van de teststrategie;
5. Een testrapport met de uitwerking van de uitgevoerde testen;
6. Opleveren van het formulierensysteem op de OTAP-omgeving

Analyserapport

Ontwikkeling van een generiek formulierenregistratiesysteem bij FourICT B.V.

Opleiding: Informatica deeltijd

Begeleidend examinator: dhr. E.M. van Doorn

Tweede examinator: mevr. A.M.J.J Lousberg

Datum: 24-03-2016

Plaats: Zoetermeer

Versie: 1.0

Geschreven door: Mark Barto (12092452)

Versiebeheer

Versie	Datum	Omschrijving
0.1	20-11-2015	Initiale versie. Opzet systeem scope.
0.2	1-12-2015	Systeem scope gemaakt
0.3	10-12-2015	Document uitgebreid met usecases, userstories
0.4	20-12-2015	Usecases toegevoegd en userstories
0.5	28-2-2016	Deployment diagram
1.0	24-3-2016	Laatste aanpassingen

Inhoudsopgave

1. Inleiding

Dit rapport bevat de wensen en eisen van het generiek formulier registratiesysteem (GFRS) waarbij de wensen van het systeem beschreven worden. De requirements zijn gebaseerd op de interviews met stakeholders. Er zal eerst begonnen worden met de scope van het systeem te definiëren. Van de gemaakte requirements zijn user stories gemaakt en use cases ontworpen. Daarnaast is er een opsomming gemaakt van de actoren en requirements en zijn deze geprioriteerd. Verder gaat dit document ook in op het ontwerp van het datamodel.

2. Systeemscope

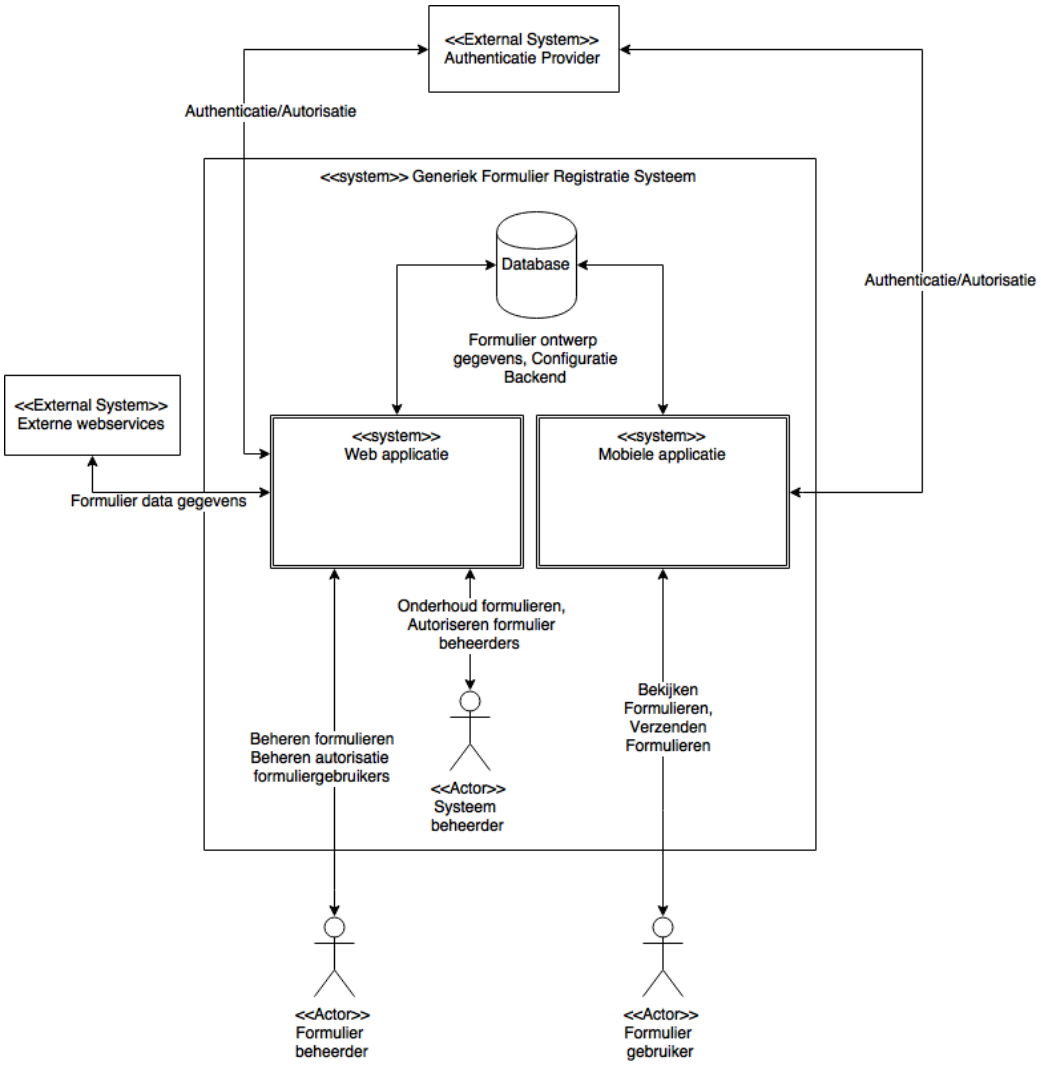
2.1. Omschrijving systeem

De belangrijkste functionaliteit van het systeem is het zelf ontwerpen van digitale formulieren en het toewijzen van formulieren aan gebruikers. De gebruiksvriendelijkheid van het systeem en daarnaast de grote flexibiliteit m.b.t. de integratie van het formulier rechtstreeks in de backoffice systeem van de klant is een kritieke succesfactor. Daarnaast wordt een hybride app beschikbaar gesteld waarop gebruikers zelf hun formulieren kunnen bekijken en versturen.

Voor de integratie van de data van het formulier zal dit alleen via e-mail beschikbaar zijn en niet via een maatwerk interface. Er zal geen gebruik gemaakt worden van specifieke onderdelen van het mobiele apparaat zoals de camera, gps. Het systeem zal niet geschikt zijn voor mobiele apparaten en is beschikbaar via internet. De hybride app zal alleen beschikbaar zijn voor mobiele apparaten waaronder iOS en Android.

2.2. Positionering systeem

Onderstaand context diagram geeft globaal de onderdelen die voorkomen in het systeem en daarnaast de externe entiteiten waar vanuit het systeem interactie mee is.



De formulier beheerder (actor) kan via een webapplicatie verbinding maken met het systeem en van daaruit beheer uitvoeren op zijn ontworpen formulieren en formulier gebruikers toegang geven. Daarnaast kan de formulier gebruiker (actor) via een hybride applicatie op zijn of haar device (bijvoorbeeld mobiel, tablet) verbinding maken met het systeem en van daaruit het formulier bekijken en verzenden. De systeembeheer (actor) kan onderhoud uitvoeren van de formulieren en autorisatie van formulier beheerders beheren.

Het interne systeem maakt gebruik van een externe authenticatie provider die de gebruikers autoriseert voor het gebruik van het systeem. Verder zorgt het systeem door middel van externe webservice (bijvoorbeeld e-mailservice, backoffice service) dat de data van de formulieren in de achterliggende systemen terecht komen.

De gegevens van het ontwerp van het formulier en de configuratie informatie wordt opgeslagen in de interne database.

3. Requirements

3.1. Functionele softwarerequirements

De volgende functionaliteiten zullen gerealiseerd worden voor het systeem:

- De gebruiker kan zich autoriseren voor het gebruik van het systeem.
- De gebruiker kan zijn eigen digitale formulier ontwerpen, invoervelden definiëren en restricties opgeven voor het valideren van invoervelden.
- De mogelijkheid om gebruikers toegang te geven tot het bekijken en versturen van het digitale formulier.
- Er moet bepaald kunnen worden op welke manier het digitale formulier wordt doorgezet. Standaard via de mail.
- De gebruiker kan zijn toegekende formulieren bekijken en versturen waarbij er validatie plaats vindt.
- Het formulier moet ook offline beschikbaar zijn zodat als er geen verbinding tot stand kan worden gebracht de gegevens lokaal worden opgeslagen en niet verloren gaan.
- Als de gebruiker onderweg is moet het digitale formulier beschikbaar zijn en verstuurd kunnen worden.

3.2. Niet-functionele softwarerequirements

Het systeem moet aan de volgende ISO 9126 standaard kwaliteitseisen voldoen:

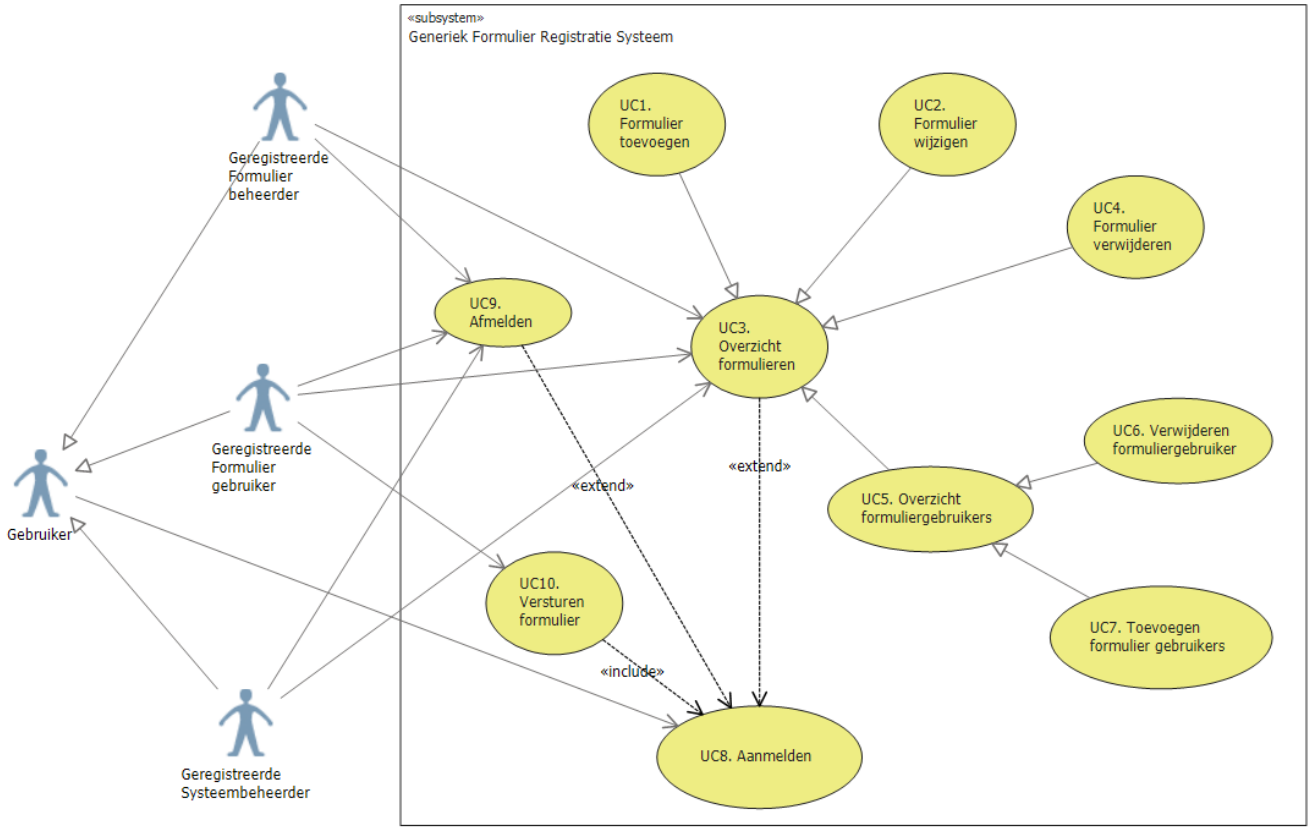
- Gebruiksvriendelijkheid: Het systeem dient eenvoudig in gebruik te zijn.
- Beveiligbaarheid: Er dient een goed rol-toegang model te worden geïmplementeerd. Daarnaast moet het systeem voldoen aan beveiligingseisen om datalekken tegen te gaan.
- Koppelbaarheid: Het systeem moet makkelijk kunnen integreren met verschillende leveranciers.
- Portabiliteit: Een generieke mobiele applicatie die beschikbaar komt voor verschillende mobiele apparaten die op verschillende platformen (iOS, Android) werkt.
- Portabiliteit: Een generiek formulierensysteem dat platformonafhankelijk is en toegankelijk is vanaf iedere computer, waarbij de klant zich kan aanmelden voor het gebruik van het systeem.
- Implementatie: De webapplicatie moet voldoen aan de FourICT userinterface.
- Onderhoudbaarheid: Moet voldoen aan de FourICT coding standaarden.
- Efficiëntie: De applicatie moet binnen < 1 seconden reageren.

3.3. User stories

#R	Titel	Sprint
1	Als formulierbeheerder wil ik op mijn computer mijn formulieren ontwerpen, zodat ik overal gebruik kan maken van het systeem.	1
2	Als formulierbeheerder wil ik mij registreren in het systeem, zodat ik toegang heb binnen het systeem.	1
3	Als formulierbeheerder wil ik mij aanmelden in het systeem, zodat ik toegang krijg tot het systeem.	1
4	Als geregistreerde formulierbeheerder wil ik een overzicht van mijn formulieren, zodat ik het juiste formulier kan bewerken.	2
5	Als geregistreerde formulierbeheerder wil ik een formulier kunnen invoeren met standaard invoervelden, zodat ik het formulier kan ontwerpen.	2
6	Als geregistreerde formulierbeheerder wil ik restricties kunnen opgeven in het formulier, zodat er minder fouten ontstaan.	2
7	Als geregistreerd formulierbeheerder wil ik het resultaat als e-mail ontvangen, zodat ik het formulier digitaal via e-mail ontvangt.	3
8	Als geregistreerde formulierbeheerder wil ik een formuliergebruiker toekennen aan mijn ontworpen formulier, zodat mijn medewerkers deze kunnen inzien.	3
9	Als geregistreerde formulierbeheerder wil ik formuliergebruikers toegang geven, zodat ik toegang kan verschaffen voor de app.	-
10	Als geregistreerde formulierbeheerder wil ik mij af kunnen melden, zodat ik niet meer ben aangemeld.	4
11	Als geregistreerde formulierbeheerder wil ik een formulier kunnen wijzigen, zodat ik het formulier kan beheren.	3
12	Als geregistreerde formulierbeheerder wil ik een formulier kunnen verwijderen, zodat ik formulier kan beheren.	4
13	Als geregistreerde formulierbeheerder wil ik formuliergebruikers verwijderen, zodat ik formuliergebruikers kan beheren.	4

#R	Titel	Sprint
14	Als formuliergebruiker wil ik op mijn mobiele apparaat mijn formulieren inzien, zodat als ik onderweg ben mijn formulieren kan bekijken en versturen.	5
15	Als formuliergebruiker wil ik met mijn mobiele apparaat kunnen aanmelden, zodat ik toegang krijg tot het systeem.	5
16	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat een overzicht van mijn formulieren, zodat ik een formulier kan selecteren.	5
17	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat het formulier invullen, zodat ik dit niet met pen en papier moet invullen.	6
18	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat het formulier versturen, zodat de formulierbeheerder het formulier digitaal aangeleverd krijgt.	6
19	Als geregistreerde formuliergebruiker wil ik met mijn mobiele apparaat een validatie melding, zodat ik mijn fouten kan herstellen in het formulier.	6
20	Als formuliergebruiker wil ik met mijn mobiele apparaat de app kunnen downloaden via de store, zodat ik gebruik kan maken van de app.	-
21	Als geregistreerde formuliergebruiker wil ik offline kunnen werken, zodat als ik onderweg ben het formulier later kan versturen.	7
22	Als geregistreerde formulierbeheerder wil ik het thema van het formulier aanpassen, zodat ik het mijn eigen huisstijl kan gebruiken.	-
23	Als geregistreerde formuliergebruiker wil ik foto's kunnen uploaden, zodat ik niet handmatig met een camera dit moet gaan doen.	-
24	Als geregistreerde formuliergebruiker wil ik automatisch het formulier laten doorsturen bij internetverbinding, zodat ik niet handmatig het formulier opnieuw moet sturen.	-
25	Als geregistreerde formuliergebruiker wil ik handtekening kunnen zetten in het formulier, zodat er bewijs is voor akkoord.	-
26	Als geregistreerde formuliergebruiker wil ik een preview zien van het ontworpen formulier, zodat ik direct kan zien hoe het formulier eruitziet. (sprint 2)	-
27	Als formuliergebruiker wil ik een overzicht van alle toegekende formulieren, zodat ik gebruikers makkelijk kan toekennen. (sprint 3)	-

3.4. Use case diagram



3.5. Opsomming actoren

Formuliergebruiker

Degene die het formulier bekijkt en verstuurt. Dit kan een gebruiker zijn die toegang heeft tot het formulier of zelf het formulier heeft ontworpen.

Formulierbeheerder

Een gebruiker die een formulier ontworpen heeft en die formulieren toewijst aan formulier gebruikers.

Systeembeheerder

Een beheerder die onderhoud uitvoert aan formulieren en autorisatie van gebruikers beheert.

3.6. Opsomming requirements

#UC	Use case	#R	Requirement	Bron
UC1	Formulier toevoegen	R5	Als geregistreerde formuliergebruiker wil ik een formulier kunnen invoeren met standaard invoervelden, zodat ik het formulier kan ontwerpen.	Interview
		R6	Als geregistreerde formuliergebruiker wil ik restricties kunnen opgeven in het formulier, zodat er minder fouten ontstaan.	Interview
UC2	Formulier wijzigen	R11	Als geregistreerde gebruiker wil ik een formulier kunnen wijzigen, zodat ik het formulier kan beheren.	Interview
UC3	Overzicht formulieren	R4	Als geregistreerde formuliergebruiker wil ik een overzicht van mijn formulieren, zodat ik het juiste formulier kan bewerken.	Interview
		R16	Als formuliergebruiker wil ik mij aanmelden in het systeem, zodat ik toegang krijg tot de app.	Interview
UC4	Formulier verwijderen	R12	Als geregistreerde gebruiker wil ik een formulier kunnen verwijderen, zodat ik formulier kan beheren.	Interview
UC5	Overzicht formuliergebruikers	R8	Als geregistreerde formuliergebruiker wil ik medewerkers toekennen aan mijn ontworpen formulier, zodat mijn medewerkers deze kunnen inzien.	Interview
UC6	Verwijderen formuliergebruikers	R13	Als geregistreerde gebruiker wil ik medewerkers verwijderen, zodat ik medewerkers kan beheren.	Interview
UC7	Toevoegen formuliergebruiker	R9	Als geregistreerde formuliergebruiker wil ik medewerkers toegang geven met	Interview

			e-mail en wachtwoord, zodat ik toegang kan verschaffen voor de app.	
UC8	Aanmelden	R2	Als formuliergebruiker wil ik mij registreren in het systeem, zodat ik toegang heb binnen het systeem.	Interview
		R3	Als formuliergebruiker wil ik mij aanmelden, zodat ik toegang krijg tot het systeem.	Interview
		R15	Als formuliergebruiker wil ik mij aanmelden, zodat ik toegang krijg tot de app.	Interview
UC9	Afmelden	R10	Als geregistreerde gebruiker wil ik mij af kunnen melden, zodat ik niet meer ben aangemeld.	Interview
UC10	Versturen formulier	R7	Als geregistreerde formuliergebruiker wil ik het resultaat als e-mail ontvangen, zodat ik het formulier digitaal via e-mail ontvangt.	Interview
		R17	Als geregistreerde formuliergebruiker wil ik het formulier invullen, zodat ik dit niet met pen en papier moet invullen.	Interview
		R18	Als geregistreerde formuliergebruiker wil ik het formulier versturen, zodat de formulierbeheerder het formulier digitaal aangeleverd krijgt.	Interview
		R19	Als geregistreerde formuliergebruiker wil ik een validatie melding, zodat ik mijn fouten kan herstellen in het formulier.	Interview
		R21	Als geregistreerde formuliergebruiker wil ik offline kunnen werken, zodat als ik onderweg ben het formulier later kan versturen.	Interview

3.7. Use case beschrijvingen

3.7.1. Beschrijving 'Toevoegen formulier'

ID	UC1
Naam	Toevoegen formulier
Samenvatting	Actor voegt nieuwe formulier toe in het systeem
Actoren	Formulierbeheerder, systeembeheerder, formuliergebruiker
Aannames	▪ Actor heeft zich aangemeld ▪ Actor bevindt zich in het formulier overzicht in het systeem
Beschrijving	1. Actor kiest voor 'Toevoegen'. 2. Systeem toont formulier met invoervelden voor een nieuw formulier [1]. 3. Systeem haalt alle veldtypes op en toont deze in de selectiebox op het formulier. 4. Systeem haalt de gerelateerde velden op en toont deze velden in de lijst. [2][3] 5. Actor vult naam, titel, omschrijving van het nieuwe formulier in. 6. Actor selecteert veldtype om toe te voegen (input, checkbox, select, textarea) 7. Actor kiest voor 'Toevoegen veld' 8. Systeem haalt template behorend bij het geselecteerde veldtype op. 9. Systeem toont de veld eigenschappen van de template met invoervelden. 10. Actor kiest voor 'Opslaan'. 11. Systeem controleert invoer. 12. Systeem voegt formulier toe aan database. 13. Systeem sluit het formulier en ververst het overzicht van formulieren.
Uitzonderingen / excepties	▪ Naam is verplicht, actor krijgt melding.
Resultaten	▪ Formulier is toegevoegd in het systeem met behorende veldtypes en eigenschappen..
Alternatieve flow	[1] Actor gebruikt de knop 'Annuleren' en keert terug naar het formulier overzicht [2] Actor gebruikt de knop 'Wijzigen' en kan de eigenschappen van het veld wijzigen in het formulier. [3] Actor gebruikt de knop 'Verwijderen' en verwijdert het veld uit de lijst.

3.7.2. Beschrijving 'Wijzigen formulier'

ID	UC2
Naam	Wijzigen formulier
Samenvatting	Actor wijzigt formulier in het systeem
Actoren	Formulierbeheerder, systeembeheerder

Aannames	▪ Actor heeft zich aangemeld ▪ Actor bevindt zich in het formulier overzicht in het systeem
Beschrijving	1. Actor selecteert onder acties voor 'Wijzigen' naast het formulier. 2. Systeem toont formulier met invoervelden met daarin de huidige gegevens van het formulier met behorende veldtypes en eigenschappen [1] 3. Actor wijzigt een of meer van de gegevens [1] [2] [3]. 4. Actor kiest voor 'Opslaan'. 5. Systeem wijzigt formulier in database. 6. Systeem sluit het formulier en ververst het overzicht van formulieren.
Uitzonderingen / excepties	▪ Naam is verplicht, actor krijgt melding. ▪ Formulierbeheerder is geen eigenaar van formulier, actor krijgt melding.
Resultaten	Formulier is gewijzigd in het systeem.
Alternatieve flow	[1] Actor gebruikt de knop 'Annuleren' en keert terug naar het formulier overzicht [2] Actor gebruikt de knop 'Wijzigen' en kan de eigenschappen van het veld wijzigen in het formulier. [3] Actor gebruikt de knop 'Verwijderen' en verwijdert het veld uit de lijst.

3.7.3. Beschrijving 'Tonen overzicht formulieren'

ID	UC3
Naam	Tonen overzicht formulieren
Samenvatting	Als actor het tonen van een overzicht van de formulieren
Actoren	Formulierbeheerder, formuliergebruiker, systeembeheerder
Aannames	▪ Actor heeft zich aangemeld
Beschrijving	1. Actor kiest in het menu voor 'Formulieren'. 2. Systeem toont op basis van actor andere overzichten: a. Als actor systeembeheerder worden alle formulieren getoond; b. Als actor formulierbeheerder/gebruiker worden formulieren getoond waar hij/zij eigenaar van is of toegang hebt gekregen.
Uitzonderingen / excepties	▪ Er zijn geen formulieren gevonden, actor krijgt melding.
Resultaten	Systeem toont formulieren op basis van actor.
Alternatieve flow	

3.7.4. Beschrijving 'Verwijderen formulier'

ID	UC4
Naam	verwijderen formulier
Samenvatting	Als actor het formulier verwijderen uit het systeem

Actoren	Formulierbeheerder, systeembeheerder
Aannames	▪ Actor heeft zich aangemeld ▪ Actor bevindt zich in het formulier overzicht in het systeem
Beschrijving	1. Actor selecteert onder acties voor 'Verwijderen' naast het formulier. [1] 2. Systeem verwijdert formulier met gerelateerde velden en eigenschapen. 3. Systeem verwijdert toegekende formuliergebruikers uit de metadata. 4. Systeem geeft melding dat het verwijderen is gelukt. 5. Systeem ververst het overzicht het formulier.
Uitzonderingen / excepties	▪ Formulier is niet gevonden, actor krijgt melding. ▪ Actor is geen eigenaar van formulier, actor krijgt melding.
Resultaten	▪ Formulier is verwijderd.
Alternatieve flow	

3.7.5. Beschrijving 'Tonen formuliergebruiker'

ID	UC5
Naam	Tonen formuliergebruikers aan formulier
Samenvatting	Als actor het bekijken van toegekende formuliergebruikers aan een formulier.
Actoren	Formulierbeheerder, systeembeheerder
Aannames	▪ Actor heeft zich aangemeld ▪ Actor bevindt zich in het formulier overzicht in het systeem
Beschrijving	1. Actor selecteert onder acties voor 'Toekennen' naast het formulier. 2. Systeem zoekt in metadata van de formuliergebruikers en geeft overzicht van alle toegekende formuliergebruikers [1] 3. Systeem toont formuliergebruikers.
Uitzonderingen / excepties	▪ Er zijn geen toegekende formuliergebruikers, actor krijgt melding.
Resultaten	Systeem toont formuliergebruikers van formulier
Alternatieve flow	

3.7.6. Beschrijving 'Toekennen formuliergebruiker'

ID	UC6
Naam	Toekennen formuliergebruiker aan formulier
Samenvatting	Als actor het toekennen van een formuliergebruiker aan een formulier.
Actoren	Formulierbeheerder, systeembeheerder
Aannames	▪ Actor heeft zich aangemeld ▪ Actor bevindt zich in het overzicht van toegekende formuliergebruikers in het systeem
Beschrijving	1. Actor vult e-mailadres in van formuliergebruiker.

	2. Actor kiest voor 'Toevoegen'. 3. Systeem controleert invoer. 4. Systeem voegt formulier toe aan metadata van formuliergebruiker. [1] 5. Systeem geeft melding dat de gebruiker is toegekend. 6. Systeem ververst het overzicht van toegekende formuliergebruikers.
Uitzonderingen / excepties	▪ Er zijn geen toegekende formuliergebruikers, actor krijgt melding. ▪ E-mailadres klopt niet, actor krijgt melding. ▪ Actor is eigenaar van formulier, actor krijgt melding. ▪ Toegekende formuliergebruiker is reeds toegekend, actor krijgt melding.
Resultaten	▪ Formulier is toegevoegd aan metadata van formuliergebruiker.
Alternatieve flow	[1] Als formuliergebruiker niet bestaat word deze automatisch aangemaakt en krijgt hij mail om zijn e-mailadres te verifiëren voordat hij gebruik kan maken van het formulier.

3.7.7. Beschrijving 'Verwijderen formuliergebruiker'

ID	UC7
Naam	verwijderen formuliergebruiker van formulier
Samenvatting	Als actor de formuliergebruiker toegang ontzeggen van het bekijken van het formulier.
Actoren	Formulierbeheerder, systeembeheerder
Aannames	▪ Actor heeft zich aangemeld ▪ Actor bevindt zich in het formulier overzicht in het systeem
Beschrijving	6. Actor selecteert onder acties voor 'Toekennen' naast het formulier. 7. Systeem geeft overzicht van alle toegekende formuliergebruikers [1] 8. Actor kiest voor 'Verwijderen' in de lijst naast toegekende formuliergebruiker. 9. Systeem haalt metadata op van de formuliergebruiker en verwijdert de toekenning. [2] 10. Systeem geeft melding dat de gebruiker is toegekend. 11. Systeem ververst het overzicht van toegekende formuliergebruikers.
Uitzonderingen / excepties	▪ Actor is geen eigenaar van formulier, actor krijgt melding. ▪ Geselecteerde formuliergebruiker is niet toegekend, actor krijgt melding. ▪ Geselecteerder formuliergebruiker is niet gevonden, actor krijgt melding.
Resultaten	▪ Formulier is toegevoegd aan metadata van formuliergebruiker.
Alternatieve flow	[1] Actor gebruikt de knop 'Annuleren' en keert terug naar het formulier overzicht [2] Als formuliergebruiker niet bestaat word deze automatisch aangemaakt en krijgt hij mail om zijn e-mailadres te verifiëren voordat hij gebruik kan maken van het formulier.

3.7.8. Beschrijving 'Aanmelden'

ID	UC8
Naam	Aanmelden
Samenvatting	Actor moet zich aanmelden op het systeem
Actoren	Formulierbeheerder, systeembeheerder, formuliergebruiker
Aannames	▪ Actor is niet aangemeld
Beschrijving	1. Systeem opent aanmeldscherm. 2. Actor geeft e-mailadres op. 3. Systeem controleert autorisatie [1] 4. Systeem stuurt e-mail met verificatie code 5. Systeem vraagt om verificatie code om aan te melden. 6. Actor geeft verificatie code op. 7. Systeem slaat JWT op. 8. Systeem toont welkom scherm
Uitzonderingen / excepties	▪ E-mailadres klopt niet, actor krijgt melding. ▪ Autorisatiecode komt niet overeen, actor krijgt melding.
Resultaten	Actor is aangemeld op het systeem
Alternatieve flow	[1] Als actor niet bestaat word deze automatisch aangemaakt en krijgt hij mail om zijn e-mailadres te verifiëren voordat hij gebruik kan maken van het formulier.

3.7.9. Beschrijving 'Afmelden'

ID	UC9
Naam	Afmelden
Samenvatting	Actor moet zich afmelden van het systeem
Actoren	Formulierbeheerder, systeembeheerder, formuliergebruiker
Aannames	▪ Actor is aangemeld
Beschrijving	1. Actor selecteert 'Afmelden'. 2. Systeem verwijdert JWT
Uitzonderingen / excepties	
Resultaten	Actor is afgemeld van het systeem
Alternatieve flow	

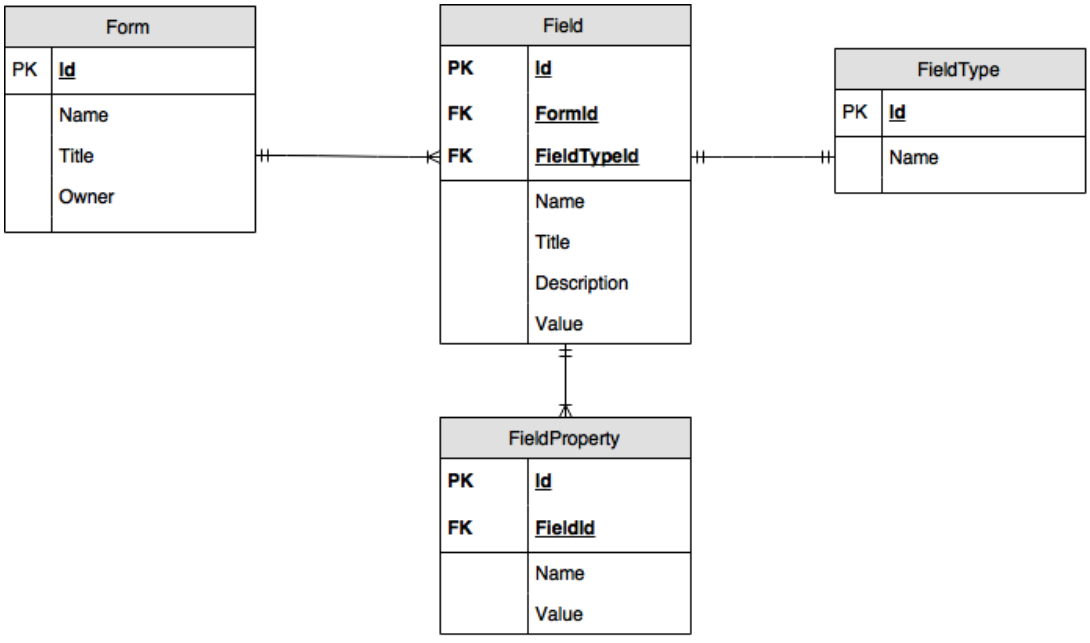
3.7.10. Beschrijving 'Versturen formulier'

ID	UC10

Naam	Versturen formulier
Samenvatting	Actor verstuurt formulier
Actoren	Formuliergebruiker
Aannames	▪ Actor is aangemeld in de mobiele applicatie
Beschrijving	1. Actor selecteert tabblad 'Formulieren'. 2. Systeem toont lijst met toegewezen en ontworpen formulieren. 3. Actor selecteert formulier. 4. Systeem haalt template op basis van geselecteerd formulier op. 5. Systeem toont formulier met invoervelden zoals deze is ontworpen. 6. Actor vult invoervelden in. 7. Actor klik op 'Versturen' 8. Systeem controleert invoer 9. Systeem verstuurt ingevulde formulier gegevens naar e-mailadres van eigenaar. [1] 10. Systeem toont melding dat het formulier is opgestuurd. 11. Systeem gaat terug naar de lijst met toegekende formulieren.
Uitzonderingen / excepties	▪ Er zijn geen formulieren gevonden, actor krijgt melding. ▪ Autorisatiecode komt niet overeen, actor krijgt melding. ▪ Verplichte velden zijn niet ingevuld, actor krijgt validatie melding
Resultaten	Actor is aangemeld op het systeem
Alternatieve flow	[1] Als er geen internet verbinding is wordt het formulier lokaal opgeslagen.

4. Datamodel

4.1. Class diagram



4.2. Modeldictionary

4.2.1. Tabel 'Form'

Naam	Type	Inhoud	Null	Lengte	Beschrijving	Opmerking
Id	uniqueidentifier	D19FFB23-A2BA-E511-9BEE-54271EAC3402	N		Unieke identificatie	PK
Owner	varchar	email|56963839f609ee1adad525ce	N	max	Unieke identificatie van gebruiker uit auth0 wie de eigenaar is van het formulier.	FK naar Form
Name	varchar	Contact	N	50	Naam van formulier	
Title	varchar	Contact formulier	Y	200	Titel van formulier	
Description	varchar	Via dit formulier kunt u een bericht sturen	Y	max	Omschrijving van formulier	

4.2.2. Tabel 'Field'

Naam	Type	Inhoud	Null	Lengte	Beschrijving	Opmerking
Id	uniqueidentifier	5BA851CF-B8BA-E511-9BEE-54271EAC3402	N		Unieke identificatie	PK
FormId	uniqueidentifier	D19FFB23-A2BA-E511-9BEE-54271EAC3402	N		Unieke identificatie naar formulier	FK naar Form
FieldTypeId	uniqueidentifier	75585D25-11BA-E511-	N		Unieke identificatie naar veldtype	FK naar FieldType

		9BEE-54271EAC3402				
Name	varchar	Naam	N	50	Naam die gebruikt word bij het versturen van formulier	
Title	varchar	Uw naam	Y	200	Titel van het veld die als label dient in het formulier	
Required	bit	0, 1	Y	1	Is het veld verplicht om ingevuld te worden?	
Description	varchar	Geef hier uw naam op	Y	max	Omschrijving van veld	

4.2.3. Tabel 'FieldProperty'

Naam	**Type**	**Inhoud**	**Null**	**Lengte**	**Beschrijving**	**Opmerking**
Id	uniqueidentifier	61DBBA1B-C5BA-E511-9BEE-54271EAC3402	N		Unieke identificatie	PK
FieldId	uniqueidentifier	5BA851CF-B8BA-E511-9BEE-54271EAC3402	N		Unieke identificatie van veld	FK naar Field
Name	varchar	Options, placeholder	N	50	Naam van eigenschap van veld	
Value	varchar	[{"name": "mark","value": "barto"},{"name":"dennis", "value":"huussen"}], Vul hier uw naam in	Y	max	Waarde van eigenschap van veld.	Dit veld kan een JSON string bevatten in het geval van het vullen van een selectiebox

4.2.4. Tabel 'FieldType'

Naam	Type	Inhoud	Null	Lengte	Beschrijving	Opmerking
Id	uniqueidentifier	75585D25-11BA-E511-9BEE-54271EAC3402	N		Unieke identificatie	PK
Name	`varchar	Input, checkbox, select, textarea	N	50	Naam van type veld	

4.3. Relationeel representatiemodel (RRM)

Form (Id, Owner, Name, Title)

- Als een formulier (Form) verwijderd wordt moeten velden (Field) ook met dezelfde FormId verwijderd worden, constraint op tabel niveau.

FieldType (Id, Name)

- Een veldtype (FieldType) mag alleen verwijderd worden als er geen velden (Field) met hetzelfde Id bestaat, constraint op tabel niveau.

Field (Id, *FormId*, *FieldTypeId*, Name, Title, Description)

- FormId is een vreemde sleutel en verwijst naar Id in Form, FormId mag niet NULL zijn.
- FieldTypeId is een vreemde sleutel en verwijst naar Id in FieldType, FieldTypeId mag niet NULL zijn.
- Als een formulier (Form) verwijderd wordt moeten de velden (Field) ook met dezelfde FormId verwijderd worden, constraint op tabel niveau.

FieldProperty (Id, *FieldId*, Name, Value)

- FieldId is een vreemde sleutel en verwijst naar Id in Field, FieldId
- Als een veld (Field) verwijderd wordt moeten veld eigenschappen (FieldProperty) ook met dezelfde FieldId verwijderd worden, constraint op tabel niveau.

4.4. Relationeel implementatiemodel (RIM)

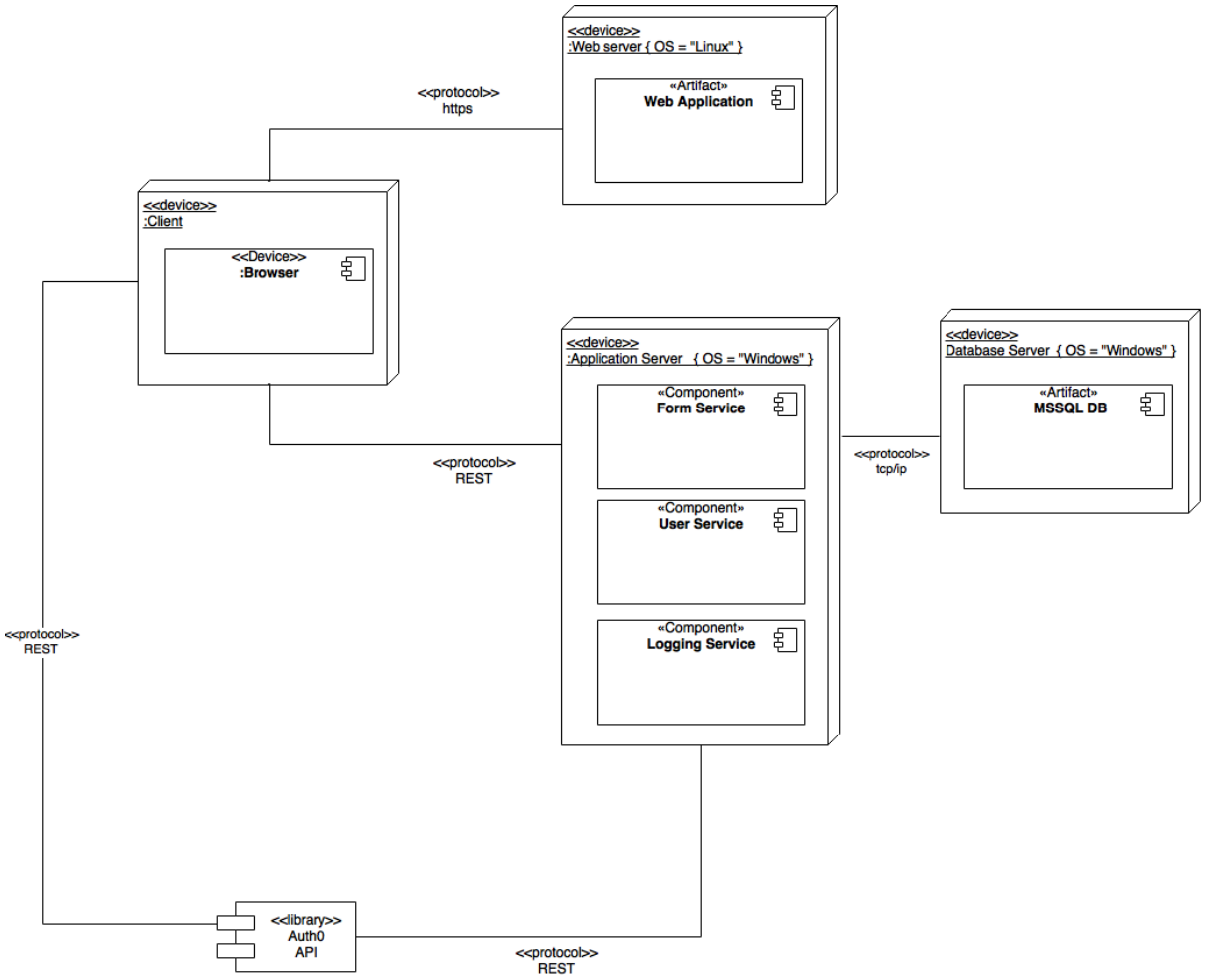
```
CREATE TABLE Form (
        Id UNIQUEIDENTIFIER NOT NULL DEFAULT NEWID(),
        Owner VARCHAR(MAX)  NOT NULL,
        Name VARCHAR(50)  NOT NULL,
        Title VARCHAR(200)  NULL,
        PRIMARY KEY (Id)
)

CREATE TABLE FieldType (
        Id UNIQUEIDENTIFIER NOT NULL DEFAULT NEWID(),
        Name VARCHAR(50)  NOT NULL,
        PRIMARY KEY (Id)
)

CREATE TABLE Field (
        Id UNIQUEIDENTIFIER NOT NULL DEFAULT NEWID(),
        FormId UNIQUEIDENTIFIER NOT NULL,
        FieldTypeId  UNIQUEIDENTIFIER NOT NULL,
        Owner VARCHAR(MAX)  NOT NULL,
        Name VARCHAR(50)  NOT NULL,
        Title VARCHAR(200)  NULL,
        Description VARCHAR(MAX)  NULL,
        PRIMARY KEY (Id),
        CONSTRAINT FK_Form FOREIGN KEY(FormId) REFERENCES Form(Id)
                ON UPDATE CASCADE
                ON DELETE CASCADE,
        CONSTRAINT FK_Field FOREIGN KEY(FieldTypeId) REFERENCES FieldType(Id)
                ON UPDATE CASCADE
                ON DELETE NO ACTION
)

CREATE TABLE FieldProperty (
        Id UNIQUEIDENTIFIER NOT NULL DEFAULT NEWID(),
        FieldId  UNIQUEIDENTIFIER NOT NULL,
        Name VARCHAR(50)  NOT NULL,
        Value  VARCHAR(MAX)  NULL,
        PRIMARY KEY (Id),
        FOREIGN KEY (FieldId) REFERENCES Field(Id)
                ON UPDATE CASCADE
                ON DELETE CASCADE
)
```

5. Deployment diagram



6. Definities van begrippen

Begrip	Definitie
Actor	Een actor definieert een rol die een gebruiker kan spelen die met het systeem werkt. Een gebruiker kan een persoon zijn of een ander systeem. Actoren hebben een naam en een korte omschrijving en worden geassocieerd met de use cases waarmee ze interactie hebben.
Alternatieve stroom	Een variatie op de basisstroom van een use-case beschrijving. Er kunnen meerdere alternatieve stromen zijn.
Gebruiker	Een belanghebbende die met het systeem werkt om zijn doelen te behalen.
Requirements	Wat het softwaresysteem moet doen om te voorzien in de behoeften van de belanghebbenden.
Use Case	Een use case omvat alle manieren waarop het systeem gebruikt kan worden om een bepaald doel voor een bepaalde gebruiker te behalen.
Use-Case Diagram	Een diagram waarop een aantal actoren, use cases en hun relaties inzichtelijk wordt gemaakt
Use-Case Beschrijving	Een beschrijving van een use case waarmee wordt verteld hoe het systeem en zijn actoren samenwerken om een bepaald doel te behalen. Het bevat een volgorde van handelingen en varianten daarop, die een systeem en zijn actoren kunnen uitvoeren om een bepaald doel te behalen
JWT	Een JWT staat voor JSON Web token en word gebruikt om tussen twee verschillende partijen veilig te kunnen communiceren. Deze wordt gebruikt voor autorisatie en bevat speciale eigenschappen om de identiteit vast te leggen.
Verificatiecode	Een unieke code bestaande uit cijfers voor verificatie.
GFRS	Generiek Formulier Registratie Systeem, naam van het systeem

Geraadpleegde literatuur

de Swart, N. (2010). *Handboek requirements: brug tussen business en ict.* Eburon Business.

Testplan

Ontwikkeling van een generiek formulierenregistratiesysteem bij FourICT B.V.

Opleiding: Informatica deeltijd

Begeleidend examinator: dhr. E.M. van Doorn

Tweede examinator: mevr. A.M.J.J Lousberg

Datum: 24-03-2016

Plaats: Zoetermeer

Versie: 1.0

Geschreven door: Mark Barto (12092452)

Documenthistorie

Versie	Datum	Omschrijving
0.1	20-11-2015	Initiale versie
1.0	24-03-2016	Laatste aanpassingen.

Inhoudsopgave

1. Inleiding

1.1. Doel van het testplan

Dit document beschrijft de testaanpak die gebruikt zal worden voor het testen van het generiek formulier registratiesysteem. Ten eerste wordt gekeken naar de testbasis en de omgeving. Daarna wordt de strategie bepaald en de aanpak. Uiteindelijk worden de tools beschreven die gebruikt gaan worden om te testen.

2. Basis voor het testplan

De volgende documenten zijn gebruikt als basis voor dit testplan.

DOCUMENTNAAM	VERSIE	DATUM	AUTEUR
PLAN VAN AANPAK	v1.0	24-03-2016	Mark Barto

3. Testbasis

De testbasis bestaat uit die documenten waaruit de testgevallen worden afgeleid. De testbasis bevat de documentatie die als basis dient voor de uit te voeren tests.

DOCUMENTNAAM	VERSIE	DATUM	AUTEUR
ANALYSE DOCUMENT	v1.0	24-03-2016	Mark Barto

4. Teststrategie

De beschikbare tijd om te testen is beperkt; niet alles kan even zwaar worden getest. Dus moesten er keuzes worden gemaakt. Daarbij is ernaar gestreefd om de testcapaciteit zo effectief en efficiënt mogelijk over het totale testtraject te verdelen.

De teststrategie legt vast *wat* er *met welke zwaarte wanneer* getest gaat worden en is erop gericht om zo vroeg mogelijk de belangrijkste fouten te vinden tegen de minste kosten, dus met optimaal gebruik van de beschikbare capaciteit en tijd.

De eerste stap bij het opstellen van de teststrategie was het uitvoeren van een *risicoanalyse*. De resultaten van die productrisicoanalyse (PRA) zijn in bijlage 1 weergegeven.

In samenspraak met de product eigenaar is bepaald wat het risico is. Het risico wordt bepaald aan de hand van de verwachte schade en de faalkans dat het kan optreden. De schade wordt in overleg met de producteigenaar bepaald en de faalkans door het ontwikkelteam. Uiteindelijk wordt de risico gebruikt per kwaliteitsattribuut om de zwaarte van de test te bepalen. Bij zwaarte gemiddeld en hoog worden testcases opgezet met logische

en fysieke testgevallen in Visual Studio Team Services (VSTS). Bij laag wordt het getest op exploratieve wijze.

5. Testaanpak

Op basis van de productrisicoanalye (PRA) wordt de teststrategie opgesteld en vertaald naar een concrete testaanpak (het hoe).

#	KENMERK	RISICOKLASSE	ZWAARTE
UC1	**Functionaliteit**	A	●●●
	Gebruiksvriendelijkheid	B	●●
	Continuïteit	B	●●
UC2	**Functionaliteit**	B	●●
UC3	**Functionaliteit**	B	●●
	Performance	C	●
	Continuïteit	B	●●
	Gebruiksvriendelijkheid	C	●
	Portabiliteit	B	●●
UC4	**Functionaliteit**	C	●
UC5	**Functionaliteit**	B	●●
UC6	**Functionaliteit**	C	●
UC7	**Functionaliteit**	B	●●
UC8	**Functionaliteit**	B	●●
	Beveiliging	B	●●
UC9	**Functionaliteit**	C	●
UC10	**Functionaliteit**	A	●●●
	Performance	C	●
	Inpasbaarheid	B	●●
	Gebruiksvriendelijkheid	B	●●
	Portabiliteit	A	●●●

● Er worden geen testcases opgesteld maar er wordt getest op exploratieve wijze.
●● Er worden globale test cases opgesteld.
●●● Er worden gedetailleerde test cases opgesteld.

6. Exit criteria

- Er moet een testcase per userstory zijn opgezet.
- De testcase moet zijn uitgevoerd.
- De testresultaten zijn geregistreerd.
- Als er fouten zijn geconstateerd moeten deze zijn opgelost en er moet een her-test hebben plaats gevonden.

7. Testomgeving

FourICT heeft de beschikking over een OTAP-omgeving:

- Ontwikkel
- Test
- Acceptatie
- Productie

8. Testproducten

DOCUMENTNAAM	VERSIE	DATUM	AUTEUR
TESTPLAN	v1.0	24-03-2016	Mark Barto
TESTRAPPORT	v1.0	24-03-2016	Mark Barto

9. Testtools

De bevindingen worden vastgelegd in Visual Studio Team Services (VSTS) waarin voor een constatering een bug wordt aangemaakt met een omschrijving en prioriteit. Daarnaast wordt gebruik gemaakt van VSTS om test cases op te zetten en logische en fysieke testgevallen vast te leggen.

10. Definities van begrippen

Begrip	Definitie
PRA	Productrisicoanalyse

Bijlage 1 – productrisicotabel

TESTDOEL

UC1: FORMULIER TOEVOEGEN	Het ontwerpen van een formulier
UC2: FORMULIER WIJZIGEN	Het wijzigen van een ontworpen formulier
UC3: OVERZICHT FORMULIEREN	Het tonen van een overzicht van een formulier.
UC4: FORMULIER VERWIJDEREN	Het verwijderen van een formulier.
UC5: OVERZICHT FORMULIER GEBRUIKERS	Het tonen van een overzicht van de gebruikers die toegang hebben tot het formulier.
UC6: VERWIJDEREN FORMULIER GEBRUIKERS	De toegang ontzeggen tot een formulier van de gebruiker.
UC7: TOEVOEGEN FORMULIER GEBRUIKERS	Toegang geven tot een formulier van de gebruiker.
UC8: AANMELDEN	Het aanmelden en registreren van een gebruiker.
UC9: AFMELDEN	Het afmelden van een gebruiker.
UC10: VERSTUREN FORMULIER	Het gebruiken en versturen van een ontworpen formulier .

Bijlage 2 – productrisicoanalye

#	KENMERK	SCHADE	FAALKANS	RISICOKLASSE	ZWAARTE
UC1	**Functionaliteit**	H	H	A	●●●
	Gebruiksvriendelijkheid	M	H	B	●●
	Continuïteit	H	L	B	●●
UC2	**Functionaliteit**	M	H	B	●●
UC3	**Functionaliteit**	H	L	B	●●
	Performance	L	L	C	●
	Continuïteit	H	L	B	●●
	Gebruiksvriendelijkheid	M	L	C	●
	Portabiliteit	H	L	B	●●
UC4	**Functionaliteit**	L	L	C	●
UC5	**Functionaliteit**	M	M	B	●●
UC6	**Functionaliteit**	L	L	C	●
UC7	**Functionaliteit**	L	H	B	●●
UC8	**Functionaliteit**	H	L	B	●●
	Beveiliging	H	L	B	●●
UC9	**Functionaliteit**	L	L	C	●●
UC10	**Functionaliteit**	H	H	A	●●●
	Performance	M	L	C	●
	Inpasbaarheid	M	M	B	●●
	Gebruiksvriendelijkheid	M	H	B	●●
	Portabiliteit	H	H	A	●●●

Testrapport

Ontwikkeling van een generiek formulierenregistratiesysteem bij FourICT B.V.

Opleiding: Informatica deeltijd

Begeleidend examinator: dhr. E.M. van Doorn

Tweede examinator: mevr. A.M.J.J Lousberg

Datum: 24-03-2016

Plaats: Zoetermeer

Versie: 1.0

Geschreven door: Mark Barto (12092452)

Documenthistorie

Versie	Datum	Omschrijving
0.1	20-11-2015	Initiale versie
1.0	24-03-2016	Laatste aanpassingen.

Inhoudsopgave

1. Resultaat uitgevoerde tests

1.1. Testgevallen

ID #	Beschrijving	Sprint	Use case
TC1	Aanmelden zonder e-mail adres.	1, 5	UC8
TC2	Aanmelden met onjuist e-mail adres.	1, 5	UC8
TC3	Aanmelden met e-mail adres die niet gevonden kan worden in het systeem.	1, 5	UC8
TC4	Aanmelden zonder verificatie code .	1,5	UC8
TC5	Aanmelden met onjuist verificatie code .	1, 5	UC8
TC6	Aanmelden met juiste gegevens.	1, 5	UC8
TC7	Toevoegen formulier zonder alle velden ingevuld	2	UC1
TC8	Toevoegen formulier zonder ‘naam’	2	UC1
TC9	Toevoegen formulier zonder velden	2	UC1
TC10	Toevoegen formulier met veld ‘input’	2	UC1
TC11	Toevoegen formulier met veld ‘input’ zonder ‘naam’	2	UC1
TC12	Toevoegen formulier met veld ‘input’ alle velden ingevuld.	2	UC1
TC13	Toevoegen formulier met veld ‘checkbox’ zonder ‘naam’	2	UC1
TC14	Toevoegen formulier met veld ‘checkbox’ alle velden ingevuld.	2	UC1
TC15	Veld ‘checkbox’ verwijderen in toegevoegd formulier	2	UC1
TC16	Toevoegen formulier met veld ‘select’ zonder ‘naam’	2	UC1
TC17	Toevoegen formulier met veld ‘select’ met opties	2	UC1
TC18	Toevoegen formulier met veld ‘select’ met verwijderde optie	2	UC1
TC19	Toevoegen formulier met veld ‘select’ alle velden ingevuld.	2	UC1
TC20	Veld ‘select’ verwijderen in toegevoegd formulier	2	UC1
TC21	Toevoegen formulier met veld ‘textarea’ zonder ‘naam’	2	UC1

TC22	Toevoegen formulier met veld ‘textarea’ alle velden ingevuld.	2	UC1
TC23	Annuleren van toegevoegde formulier	2	UC1
TC24	Meerdere velden (‘select en ‘input’) toevoegen in formulier	2	UC1
TC25	Formulier wordt getoond in de lijst na toevoegen van formulier	2, 5	UC3
TC26	Alle formulieren worden getoond bij rol ‘systeembeheerder’	2, 5	UC3
TC27	Alleen formulieren worden getoond die de formuliergebruiker heeft aangemaakt.	2, 5	UC3
TC28	Toekennen gebruiker met juiste gegevens	3	UC6
TC29	Toekennen gebruiker als eigenaar van formulier	3	UC6
TC30	Toekennen gebruiker met verkeerd e-mail adres	3	UC6
TC31	Toekennen gebruiker die reeds is toegekend	3	UC6
TC32	Verwijderen toegekende gebruiker.	4	UC7
TC33	Verwijderen toegekende gebruiker zonder dat de gebruiker is toegekend.	4	UC7
TC34	Verwijderen toegekende gebruiker waarbij gebruiker niet de eigenaar is van het formulier.	4	UC7
TC35	Verwijderen toegekende gebruiker als systeembeheerder.	4	UC7
TC36	Formulier versturen met ‘input’ zonder verplicht veld.	6	UC10
TC37	Formulier versturen met ‘input’ als verplicht veld.	6	UC10
TC38	Formulier versturen met ‘checkbox’ niet als verplicht veld.	6	UC10
TC39	Formulier versturen met ‘checkbox’ als verplicht veld.	6	UC10
TC40	Formulier versturen met ‘select’ en opties niet als verplicht veld .	6	UC10
TC41	Formulier versturen met ‘select’ en opties als verplicht veld.	6	UC10
TC42	Formulier versturen met ‘textarea’ niet als verplicht veld.	6	UC10
TC43	Formulier versturen met ‘textarea’ als verplicht veld.	6	UC10
TC44	Formulier versturen met meerdere velden als verplicht veld.	6	UC10
TC45	Formulier versturen zonder internet	7	UC10

1.2. Resultaat uitgevoerde tests

Testnummer	Data		Verwacht resultaat	Werkelijk resultaat
	email	Verificatiecode		
TC1	<leeg>	nvt	Er kan geen leeg e-mail adres ingevuld worden.	V
TC2	test@	nvt	Foutmelding onjuist e-mail adres	V
TC3	info@fourict.nl	<leeg>	Foutmelding onjuist e-mail adres	V
TC4	mark@fourict.nl	<leeg>	Er kan geen lege verificatie code ingevuld worden.	V
TC5	mark@fourict.nl	12345	Foutmelding onjuiste verificatie code ingevuld	V
TC6	info@fourict.nl	QWRTYDDT	Autorisatie gelukt	V

Sprint 2

Testnummer	Actie	Verwacht resultaat	Werkelijk resultaat
TC7	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Klik op de button ‘Opslaan’	Foutmelding formulier kan niet worden toegevoegd	Er wordt geen foutmelding gegeven

TC8	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het input veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Foutmelding ‘naam’ is verplicht	V
TC9	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Vul ‘Naam’ in van formulier	Naam is ingevuld	V
	Klik op de button ‘Opslaan’	Foutmelding er moet tenminste een veld worden toegevoegd.	V
TC10	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘input’ als veldtype	Het veld ‘input’ is toegevoegd.	V
TC11	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V

	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het input veld met bijbehorende eigenschappen wordt getoond.	V
	Klik op de button ‘Opslaan’	Foutmelding ‘naam’ van veld is verplicht.	V
TC12	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het input veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Formulier is toegevoegd met een ‘input’ veld.	V
TC13	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘checkbox’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het checkbox veld met bijbehorende eigenschappen wordt getoond.	V
	Klik op de button ‘Opslaan’	Foutmelding ‘naam’ van veld is verplicht.	V
TC14	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V

	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘checkbox’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het checkbox veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Formulier is toegevoegd met een ‘checkbox’ veld.	V
TC15	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘checkbox’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het checkbox veld met bijbehorende eigenschappen wordt getoond.	V
	Druk op ‘Verwijderen’	Veld ‘checkbox’ is verwijderd.	V
TC16	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘select’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het select veld met bijbehorende eigenschappen wordt getoond.	V
	Klik op de button ‘Opslaan’	Foutmelding ‘naam’ van veld is verplicht.	V

TC17	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘select’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het select veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Vul ‘Naam’ en ‘Waarde’ in en druk op ‘Toevoegen optie’	Optie is toegevoegd	V
	Klik op de button ‘Opslaan’	Formulier is toegevoegd met een ‘select’ veld.	V
TC18	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘select’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het select veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ en ‘Waarde’ in en druk op ‘Toevoegen optie’	Optie is toegevoegd	V
	Druk op ‘Verwijderen Optie’	Optie is verwijderd uit het ‘select’ veld.	V
TC19	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V

	Selecteer ‘select’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het select veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Vul ‘Naam’ en ‘Waarde’ in en druk op ‘Toevoegen optie’	Optie is toegevoegd	V
	Klik op de button ‘Opslaan’	Formulier is toegevoegd met een ‘select’ veld.	V
TC20	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘select’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het select veld met bijbehorende eigenschappen wordt getoond.	V
	Druk op ‘Verwijderen’	Veld ‘select’ is verwijderd	V
TC21	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘textarea’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het textarea veld met bijbehorende eigenschappen wordt getoond.	V
	Klik op de button ‘Opslaan’	Foutmelding ‘naam’ van veld is verplicht.	V

TC22	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘textarea’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het textarea veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Formulier is toegevoegd met een ‘textarea’ veld.	V
TC23	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Klik op de button ‘Annuleren’	Formulier is niet toegevoegd en de lijst van formulieren worden getoond.	V
TC24	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘select’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het select veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Vul ‘Naam’ en ‘Waarde’ in en druk op ‘Toevoegen optie’	Optie is toegevoegd	V

	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het input veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Formulier is toegevoegd met velden ‘select’ en ‘input’	V
TC25	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het ‘input’ veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Na toevoegen van formulier wordt de naam van het formulier getoond in de lijst.	V
TC26	Meld aan als formuliergebruiker	Aangemeld als formuliergebruiker	V
	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Vul ‘Naam’ in van formulier	Naam is ingevuld	V
	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het input veld met bijbehorende eigenschappen wordt getoond.	V

	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Overzicht van de formulieren wordt getoond	V
	Meld aan systeembeheerder	Aangemeld als systeembeheerder	V
TC27	Klik in de navigatie op ‘Forms’	Het formulier die door de formuliergebruiker is aangemaakt wordt getoond in de lijst.	V
	Meld aan als formuliergebruiker	Aangemeld als formuliergebruiker	V
	Klik in de navigatie op ‘Forms’	De lijst van formulieren wordt getoond	V
	Klik op de button ‘Toevoegen’	Het scherm voor het ontwerpen van een formulier wordt getoond	V
	Vul ‘Naam’ in van formulier	Naam is ingevuld	V
	Selecteer ‘input’ als veldtype	Veldtype is geselecteerd	V
	Druk op ‘Toevoegen’	Het input veld met bijbehorende eigenschappen wordt getoond.	V
	Vul ‘Naam’ van veld in	Naam van veld is ingevuld	V
	Klik op de button ‘Opslaan’	Alleen de formulieren die de formuliergebruiker heeft gemaakt wordt getoond.	V

Sprint 3

Testnummer	**Verwacht resultaat**	**Werkelijk resultaat**
TC28	Formulier gebruiker is toegevoegd.	V
TC29	Foutmelding ‘Je bent eigenaar van het formulier’	V

TC30	Foutmelding ‘E-mail adres klopt niet’	V
TC31	Foutmelding ‘Gebruiker is reeds toegekend’	V

Sprint 4

Testnummer	**Verwacht resultaat**	**Werkelijk resultaat**
TC32	Formulier gebruiker is verwijderd.	V
TC33	Foutmelding ‘Formuliergebruiker is niet toegekend aan formulier’	V
TC34	Foutmelding ‘Je bent geen eigenaar van het formulier’	V
TC35	Formulier gebruiker is verwijderd.	X, Foutmelding dat ik geen eigenaar ben.

Sprint 5

Testnummer	**Data**		**Verwacht resultaat**	**Werkelijk resultaat**
	email	**Verificatiecode**		
TC1	<leeg>	nvt	Er kan geen leeg e-mail adres ingevuld worden.	V
TC2	test@	nvt	Foutmelding onjuist e-mail adres	V
TC3	info@fourict.nl	<leeg>	Foutmelding onjuist e-mail adres	V
TC4	mark@fourict.nl	<leeg>	Er kan geen lege verificatie code ingevuld worden.	V
TC5	mark@fourict.nl	12345	Foutmelding onjuiste verificatie code ingevuld	V
TC6	info@fourict.nl	QWRTYDDT	Autorisatie gelukt	V

TC25	Na toevoegen van formulier wordt de naam van het formulier getoond in de lijst.	V
TC26	Als rol ‘systeembeheerder’ worden alle formulieren getoond in de lijst.	Nee, er zijn geen resultaten gevonden ondanks dat er een formulier is toegevoegd door een andere gebruiker.
TC27	Alleen de formulieren die de formuliergebruiker heeft gemaakt wordt getoond.	V

Sprint 6

Testnummer	Verwacht resultaat	Werkelijk resultaat
TC31	Formulier gebruiker is verwijderd.	V
TC32	Foutmelding ‘Formuliergebruiker is niet toegekend aan formulier’	V
TC33	Foutmelding ‘Je bent geen eigenaar van het formulier’	V
TC34	Formulier gebruiker is verwijderd.	X, Foutmelding dat ik geen eigenaar ben.

Sprint 7

Testnummer	Verwacht resultaat	Werkelijk resultaat
TC31	Formulier gebruiker is verwijderd.	V
TC32	Foutmelding ‘Formuliergebruiker is niet toegekend aan formulier’	V
TC33	Foutmelding ‘Je bent geen eigenaar van het formulier’	V
TC34	Formulier gebruiker is verwijderd.	X, Foutmelding dat ik geen eigenaar ben.

Sprint 6

Testnummer	Actie	Verwacht resultaat	Werkelijk resultaat
TC36	Uitvoeren stappen van TC12	Formulier ontworpen met ‘input’ veld	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V

	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘input’ veld.	V
	Klik op de button ‘Versturen’	Formulier wordt per e-mail verstuurd aan gebruiker	V
TC37	Uitvoeren stappen van TC12	Formulier ontworpen met ‘input’ veld als verplicht veld.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘input’ als verplicht veld.	V
	Klik op de button ‘Versturen’	Foutmelding ‘input is verplicht’	V
TC38	Uitvoeren stappen van TC14	Formulier ontworpen met ‘checkbox’ veld.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘checkbox’ veld.	V
	Klik op de button ‘Versturen’	Formulier wordt per e-mail verstuurd aan gebruiker	V
TC39	Uitvoeren stappen van TC14	Formulier ontworpen met ‘checkbox’ veld als verplicht veld.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V

	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘checkbox’ veld als verplicht veld.	V
	Klik op de button ‘Versturen’	Foutmelding ‘checkbox’ is verplicht’	V
TC40	Uitvoeren stappen van TC19	Formulier ontworpen met ‘select’ veld en opties.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘select’ veld en opties.	V
	Klik op de button ‘Versturen’	Formulier wordt per e-mail verstuurd aan gebruiker	V
TC41	Uitvoeren stappen van TC19	Formulier ontworpen met ‘select’ veld en opties als verplicht.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘select’ veld en opties als verplicht veld.	V
	Klik op de button ‘Versturen’	Foutmelding ‘select’ is verplicht	V
TC42	Uitvoeren stappen van TC22	Formulier ontworpen met ‘textarea’ veld.	V

	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘textarea’ veld	V
	Klik op de button ‘Versturen’	Formulier wordt per e-mail verstuurd aan gebruiker	V
TC43	Uitvoeren stappen van TC22	Formulier ontworpen met ‘textarea’ veld als verplicht.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met ‘textarea’ veld als verplicht	V
	Klik op de button ‘Versturen’	Foutmelding ‘textarea’ is verplicht	V
TC44	Uitvoeren stappen van TC24	Formulier ontworpen met meerdere velden als verplicht veld.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met meerdere velden als verplicht veld.	V
	Vul verplicht velden in	Verplichte velden zijn ingevuld.	V
	Klik op de button ‘Versturen’	Formulier wordt per e-mail verstuurd aan gebruiker	V

TC45	Uitvoeren stappen van TC24	Formulier ontworpen met meerdere velden als verplicht veld.	V
	Start mobiele applicatie en autoriseer	Mobiele applicatie gestart en je bent geautoriseerd.	V
	Selecteer tabblad ‘Formulieren’	Formulierlijst wordt getoond	V
	Selecteer ontworpen formulier	Ontworpen formulier wordt getoond met meerdere velden als verplicht veld.	V
	Vul verplicht velden in	Verplichte velden zijn ingevuld.	V
	Zet internet uit op de mobiele applicatie.	Er is geen internet verbinding	V
	Klik op de button ‘Versturen’	Formulier wordt lokaal opgeslagen	V

Afstudeerplan

Informatie afstudeerder en gastbedrijf (*structuur niet wijzigen*)

Afstudeerblok: 2015-2.2 (start uiterlijk 16 november 2015)
Startdatum uitvoering afstudeeropdracht: 16 november 2015
Inleverdatum afstudeerdossier volgens jaarrooster: 29 maart 2016

Studentnummer: 12092452
Achternaam: dhr Barto (*) *weghalen niet van toepassing*
Voorletters: M.A.
Roepnaam: Mark
Adres: Hoedenmaker 31
Postcode: 2761LW
Woonplaats: Zevenhuizen
Telefoonnummer: 0180756704
Mobiel nummer: 0643890244
Privé emailadres: mark.barto@me.com

Opleiding: Informatica
Locatie: Den Haag
Variant: deeltijd c.q. avond

Naam studieloopbaanbegeleider: Mevr. G.C.M. Heijne
Naam begeleidend examinator: dhr. E.M. van Doorn
Naam tweede examinator: mevr. A.M.J.J Lousberg

Naam bedrijf: FourICT B.V.
Afdeling bedrijf:
Bezoekadres bedrijf: Rokkeveenseweg 44c
Postcode bezoekadres: 2712 XZ
Postbusnummer:
Postcode postbusnummer:
Plaats: Zoetermeer
Telefoon bedrijf: +31 79 7 630 480
Telefax bedrijf:
Internetsite bedrijf: http://www.fourict.nl

Achternaam opdrachtgever: dhr. van Ewijk (*) *weghalen niet van toepassing*
Voorletters opdrachtgever: H.
Titulatuur opdrachtgever:
Functie opdrachtgever: Managing Partner
Doorkiesnummer opdrachtgever:
Email opdrachtgever: henk@fourict.nl

Achternaam bedrijfsmentor: dhr. van Leeuwe (*) *weghalen niet van toepassing*
Voorletters bedrijfsmentor: R.
Titulatuur bedrijfsmentor:
Functie bedrijfsmentor: Managing Partner
Doorkiesnummer bedrijfsmentor:
Email bedrijfsmentor: richard@fourict.nl

NB: bedrijfsmentor mag dezelfde zijn als de opdrachtgever

Doorkiesnummer afstudeerder:
Functie afstudeerder (deeltijd/duaal): deeltijd

Titel afstudeeropdracht:

Ontwikkeling van een generiek formulierenregistratiesysteem bij FourICT B.V.

Opdrachtomschrijving

1. Bedrijf

FourICT B.V. is een IT bedrijf dat in 2008 is opgericht en zich heeft gespecialiseerd in het ontwikkelen en integreren van complexe ICT toepassingen in de Telecom, Media en Healthcare sector. FourICT bestrijkt het gehele traject van analyse, ontwerp, ontwikkeling, test en implementatie tot en met projectmanagement en beheer.

Het bedrijf bestaat uit een team van 10 vaste medewerkers en 3 zelfstandige ondernemers welke ondersteund worden door een project secretariaat. FourICT stelt kwaliteit boven kwantiteit en maakt een beheerste groei door in aantal vaste en externe medewerkers. Met het aantrekken van de economie en de toename van het aantal prospects en concrete opdrachten verwacht FourICT in 2016 een stap te kunnen maken naar 15 tot 20 medewerkers. FourICT richt zich in het bijzonder op proces automatisering (BPMS AuraPortal), Enterprise Service Bus oplossingen, Microsoft development en mobiele app ontwikkeling.

Voor dit project is de opdrachtgever Richard van Leeuwe van FourICT.

2. Probleemstelling

Er zijn nog veel bedrijven die gebruik maken van handmatige ingevulde (papieren) formulieren, waarvan de inhoud vervolgens weer handmatig moet worden overgezet naar één of meerdere backoffice systemen. Een kostbare activiteit, die niet alleen veel tijd kost maar ook fouten in de hand werkt en impact heeft op de uiteindelijke kwaliteit van de werkzaamheden.

Er moet een webapplicatie ontwikkeld worden waarbij een klant zelf een digitaal formulier kan ontwerpen ter vervanging van een papieren formulier. Het digitale formulier kan vervolgens in een app geladen worden op een mobiel apparaat van een werknemer van de klant. In de app zijn diverse controles ingebouwd die zorgdragen voor een kwalitatief hoogwaardig product (formulier). Standaard kan een ingevuld formulier per mail worden opgestuurd naar een vast backoffice emailadres. Als er specifieke koppelingen moeten komen met backend (klant) systemen dan is er de mogelijkheid om een maatwerk oplossing te realiseren op basis van de reeds beschikbare tools. Denk hierbij aan MS BizTalk, nServicebus en eventueel in combinatie met een Business Process Management suite (AuraPortal). Hiermee wordt de mogelijkheid gecreëerd om de data van het formulier rechtstreeks (geautomatiseerd) in de backend systemen op te nemen.

De kracht van de applicatie zal een gebruiksvriendelijk systeem zijn dat eenvoudig te onderhouden is. Tevens heeft de applicatie een grote flexibiliteit m.b.t. het integreren van data van het formulier via e-mail of tot een maatwerk interface met het backoffice systeem van de klant.

3. Doelstelling van de afstudeeropdracht

Het ontwikkelen van een webapplicatie waarbij klanten zelf digitale formulieren kunnen ontwerpen en beheren. Daarnaast wordt een hybride app ontwikkeld waarbij het formulier vervolgens geladen wordt en die beschikbaar is voor mobiele apparaten. Met behulp van deze hybride app worden gegevens gecontroleerd en standaard doorgezet via de e-mail of tot een maatwerk interface van de klant voor verdere verwerking.

4. Resultaat

De volgende functionaliteiten worden gerealiseerd voor het digitaal formulierensysteem:

- Een generiek formulierensysteem (webapplicatie) dat platformonafhankelijk is en toegankelijk is vanaf iedere computer, waarbij de klant zich kan aanmelden voor het gebruik van het systeem.
- De klant kan zijn eigen digitale formulier ontwerpen, invoervelden definiëren en restricties opgeven voor het valideren van invoervelden.
- De mogelijkheid dat de klant binnen het systeem medewerkers toegang kan geven tot het digitale formulier.
- De klant moet kunnen bepalen op welke manier het digitale formulier wordt doorgezet. Standaard kan dit via de mail.

De volgende functionaliteiten worden gerealiseerd voor de app:

- Een generieke app (hybride) die beschikbaar komt voor verschillende mobiele apparaten die op verschillende platformen (iOS, Android) werkt.
- De medewerker van de klant kan zich aanmelden in de app om het digitale formulier op te halen.
- Het formulier moet ook offline beschikbaar zijn in de app zodat als er geen verbinding tot stand kan worden gebracht de gegevens lokaal worden opgeslagen en niet verloren gaan.
- Bij het versturen van het formulier moet er een validatie plaats vinden van de invoervelden.

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

Activiteit	Aantal dagen
Plan van aanpak schrijven met daarin een globale planning.	1
Een marktonderzoek doen naar bestaande oplossingen van het formulierensysteem.	3
Onderzoek naar gebruikte ontwikkeltools en frameworks om de app en het systeem te ontwikkelen.	3
Het opstellen van een analyse document met daarin uitgewerkt de werking van het systeem met behulp van een context diagram. Daarnaast bijbehorende procesbeschrijvingen, functionele en niet-functionele eisen, use case diagrammen incl. beschrijvingen.	10
Het ontwerpen van de database voor het generieke formulierensysteem uitgewerkt in een UML class diagram, relationeel representatiemodel en relationeel implementatiemodel.	5
Ontwikkeling digitaal formulierensysteem.	25
Ontwikkeling van de app.	10
Automatisch testen van het digitaal formulierensysteem met behulp van unit tests / end-to-end test.	5
Productie integratietest / acceptatietest uitvoeren	3
Het implementeren / deployen van het digitaal formulierensysteem en de app.	5
Het bijwerken van het afstudeerdossier tijdens elke activiteit.	15
Totaal	85

6. Op te leveren (tussen)producten

1. Plan van aanpak.
2. Onderzoeksrapport waarbij onderzocht is welke huidige oplossingen van het formulierensysteem er momenteel zijn.
3. Analyse document met daarin uitgewerkt de werking van het systeem in een sequence diagram, een class diagram, de procesbeschrijvingen, de functionele en niet-functionele eisen, de use case diagrammen inclusief beschrijvingen en tenslotte een deployment diagram.
4. Database uitgewerkt in een UML class diagram, relationeel representatiemodel en relationeel implementatiemodel.
5. Een testplan met de uitwerking van de teststrategie
6. Een testrapport met de uitwerking van de uitgevoerde testen.
7. Opleveren van het formulierensysteem op de OTAP omgeving.
8. Opleveren van de app in de Apple app store en Google play store.

7. Te demonstreren competenties en wijze waarop

De volgende beroepstaken zullen binnen dit project worden uitgelicht:

Analyseren en adviseren
Uitvoeren analyse door definitie van requirements (Niveau 3)

Databaseontwerp en –ontwikkeling
Opstellen gegevensmodel voor een database (Niveau 3)
Ontwerpen, bouwen en bevragen van een database (Niveau 3)

Softwareontwerp en -ontwikkeling
Ontwerpen systeemdeel (Niveau 3)
Bouwen applicatie (Niveau 3)
Uitvoeren van en rapporteren over het testproces (Niveau 3)