AFSTUDEERVERSLAG

LetsGrow.com

Naam: Ron Persoon
Bedrijf: LetsGrow.com
Versie: 1.3
Datum: 22-12-16
Plaats: Vlaardingen

Opdrachtgever:
Peter Hendriks

Begeleider:
Leon Batta

1e examinator:
Mevrouw Lousberg

2e examinator:
Mevrouw van der Hoek

Voorwoord en referaat

In dit verslag worden de gemaakte beslissingen, problemen en oplossingen, motivaties en evaluaties van mijn afstudeertraject beschreven. Het afstuderen vond plaats bij LetsGrow.com te Vlaardingen in de periode van 31 augustus tot 22 december 2016.

Het verslag is in chronologische volgorde opgebouwd, zodat het verslag steeds verder kon worden uitgebreid naar mate er meer beslissingen en keuzes gemaakt werden.

LetsGrow.com heeft mij precies kunnen bieden wat ik van een afstudeeropdracht verwachtte. Ik heb zelfstandig kunnen werken aan een nieuwe applicatie, als toevoeging op het huidige systeem. Er hebben regelmatig gesprekken plaatsgevonden met de opdrachtgever en begeleider, om zo de juiste keuzes voor deze applicatie te kunnen maken.

Ten eerste wil ik Leon Batta, mijn begeleider bij LetsGrow.com, bedanken voor alle besprekingen, feedback en ondersteuning op mijn werk. Dit heeft zeer veel positieve invloed gehad op het eindresultaat. Ten tweede wil ik mijn opdrachtgever, Peter Hendriks, bedanken voor de gesprekken, ideeën en feedback. Ook dit heeft zeer veel meegeholpen met het opleveren van een mooie applicatie. Ten derde wil ik Edwin Vis van harte bedanken voor de ondersteuning en feedback op mijn werk. Dit heeft absoluut geleid tot een kwalitatief beter eindproduct. Tot slot wil ik iedereen van het LetsGrow.com team bedanken voor de goede sfeer en gezelligheid. Ik heb het erg naar mijn zin gehad in het afgelopen half jaar.

Vanuit school wil ik mevrouw Lousberg bedanken voor de goede ondersteuning en feedback op mijn gemaakte producten. Ik heb hier erg veel aan gehad en dit heeft mij absoluut geholpen om het afstudeerverslag op een hoger niveau te brengen. Ook wil ik mevrouw van der Hoek ten zeerste bedanken voor de feedback op mijn documenten. Ook deze feedback heeft geleid tot een beter eindresultaat.

Referaat

Ron Persoon, “Ontwikkeling dashboard applicatie voor LetsGrow.com”. Afstudeerverslag opleiding Informatica, Haagse Hogeschool, 2016

Inhoud

Samenvatting

Tijdens de afstudeerperiode bij LetsGrow.com is er een dashboard ontwikkeld. Middels dit dashboard krijgen klanten de mogelijkheid om sneller en eenvoudiger hun belangrijkste data in te kunnen zien. Dit dashboard zal de klanten een betere gebruikerservaring bieden en zorgt ervoor dat LetsGrow.com voor iedereen eenvoudig in gebruik is.

Tevens zorgt het dashboard ervoor dat de klant nu ook via zijn mobiele apparaten gebruik kan maken van LetsGrow.com. Dit was in de oude situatie niet goed mogelijk.

Het afstudeertraject is volgens een iteratieve methode doorlopen. Iedere iteratie zijn, indien er nieuwe requirements bijgekomen waren, de requirements opnieuw geprioriteerd. Dit zorgde ervoor dat de belangrijkste functionaliteiten eerst werden ontwikkeld. Verder zijn er (indien nodig) tijdens iedere iteratie wijzigingen doorgevoerd in het functioneel- en technisch ontwerp, detailtestplan, testontwerp en heeft er een stuk ontwikkeling plaatsgevonden.

Tijdens het afstudeertraject is er ook onderzoek gedaan naar bestaande dashboard pakketten. Dit bood de mogelijkheid om, indien er een geschikt pakket beschikbaar was, meer tijd te steken in het ontwikkelen van nieuwe functionaliteiten. Tevens ging de voorkeur van LetsGrow.com uit naar het gebruiken van een bestaand dashboard pakket. Er bleek tijdens dit onderzoek een geschikt dashboard pakket beschikbaar te zijn en deze is vervolgens als basis gebruikt voor het nieuwe dashboard voor LetsGrow.com.

Dit dashboard is vervolgens uitgebreid met nieuwe functionaliteiten. Tevens zijn een aantal nieuwe wensen van de klant gerealiseerd. Hiervoor is gebruik gemaakt van een aantal programmeertalen die binnen het bedrijf worden toegepast.

Tot slot zijn er iedere iteratie testen opgesteld en uitgevoerd. Deze testen zijn geïnspireerd op TestGoal. Hierbij is er onder andere met behulp van een risicoanalyse bepaald waar de grootste risico's zitten en met welke diepgang er getest moest worden.

1. Inleiding

In dit verslag wordt op chronologische volgorde het proces besproken dat is doorlopen tijdens mijn afstudeertraject bij LetsGrow.com (hierna LetsGrow). Het document begint met een introductie over LetsGrow waarin besproken wordt wat LetsGrow is en wat zij doet.

Vervolgens zal de afstudeeropdracht worden besproken, waarbij de probleem- en doelstelling aan bod komen en zal vervolgens de nieuwe situatie worden geschetst. Hierna zal worden beschreven hoe de opdracht aangepakt gaat worden, waarbij onder andere het plan van aanpak zal worden besproken. Het plan van aanpak is achter in dit document bijgevoegd.

1.1 Over LetsGrow

Historie

LetsGrow is opgericht in 2000 uit een samenwerking tussen Hoogendoorn Growth Management (HGM) en de WUR Glastuinbouw (Wageningse Universiteit Researchcentrum). Hoogendoorn is actief in de tuinbouwsector in binnen- en buitenland en is fabrikant van onder andere klimaatcomputers. Met een klimaatcomputer kunnen kwekers (vooraf) geconditioneerd het klimaat in de kas aanpassen waarbij er ingespeeld kan worden op invloeden van buitenaf om zo voor de teelt het optimale klimaat te realiseren. In 2006 kwam de samenwerking tussen Hoogendoorn en de WUR ten einde en is LetsGrow in zijn geheel overgenomen door Hoogendoorn.

De organisatie

Het team van LetsGrow bestaat uit 10 werknemers, waarvan een manager, vier ontwikkelaars, een tester, een administratief/helpdesk medewerker en drie helpdesk medewerkers.

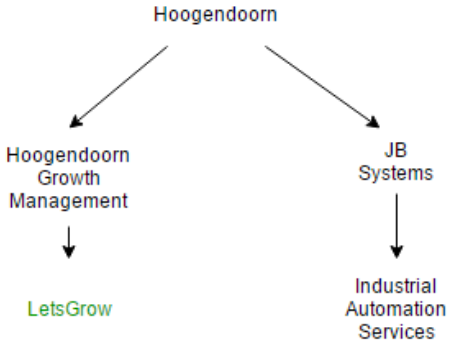
De personen waar ik tijdens het afstuderen voornamelijk mee te maken krijg zijn Peter Hendriks, Leon Batta en Edwin Vis. Over deze personen geef ik in de tabel hieronder wat meer informatie:

Naam	Rol	Beschrijving
Peter Hendriks	Opdrachtgever	Peter is de manager van LetsGrow en is tijdens mijn afstudeertraject ook opdrachtgever. Peter zal ook bepalen of eventuele benodigde software aangeschaft mag worden.
Leon Batta	Begeleider	Leon is een van de ontwikkelaars van LetsGrow. Hij zal de gemaakte documentatie en software reviewen en hier feedback op geven. Leon zal mij ook ondersteunden tijdens het ontwikkelproces. De expertise van Leon ligt voornamelijk bij het front-end van de applicaties van LetsGrow
Edwin Vis	Ondersteuning	Indien Leon afwezig of niet beschikbaar was kon ik bij Edwin langs voor technische ondersteuning. Edwin is ook een ontwikkelaar van LetsGrow. De expertise van Edwin ligt voornamelijk bij het backend van de LetsGrow applicaties.

Tabel 1: Beschrijving rollen LetsGrow

In de eerste week is gebleken dat mijn collega's erg behulpzaam zijn en mocht ik tegen een probleem aan lopen, is iedereen bereid te helpen.

Het kantoor van LetsGrow is gevestigd in het pand van de Hoogendoorn Groep te Vlaardingen. Naast LetsGrow is ook JB Systems in het zelfde pand gevestigd. JB Systems is net als LetsGrow een dochterbedrijf van de Hoogendoorn Groep en richt zich op industriële automatisering.



Afbeelding 1: Structuur Hoogendoorn

Het product

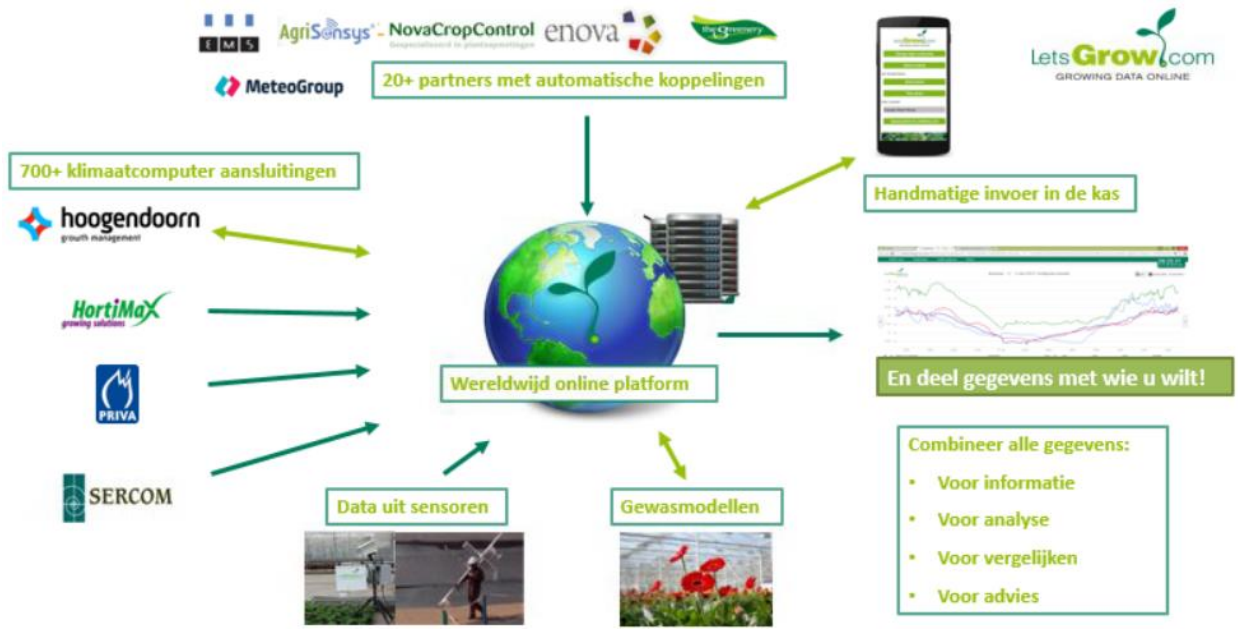
LetsGrow is een online platform en biedt haar klanten, waaronder tuinders in de glastuinbouw sector, de mogelijkheid om meetgegevens van onder andere klimaatcomputers te analyseren en te vergelijken. Het gaat hier bijvoorbeeld om data zoals lichtsterkte, temperatuur, luchtvochtigheid, arbeid en energieverbruik. Er worden 24 uur per dag klimaatgegevens naar LetsGrow verstuurd of actief opgehaald en vervolgens opgeslagen in een database.

Het product van LetsGrow is gericht op drie pijlers, namelijk:

Collect data: waarbij er niet alleen data uit klimaatcomputers wordt verzameld, maar ook uit vele andere bronnen, zoals de thermoview (warmtebeeldcamera), draadloze sensoren, plant sap metingen, fotosynthese metingen, handmatige invoer enzovoorts.
Add Value: door diverse berekeningen, groei- en plantmodellen en analyses aan te bieden kan de klant zijn/haar data volledig benutten.
Information: deze data wordt vervolgens op verschillende manieren gepresenteerd, zoals in grafiek of rapport vorm en helpt de telers in het maken van keuzes voor het optimale resultaat. Tevens kan LetsGrow op basis van deze informatie de klimaatcomputer beïnvloeden om het klimaat te optimaliseren .

Klanten hebben de mogelijkheid om modules aan te schaffen. Een module is een commercieel item met een bepaalde functionaliteit en looptijd. Een module biedt de klanten bijvoorbeeld een aantal grafieken, rapporten en mogelijk ook modelmatige berekeningen. Deze modelmatige berekeningen tonen bijvoorbeeld aan of de plant op de optimale manier groeit. LetsGrow biedt plantmodellen voor diverse teelten, zoals rozen, gerbera's, en tomaten.

Klanten kunnen de data uit de klimaatcomputer aan de modules in LetsGrow koppelen. De module zorgt voor het ophalen van de data en de presentatie van deze gegevens in de vorm van grafieken en/of rapporten. Doordat klanten zelf de modules kunnen kiezen die ze nodig hebben, hoeft er niet betaald te worden voor ongebruikte functionaliteiten.



Afbeelding 2: LetsGrow overzicht

Ook biedt LetsGrow.com de mogelijkheid om zogenoemde “Teeltgroepen” aan te maken waarbij er data gedeeld kan worden tussen verschillende tuinders of meerdere vestigingen.

De klanten
Momenteel maken meer dan 700 klanten gebruik van LetsGrow. Klanten zijn onderzoekers, teeltvoorlichters, energiemanagers, telers met kleine bedrijven en telers met grote bedrijven en/of meerdere vestigingen.

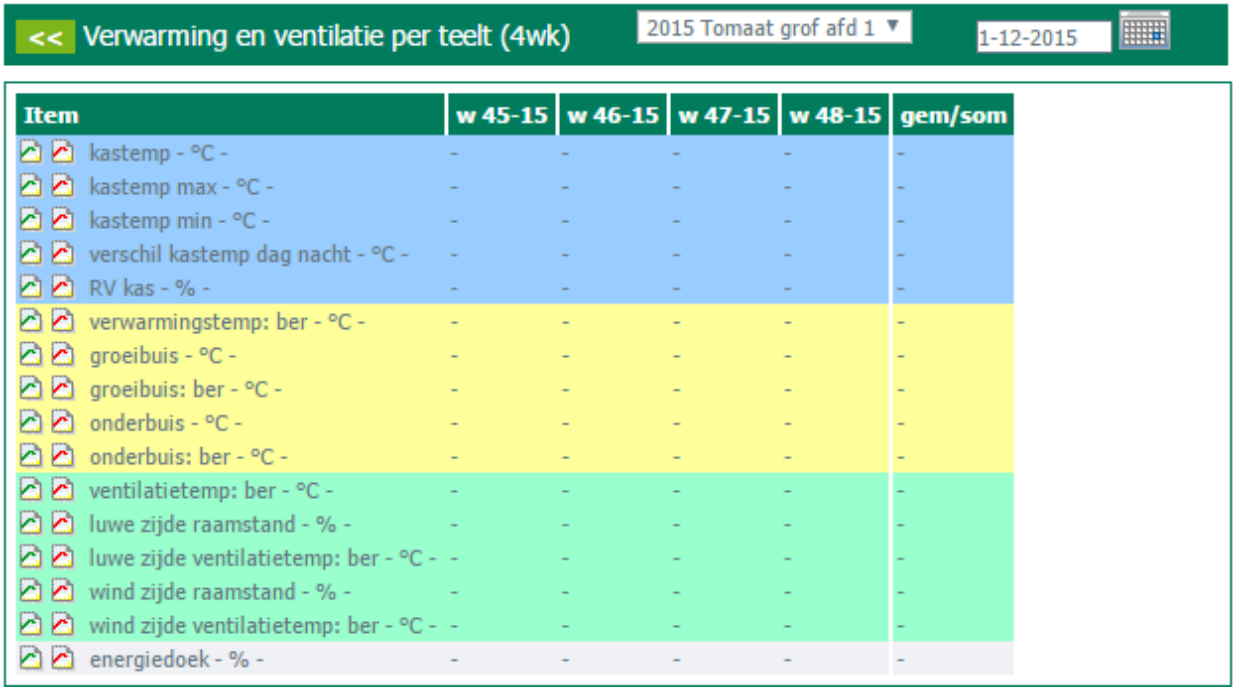
Tevens wordt LetsGrow dagelijks gebruikt door energie managers. Met data uit de klimaatcomputer van een teler wordt bepaalt hoe groot zijn/haar verwachtte energievraag zal worden. Hiermee kunnen de energiemanagers vervolgens biedingen doen op de Amsterdam Power Exchange (APX).

Het doel
Doordat LetsGrow de mogelijkheid biedt om onderling te vergelijken, bijvoorbeeld tussen meerdere locaties van een bedrijf of met collega’s, kunnen telers van elkaar leren en fouten voorkomen. Op deze manier realiseren ze in hun tuinbouwbedrijf het beste klimaat. Dit leidt tot meer productie, een betere kwaliteit en reducering van de kosten.

De visie
LetsGrow streeft ernaar wereldwijd dé online service te zijn voor tuinbouwondernemers, adviseurs, onderzoekers en andere belanghebbenden, zoals Svensson (schermdoeken) en Philips (verlichting). De waarde voor de klanten wordt gecreëerd door het omzetten van data naar informatie middels een online platform. LetsGrow streeft naar maximale kwaliteit en continue verbetering .

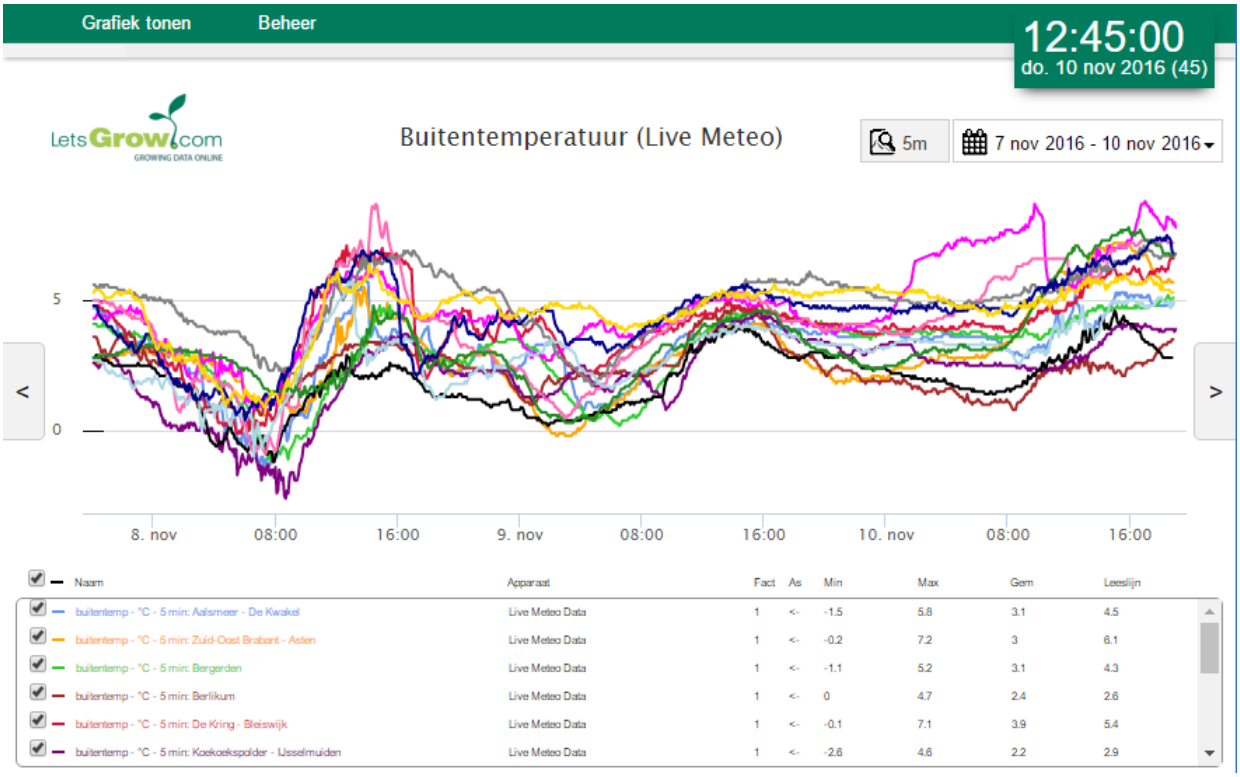
Meer informatie over LetsGrow en haar werking is te vinden op:
http://www.letsgrow.com/nl/wat-is-letsgrowcom

De afbeeldingen 3 en 4 hieronder laten een voorbeeld zien van de rapport en grafiek applicatie van LetsGrow.



Item	w 45-15	w 46-15	w 47-15	w 48-15	gem/som
kastemp - °C -	-	-	-	-	-
kastemp max - °C -	-	-	-	-	-
kastemp min - °C -	-	-	-	-	-
verschil kastemp dag nacht - °C -	-	-	-	-	-
RV kas - % -	-	-	-	-	-
verwarmingstemp: ber - °C -	-	-	-	-	-
groeibuis - °C -	-	-	-	-	-
groeibuis: ber - °C -	-	-	-	-	-
onderbuis - °C -	-	-	-	-	-
onderbuis: ber - °C -	-	-	-	-	-
ventilatietemp: ber - °C -	-	-	-	-	-
luwe zijde raamstand - % -	-	-	-	-	-
luwe zijde ventilatietemp: ber - °C -	-	-	-	-	-
wind zijde raamstand - % -	-	-	-	-	-
wind zijde ventilatietemp: ber - °C -	-	-	-	-	-
energiedoek - % -	-	-	-	-	-

Afbeelding 3: Voorbeeld rapport LetsGrow: Verwarming en ventilatie per teelt (4wk)



Afbeelding 4: Voorbeeld grafiek LetsGrow: Buitentemperatuur verschillende locaties in Nederland

2. Opdracht

In dit hoofdstuk worden een aantal aspecten van de afstudeeropdracht besproken. Er wordt besproken wat de doelstelling van de opdracht is, hoe de nieuwe situatie eruit zal komen te zien, wie de doelgroep is van de afstudeeropdracht en tot slot zal het project worden afgebakend.

2.1 Probleemstelling

Momenteel krijgen de klanten na inloggen op LetsGrow een pagina te zien met zeer beperkte informatie. De klant heeft hier niet de mogelijkheid om in een oogopslag de belangrijkste informatie en meetgegevens in te zien. Er worden op dit moment dan ook nog geen grafieken en meetgegevens getoond op de eerste pagina die de klant ziet na het inloggen.

Verder is het huidige systeem niet voorbereid voor het gebruik op mobiele telefoons. Dit is een grote beperkende factor in de huidige situatie, aangezien hier steeds meer vraag naar is. Verder is er ook de vraag om op het klantenportal data uit verschillende externe bronnen te kunnen tonen, zoals twitter, het weerbericht en eventueel data van de veiling. Deze functionaliteit is namelijk in het huidige systeem niet beschikbaar.

2.2 Doelstelling

De doelstelling van de opdracht is het ontwikkelen van een overzichtelijke user interface in de vorm van een dashboard, waarbij klanten zelf het dashboard kunnen inrichten en sneller relevante informatie tot hun beschikking hebben. Dit dashboard dient tevens voor een betere uitstraling van LetsGrow te zorgen en zal op een intuïtieve manier te bedienen zijn, zodat iedere klant goed overweg kan met LetsGrow. Dit kan vervolgens zorgen voor meer (internationale) klanten. Het dashboard zal in meerdere talen beschikbaar zijn, namelijk Nederlands, Engels en Frans.

Het dashboard zal verder responsive gemaakt moeten worden, om deze ook bruikbaar te maken op mobiele apparaten zoals mobiele telefoons en tablets. Ook zal er ook de mogelijkheid geboden moeten worden om data uit externe bronnen te tonen.

Tot slot is het belangrijk dat het dashboard de huisstijl van LetsGrow implementeert, zodat het dashboard een geheel wordt met de rest van de LetsGrow applicaties en dat het voor klanten duidelijk is dat het dashboard een onderdeel is van LetsGrow. Het dashboard is dus een volledig nieuwe functionaliteit binnen LetsGrow, aangezien er momenteel geen dashboard aanwezig is.

2.3 Nieuwe situatie

In de nieuwe situatie heeft de klant beschikking over een dashboard. Dit dashboard zal eenvoudig in gebruik zijn en klanten kunnen deze makkelijk inrichten naar eigen inzicht.

Op dit dashboard kunnen de klanten op een overzichtelijke manier hun (belangrijkste) meetgegevens inzien. Hiervoor kunnen elementen, zoals grafieken en rapporten, op het dashboard worden geplaatst. Verder zal de mogelijkheid geboden worden om data te kunnen tonen uit externe bronnen.

Het ontwikkelde dashboard zal responsive zijn, zodat klanten de mogelijkheid geboden wordt om LetsGrow ook te gebruiken via hun mobiele telefoon of tablet. Ook zal het dashboard opgemaakt zijn volgens de huisstijl van LetsGrow. Het resultaat van de opdracht is een werkend dashboard.

2.4 Doelgroep

De doelgroep van het dashboard zijn de bestaande en potentiële klanten van LetsGrow. Het klantenbestand van LetsGrow wordt steeds verder uitgebreid, waaronder ook steeds meer internationale klanten. Om deze klanten, die mogelijk minder ervaren zijn met het gebruik van computers, ook goed van dienst te kunnen zijn, is het noodzakelijk om hier een duidelijk en overzichtelijk dashboard voor te ontwikkelen.

2.5 Afbakening

De huidige grafiek en rapport applicatie zullen gebruikt worden om de data te tonen, waardoor het niet nodig is om voor het dashboard een nieuwe grafiek of een nieuw rapport te ontwikkelen.

Eventuele configuratie wijzigingen van de database van LetsGrow zullen worden uitgevoerd door Edwin of Leon. Deze wijzigingen kunnen nodig zijn om een koppeling te creëren tussen LetsGrow en het nieuwe dashboard. Eventuele nieuwe database tabellen zullen wel door mij worden toegevoegd aan de database.

Het project zal door mij zelf worden voorbereid om op de test- of productieomgeving te plaatsen. Het daadwerkelijk op de test- of productieomgeving plaatsen zal gerealiseerd worden door Edwin of Leon middels hun zelfgemaakte tool. De ervaring leert namelijk dat er nog wel eens iets fout wil gaan tijdens het doorzetten en er dan extra privileges nodig zijn om dit aan de server kant op te lossen. Volgens de procedure van LetsGrow hebben alleen Edwin en Leon deze privileges, aangezien zij al vele jaren ervaring hebben met deze werkzaamheden. Dit is dan ook de reden dat dit niet door mij uitgevoerd zal gaan worden.

3. Aanpak

Dit hoofdstuk bespreekt de aanpak van de afstudeeropdracht. In deze iteratie is er voornamelijk tijd gestoken in het helder krijgen van de huidige situatie en zijn de eerste requirements vastgesteld waar het dashboard uiteindelijk aan zal moeten voldoen. De activiteiten die in dit hoofdstuk worden besproken, betreffen hoofdzakelijk het inrichten van de werkplek, het opstellen van het plan van aanpak en het vaststellen van de requirements planning en de requirements.

3.1 Plan van aanpak

In het plan van aanpak worden diverse keuzes gemaakt om de aanpak te bepalen van het ontwikkeltraject. Hier wordt onder andere bepaald welke ontwikkelmethodiek er gebruikt gaat worden, hoe de requirements worden geanalyseerd en geprioriteerd, welke modelleringstechniek er gebruikt gaat worden.

Ontwikkelmethode

Voor het ontwikkelen van het dashboard zal ik een iets andere werkwijze hanteren dan dat LetsGrow doet, aangezien LetsGrow gebruik maakt van een eigen interpretatie van Scrum. De reden hiervoor is dat LetsGrow bestaat uit een relatief klein team met ieder zijn eigen expertise. Ook is er sprake van tussentijdse klantwensen en bugs, waardoor het niet mogelijk is om van tevoren precies in te plannen welke activiteiten er zullen worden uitgevoerd in een bepaalde sprint. Wel wordt er gebruik gemaakt van een scrum bord om de voortgang van alle taken bij te kunnen houden. Ook worden er stand-ups gehouden, maar dit gebeurt niet iedere dag, omdat niet iedereen dagelijks aanwezig is op kantoor en hier niet altijd behoefte aan is.

Voor mij is het, bij het ontwikkelen van het dashboard, wel mogelijk om een wat meer gestructureerde methode aan te houden. De reden hiervoor is dat ik mij kan focussen op het ontwikkelen van het dashboard en niet gehinderd zal worden door tussentijdse klantwensen of problemen.

Het is voor mij echter niet mogelijk om de officiële scrum methode aan te houden. Dit komt doordat ik slechts de enige ben in een scrumteam, er binnen LetsGrow niet aan retrospectives wordt gedaan en er ook geen scrummaster is.

Hierdoor is er gekozen om deels de scrum methodiek en deels de werkwijze van LetsGrow te volgen. Dit is voor mij de beste manier, aangezien ik zo ook gebruik kan maken van het scrumbord, deel kan nemen aan de stand-ups en toch op een gestructureerde manier in sprints kan werken. Ook kan, door het gebruik van scrum, na iedere iteratie opnieuw gekeken worden naar de prioritering van requirements en kan eventuele (klant) feedback worden ingepland met een hogere prioriteit.

Er zal gewerkt worden in iteraties van twee weken, om zo voldoende mogelijkheden te hebben om eventuele nieuwer en belangrijkere requirements voorrang te geven.

Het scrumbord dat ik ga gebruiken is ingericht op de muur van het kantoor waar ik in werk. Er zijn hier verschillende vakken gecreëerd, om zo bij te kunnen houden in welke fase een back-log item zich bevindt. Afbeelding 5 laat het volledige scrumbord zien dat door LetsGrow wordt gebruik en waar ik ook gebruik van ga maken.

Afbeelding 5: Productbacklog LetsGrow

De muur is van rechts naar links onderverdeeld in de volgende vakken: idea/concept, todo, in progress, in test, done en impediment.

- Idea/concept: Functionaliteiten die ooit in het systeem ingebouwd kunnen gaan worden
- Todo: Functionaliteiten die in het systeem moeten komen, maar waar nog niemand aan begonnen is.
- In progress: Hier hangen de functionaliteiten waar aan gewerkt wordt.
- In Test: Hier hangen die functionaliteiten die af zijn en getest worden.
- Done: Deze functionaliteit is succesvol getest en kan gereleased worden.
- Impediment: Hier worden functionaliteiten opgehangen die niet gemaakt/afgemaakt kunnen worden vanwege een externe oorzaak.

Versiebeheer

Als versiebeheertool wordt er gebruik gemaakt van Microsoft Team Foundation Server. Deze dienst is geïntegreerd met Visual studio en is dus uitstekend te gebruiken voor Visual Studio projecten. Een andere cloud service van de Microsoft Team Foundation Services die gebruikt wordt, is de automated build. Hiermee kunnen de ingecheckte projecten worden gebouwd, om deze later op de test of productieomgeving te kunnen plaatsen. Onder het kopje “ontwikkelstraat” wordt besproken welke omgevingen er binnen LetsGrow gebruikt worden.

Requirements analyse
Om requirements op een duidelijke manier te kunnen ordenen aan de hand van de prioriteiten wordt er gebruik gemaakt van MoSCoW(Clegg, 1994). MoSCow staat voor Must have, Should have, Could have en Won't have. Na het ordenen van de requirements ontstaat er een duidelijk beeld van welke requirements er als eerst gerealiseerd moeten worden.

- Must have: Deze functionaliteit moet in het eindproduct zijn opgenomen. Dit is belangrijk voor de werking van het systeem.
- Should have: Deze functionaliteit zou in het eindproduct opgenomen moeten zijn, maar de applicatie is ook bruikbaar zonder deze functionaliteit.
- Could have: Het zou mooi zijn als deze functionaliteit in het eindproduct wordt opgenomen, maar hier hoeft pas aandacht aan besteed te worden indien de overige functionaliteit is gerealiseerd.
- Won't have: Deze functionaliteit zal (nog) niet in het systeem opgenomen worden, maar zal mogelijk in latere versies worden opgenomen.

Ontwikkelstraat
LetsGrow werkt met verschillende omgevingen, namelijk de Ontwikkel, Test en Productieomgeving. De ontwikkelomgeving is sterk verbonden met de testomgeving, aangezien de ontwikkelomgeving de data gebruikt van de servers van de testomgeving. De ontwikkelomgeving draait ook niet op een aparte server, maar lokaal op de pc van de software ontwikkelaar.

Nadat de software ontwikkelaar klaar is met een functionaliteit wordt deze gereviewed en, indien de wijzigingen worden goedgekeurd, doorgevoerd naar de testomgeving. Zodra de functionaliteit hierop ook wordt goedgekeurd, zal deze worden doorgezet naar de productieomgeving. Er wordt bewust geen gebruik gemaakt van een acceptatie omgeving. Het gebruik hiervan weegt niet op tegen de extra tijd en kosten die inrichting en onderhoud van een dergelijke omgeving met zich mee brengt.

Ik heb er voor gekozen om de ontwikkelstraat van LetsGrow aan te houden, omdat dit voor het bedrijf goed werkt en deze ontwikkelstraat ook prima te gebruiken is voor mijn opdracht. Voor mijn opdracht is de acceptatieomgeving ook niet nodig, aangezien de testomgeving voldoende op de productieomgeving lijkt.

Modelleringstechniek
Tijdens dit afstudeertraject zal er gebruik gemaakt gaan worden van UML(Unified Modeling Language) (Grady Booch, James Rumbaugh en Ivar Jacobson, 1997). Deze techniek wordt op grote schaal gebruikt en aangezien ik hier al veel ervaring mee heb is dit voor mij een goede techniek. UML zal gebruikt worden voor verschillende diagrammen, zoals klassendiagrammen, sequentiediagrammen, use cases en use case tabellen.

De diagrammen zullen worden gebruikt om documentatie op te bouwen van het te ontwikkelen dashboard. Door de werking van de dashboard applicatie te documenteren wordt het eenvoudiger om hier (in de toekomst) wijzigingen in aan te brengen.

Om de diagrammen en tabellen te modelleren wordt er gebruik gemaakt van de tool Visual Paradigm. Deze tool beschikt over voldoende mogelijkheden en doordat ik deze tool op school ook vaak heb gebruikt heb ik hier al ervaring mee.

Testmethodiek
Bij LetsGrow wordt voornamelijk getest op usability en functionality. Bij het testen wordt er ingeschat hoe groot de eventuele imagoschade kan zijn, indien er nieuwe functionaliteit wordt opgeleverd waarbij er enkele bugs kunnen optreden. Dit is een bewuste keuze, aangezien er geen tijd is om de nieuwe en huidige functionaliteit steeds opnieuw te testen. Op dit punt wijkt LetsGrow dan ook af van de officiële scrum methodiek, waar er gebruik gemaakt wordt van testautomatisering en regressietesten.

Om het dashboard te kunnen testen en zo de kwaliteit vast te stellen, zal er wel gebruik gemaakt moeten gaan worden van een testmethodiek. Ik heb gekozen voor TestGoal(de Grood, 2012), aangezien ik hier al bekend mee ben en er hier veel documentatie over te vinden is. Verder is testgoal geschikt voor deze opdracht, omdat dit veel verschillende testmogelijkheden biedt met verschillende dieptes. De testen die tijdens de iteraties worden gemaakt, zullen tevens ook dienen als regressietesten om te valideren of de bestaande functionaliteit nog steeds naar behoren functioneert.

Er zal zowel getest worden via de black- als de whitebox methode. Er wordt blackbox getest, omdat er vanuit het dashboard gegevens worden opgevraagd uit verschillende applicaties van LetsGrow. De whitebox testen zullen worden opgezet voor de door mij zelf ontwikkelde delen van het dashboard.

Testgoal past binnen de ontwikkelmethodiek, aangezien iedere iteratie testen worden opgezet voor het stuk functionaliteit dat in de betreffende sprint ontwikkeld is. Hierdoor zal het detailtestplan, de testontwerpen en de testrapportage steeds verder worden aangevuld.

Om de risico's te bepalen zal er gebruikt gemaakt worden van de testrisicoanalyse om hier vervolgens een strategie matrix mee op te stellen.

Risico's
Tijdens het afstudeertraject kunnen verschillende risico's optreden. Deze risico's kunnen leiden tot vertraging van het project. In de tabel hieronder worden een aantal risico's vermeld.

Wat	Kans	Impact	Oplossing
Ik word ziek	Midden	Klein	Een enkele dag ziek zal niet voor grote problemen zorgen, maar mocht dit te lang duren zal er contact opgenomen worden met school om te kijken naar een goede oplossing.
Mijn afstudeerbegeleider wordt ziek	Midden	Midden	Mocht dit slechts om een dag of paar dagen gaan, dan kan Edwin mij ondersteunen. Mocht dit om een langere periode gaan zal dit gemeld worden op school en zal er gekeken moeten worden of iemand anders van LetsGrow de rol van begeleider over kan nemen.
Mijn kennis is niet genoeg om de opdracht te kunnen voltooien	Midden	Midden	Ik zal zelf moeten zorgen dat ik over de benodigde kennis ga beschikken door mij uitgebreid in te lezen op de benodigde materie. Mocht dit niet voldoende zijn zal er contact opgenomen worden met school.

Tabel 2: Risico's tijdens afstudeertraject

3.2 Planning

Om de afstudeerperiode goed te doorlopen is het noodzakelijk om een goede planning te maken. Hierdoor is het nodig om na te denken over wat er mogelijk is binnen een tijdsbestek om de tijd optimaal te kunnen benutten. Ook bevat de planning een aantal mijlpalen. Deze mijlpalen zijn een aantal belangrijke punten waar ik naar toe kan werken. Doordat er niet volgens de officiële scrummethodiek wordt gewerkt, heb ik gekozen om de term iteraties te gebruiken in plaats van sprints. Deze term zal vanaf nu dan ook gebruikt worden.

Week 1 en 2: 29 Augustus t/m 9 September
In de eerste twee weken wordt het plan van aanpak opgezet, wordt er een start gemaakt met het opstellen van de requirements en wordt er onderzoek gedaan naar de werking van het huidige software pakket (Het systeem LetsGrow).
Mijlpalen:
- **1.** Definitieve versie Plan van Aanpak
- **2.** Eerste versie requirements lijst.

Week 3 en 4 – Iteratie 1: 12 September t/m 23 September
In iteratie 2 wordt er onderzoek gedaan naar bestaande dashboard pakketten en wordt er een klein onderzoekje uitgevoerd naar de aanlevering van data uit externe bronnen. Verder is hier een eerste versie opgezet van het functioneel en technisch ontwerp.
Mijlpaal:
- **3.** Resultaat onderzoek pakketselectie

Week 5 t/m 14 - Iteraties 2 t/m 6: 26 September t/m 2 December
In de weken 5 t/m 14 wordt het ontwikkelproces iteratief uitgevoerd. Hierbij zal steeds (indien nodig) het functioneel- en technisch ontwerp worden aangepast en er zal een deel ontwikkeld worden. Verder zal iedere iteratie het detailtestplan, de testontwerpen en het testrapportage worden uitgebreid, om zo iedere iteratie te eindigen met een werkend product. De structuur van de iteraties zal gelijk zijn, maar de daadwerkelijke implementatie van de sprintbacklog zal na iedere iteratie opnieuw bepaald worden met behulp van de productbacklog. De productbacklog is gebaseerd op de requirements en de overige activiteiten die nodig zijn om tot een werkend dashboard te komen. Deze productbacklog wordt bijgehouden op de muur in het kantoor waar ik werk (zie afbeelding 5). Een overzicht van de invulling van de iteraties is te vinden in bijlage F: “Productbacklog”. Deze bijlage is iedere iteratie aangevuld.
Mijlpalen:
- **4.** Definitieve versie functioneel ontwerp
- **5.** Definitieve versie technisch ontwerp
- **6.** Definitieve versie testontwerpen
- **7.** Definitieve versie testrapportage

Week 15, 16 en 17: 5 December t/m 23 December
De laatste drie weken zullen besteed worden aan het afronden van de documentatie en het opleveren van de producten.
Mijlpalen:
- **8.** Definitieve versie afstudeerverslag

3.2 Vaststellen requirements

Zoals in hoofdstuk 3.1 is besproken, is het niet mogelijk om direct alle requirements te kunnen bepalen. Zo is het mogelijk dat er in de loop van de tijd verschillende requirements bij kunnen komen die de prioriteit van de andere requirements naar beneden zullen brengen.
De requirements zijn voornamelijk opgesteld op basis van de gesprekken met Leon en Peter. De notulen van deze gesprekken zijn te vinden in bijlage E (Gesprek 1 t/m 4). Deze gesprekken zijn gehouden in de week van 30 augustus. De exacte details staan in het gesprek beschreven.

Functionele requirements

Nr.	Requirement	Prioriteit	Bron
1.1	Het moet mogelijk zijn om een nieuwe tegel toe te voegen aan het dashboard	Must have	Gesprek 1
1.1.1	Het moet mogelijk zijn om een rapport toe te voegen aan het dashboard	Must have	Gesprek 1
1.1.2	Het moet mogelijk zijn om een grafiek toe te voegen aan het dashboard	Must have	Gesprek 1
1.1.3	Het moet mogelijk zijn om een afbeelding toe te voegen aan het dashboard	Must have	Gesprek 2
1.1.4	Het moet mogelijk zijn om een meter toe te voegen aan het dashboard	Must have	Gesprek 4
1.2	Het moet mogelijk zijn om een tegel te verwijderen van het dashboard	Must have	Gesprek 1
1.3	Het dashboard moet de mogelijkheid bieden om terug te keren naar het oude klantmenu	Must have	Gesprek 1
1.4	Het moet mogelijk zijn om meerdere dashboards aan te maken	Must have	Gesprek 2
1.5	Het moet mogelijk zijn om tussen meerdere dashboards te wisselen	Must have	Gesprek 2
1.6	Het moet mogelijk zijn om een dashboard een naam te geven	Should have	Gesprek 2
1.7	De tegels op het dashboard moeten kunnen worden verplaatst	Should have	Gesprek 1
1.8	De tegels op het dashboard moeten kunnen worden open en dichtgeklapt	Should have	Gesprek 1
1.9	Het dashboard moet de mogelijkheid bieden om het weer te kunnen tonen	Should have	Gesprek 3
1.10	Het dashboard moet de mogelijkheid bieden om twitterberichten te kunnen tonen	Should have	Gesprek 3
1.11	Indien er op een grafiek geklikt wordt op het dashboard wordt deze vergroot en kan de gebruiker hier op inzoomen	Should have	Gesprek 2
1.12	Het moet mogelijk zijn om een dashboard te delen met andere gebruikers	Should have	Gesprek 4
1.13	Het dashboard moet de mogelijkheid bieden om informatie van de veiling te kunnen tonen	Could have	Gesprek 3
1.14	Het dashboard moet de mogelijkheid bieden om het laatste nieuws te kunnen tonen.	Could have	Gesprek 3
1.15	De tegels op het dashboard moeten kunnen worden vergroot en verkleind	Could have	Gesprek 1

Tabel 3: Initiële functionele requirements

Niet-functionele requirements

Nr.	Requirement	Prioriteit	Bron
2.1	Het dashboard moet direct getoond worden nadat de klant in logt	Must have	Gesprek 1
2.2	Het dashboard moet responsive zijn	Must have	Gesprek 1
2.3	Het dashboard moet ontwikkeld worden met een veelgebruikte programmeertaal	Must have	Gesprek 3
2.4	Het dashboard moet ontwikkeld worden met behulp van de laatste technieken	Must have	Gesprek 3
2.5	Het dashboard moet ontworpen zijn/worden volgens de design standaarden	Must have	Gesprek 3
2.6	Het dashboard moet in weinig klikken te bedienen zijn	Must have	Gesprek 1
2.7	Een dashboard tegel moet functioneel beperkt zijn, mag maar een ding doen	Must have	Gesprek 1
2.8	Het dashboard moet er uitnodigend uit zien	Must have	Gesprek 1
2.9	Het dashboard moet eenvoudig te bedienen zijn	Must have	Gesprek 1
2.10	De content op het dashboard moet worden onderverdeeld in tegels	Must have	Gesprek 1
2.11	Het dashboard moet de stijl van LetsGrow implementeren.	Must have	Gesprek 1
2.12	Het dashboard moet in meerdere talen beschikbaar zijn	Should have	Gesprek 1
2.13	Het dashboard moet uitgebracht worden in app vorm	Could have	Gesprek 3
2.14	Het dashboard moet geïntegreerd worden met het huidige klantmenu	Won't have	Gesprek 1

Tabel 4: Initiële niet-functionele requirements

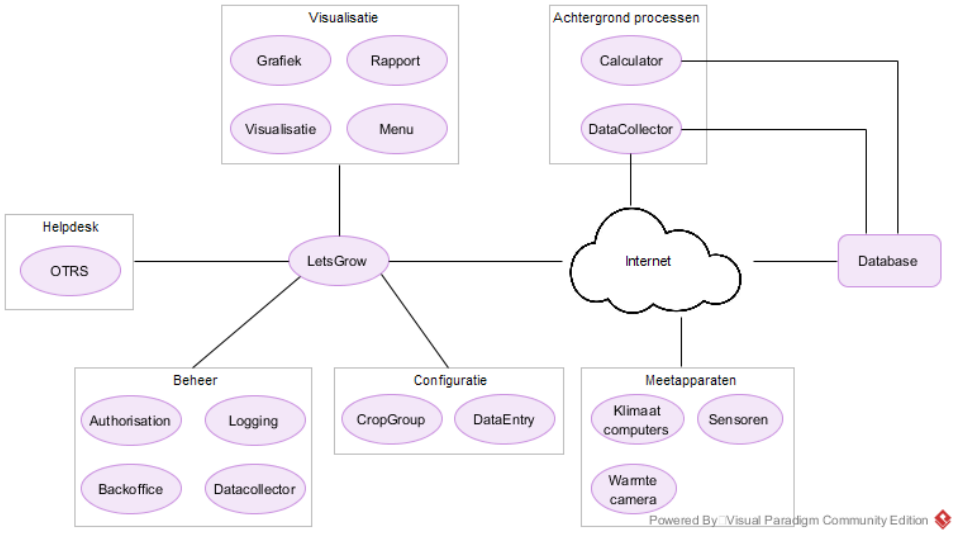
3.3 Onderzoek werking huidige applicatie LetsGrow

Tijdens deze iteratie is ook tijd besteed aan het uitzoeken van de werking van LetsGrow. LetsGrow bestaat uit verschillende kleinere applicaties die samen een grote applicatie vormen.

De startpagina van LetsGrow.com draait op Joomla. Zodra de klant in logt op LetsGrow.com wordt er een web frame geladen met daarin de menu applicatie van LetsGrow. Deze applicatie zorgt ervoor dat de gebruiker links in zijn/haar scherm een menu ziet waar doorheen geklikt kan worden. Zodra de gebruiker een grafiek aanklikt, wordt er een andere applicatie van LetsGrow geopend, namelijk de grafiek applicatie. Deze grafiek applicatie is verantwoordelijk voor het ophalen van de data en het tonen van deze data in een grafiek.

Binnen LetsGrow wordt er gebruik gemaakt van definitie objecten. Vrijwel alles in LetsGrow wordt gezien als een definitie. Een definitie object bevat een type (bijvoorbeeld grafiek, rapport, gebruikersprofiel) en een referentie naar welke data er aan de definitie gekoppeld is. Zo is bijvoorbeeld een grafiek definitie op zijn beurt weer gekoppeld aan een collectie regel. Deze collectie regels bevatten de meetgegevens (data) die onder andere gebruikt wordt om een grafiek te kunnen tonen aan de klant. Zo kan een definitie van een grafiek ook gekoppeld zijn aan meerdere collectie regels, aangezien een grafiek meerdere leeslijnen kan bevatten. Om de grafiek te tonen wordt het id van de grafiek definitie meegegeven aan de grafiek applicatie, die daarmee vervolgens de meetgegevens ophaalt en toont als een grafiek.

De gehele applicatie van LetsGrow werkt volgens deze manier, zo ook bijvoorbeeld de rapport applicatie. De rapportapplicatie verwacht net als de grafiek applicatie als parameter een definitie id. Ook hier worden aan de hand van dit id de bijbehorende meetgegevens opgehaald en aan de klant getoond.



Afbeelding 6: Overzicht werking LetsGrow

Binnen LetsGrow wordt er gebruik gemaakt van een menuboom. (zie afbeelding 7) Door middel van deze menuboom kan de klant navigeren naar bijvoorbeeld zijn of haar grafieken en rapporten, om deze vervolgens te kunnen bekijken. Deze menuboom is opgedeeld in drie categorieën, namelijk modules, teeltgroepen en overige definities

De eerst categorie zijn de modules, die door de gebruiker aangeschaft kunnen worden bij LetsGrow om bepaalde meetgegevens te registreren. Een module bevat een aantal voor-geconfigureerde grafieken en rapporten waarin de klant zijn meetgegevens kan zien. De tweede categorie zijn de teeltgroepen. Een teeltgroep is een verzameling grafieken en rapporten die gedeeld kan worden met verschillende mensen. De beheerder van de teeltgroep kan grafieken aanmaken en wijzigen. De overige leden van de teeltgroep hebben alleen rechten om deze grafieken en rapporten te bekijken. De derde categorie zijn de overige definities. Hier kunnen klanten naar eigen wens grafieken en rapporten aanmaken en configureren.

Om de applicatie en data te beveiligen, wordt er gebruik gemaakt van een sessie id. Dit id wordt gegenereerd op het moment dat de gebruiker inlogt op Joomla. Vervolgens wordt bij het openen van iedere pagina het Sessie ID meegegeven om te controleren of de gebruiker de juiste rechten heeft om de data te mogen bekijken. De parameters, zoals het definitie id en het sessie id worden aan de web URL meegegeven bij het openen van bijvoorbeeld een grafiek of rapport.

Het achtergrond proces “datacollector” uit afbeelding 6 is verantwoordelijk voor het ophalen van de data uit de meetapparaten en schrijft deze vervolgens weg in de database. De “calculator” voert bijvoorbeeld, op basis van deze data, berekeningen uit voor onder andere de plantmodellen.

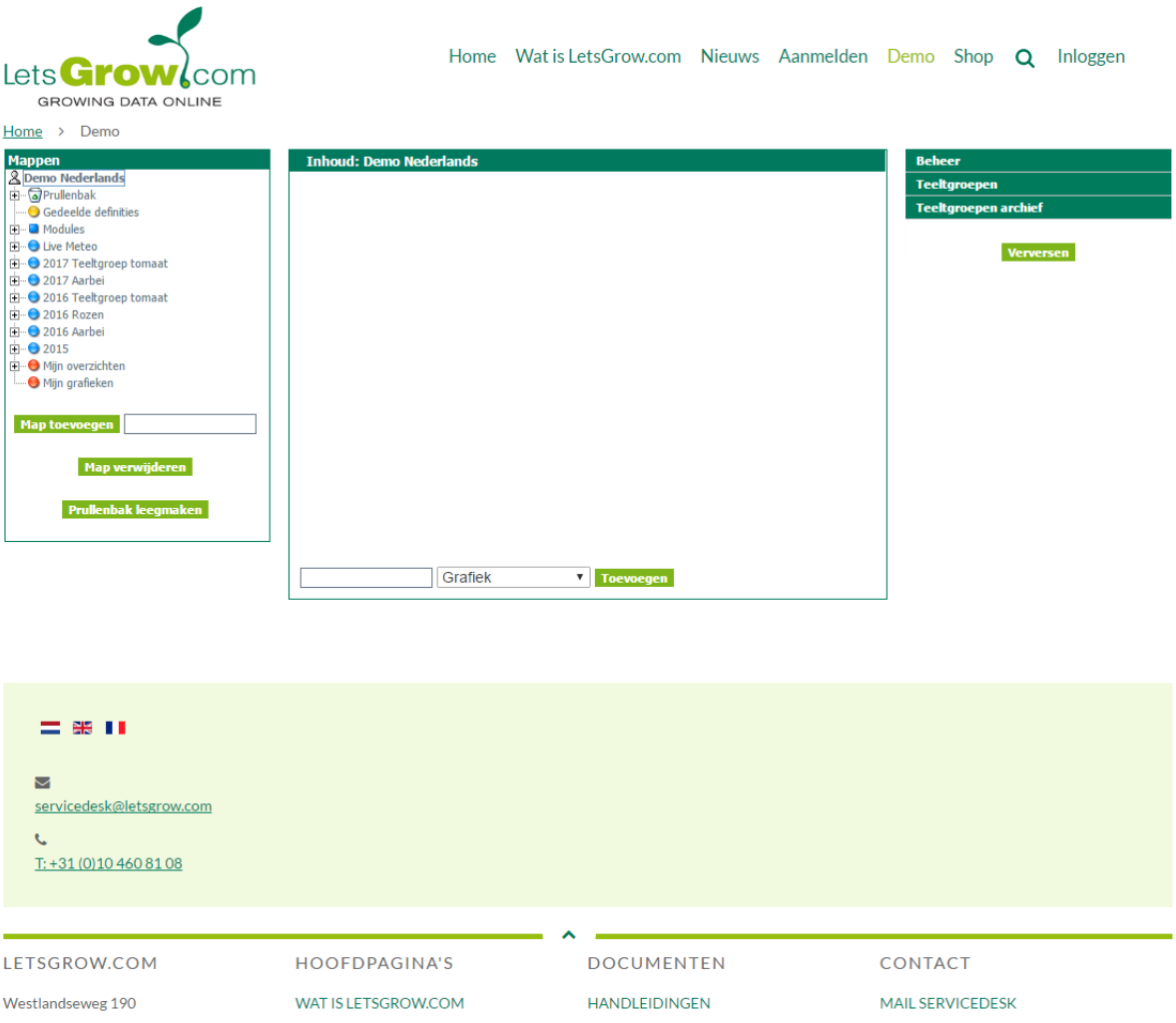
De gehele omgeving van LetsGrow is redundant opgezet. Dit geldt voor de database, services, active directory en storage. Dit houdt bijvoorbeeld in dat wanneer een van de twee databases uit valt, de

andere database het werk overneemt. In bijlage A wordt een volledige overzicht gegeven van de infrastructuur van de LetsGrow omgeving.

De URL hieronder is een vereenvoudigde weergave van een URL om een grafiek te openen:
http://charts.letsgrow.com/Default.aspx?DefId=57384&SID=17040ed5-d0e9-423a-b10f-d00b4850c7ca

Ik zal dus ook moeten werken met deze constructie, aangezien ik ook data uit deze applicaties op ga halen om te kunnen tonen op het dashboard. Om de autorisatie te regelen van LetsGrow wordt er gebruik gemaakt van een autorisatie applicatie die ook ontwikkeld is door LetsGrow.

Op LetsGrow.com is er een demo beschikbaar waar eventuele nieuwe klanten een eerste indruk kunnen krijgen van de werking en functionaliteiten van LetsGrow. Deze demo is beschikbaar op:
http://www.letsgrow.com/nl/demo



Afbeelding 7: Joomla website waar het LetsGrow menu in draait

3.4 Mijn werkzaamheden

De focus van mijn werkzaamheden tijdens het afstudeertraject ligt op het ontwikkelen en implementeren van een dashboard, waar de grafieken en rapporten van LetsGrow op getoond kunnen worden. Er zal een onderzoek gedaan moeten worden om te achterhalen of er een dashboard pakket bestaat dat ik als basis voor het LetsGrow dashboard kan gebruiken. Indien dit niet het geval is, zal er een nieuwe dashboard applicatie ontwikkeld moeten worden.

In eerste instantie zullen bestaande grafieken van LetsGrow geselecteerd en getoond moeten kunnen worden op het dashboard. Het uiterlijk van de dashboard applicatie zal zo ontworpen moeten worden dat het overeen komt met de overige applicaties van LetsGrow.

3.5 Problemen

In verband met wat kleine problemen met de accounts binnen LetsGrow duurde het wat langer voordat ik toegang kreeg tot de Team Foundation Server(TFS) van Microsoft. Om toch de huidige code te kunnen bestuderen is er een offline kopie gemaakt, waarin ik de opbouw en interne werking van de applicatie kon inzien.

Na veel interne mailwisseling en nadat ik bij verschillende mensen ben langs geweest is kon dit probleem uiteindelijk toch worden opgelost en had ik toegang tot de TFS server.

3.6 Mijlpalen en evaluatie

In deze iteratie zijn er twee mijlpalen behaald, namelijk:

Nummer	Mijlpaal
1	Definitieve versie plan van aanpak
2	Basislijst requirements

De eerste twee weken stonden voornamelijk in het teken van voorbereiding. De werkplek werd ingericht en het plan van aanpak is tot stand gekomen. Er zijn hierin verschillende keuzes gemaakt, zoals welke ontwikkelmethode, versiebeheertool, modelleringstechniek en testtechniek er gebruikt gaan worden.

Er is gekozen om niet de exacte ontwikkelmethodiek van LetsGrow over te nemen, omdat ik tijdens het ontwikkelproces niet gehinderd zal worden door tussentijdse klantwensen en mij kan focussen op de ontwikkeling van het dashboard. Hierdoor is het voor mij wel mogelijk om steeds iteraties van twee weken in te plannen en dit zorgt voor wat meer structuur tijdens het ontwikkelproces.

De testmethodiek kon niet worden overgenomen van het bedrijf, aangezien bij LetsGrow volgens eigen interpretatie wordt getest. Doordat er voor de afstudeeropdracht op een iets meer gestructureerde manier getest zal moeten worden is gekozen voor een andere testmethodiek, namelijk TestGoal (de Grood, 2012).

Deze iteratie is, op het probleem beschreven in hoofdstuk 3.5 na, geheel volgens planning verlopen en de geplande mijlpalen zijn behaald.

4. Iteratie 1: Onderzoeken en uitzoeken

Deze iteratie is onderzoek gedaan naar een geschikt dashboard pakket voor LetsGrow. De aanleiding van het onderzoek is dat er in verband met de beperkte tijd een keuze gemaakt moeten worden wat betreft de prioriteiten en haalbaarheid van het afstudeertraject.

Als er gekozen zou worden voor het ontwikkelen van een nieuw dashboard zal er geen of erg weinig tijd over blijven om deze ook daadwerkelijk in te richten en te kunnen gebruiken. Aangezien software ontwikkelen tegenwoordig vaak voor een groot deel bestaat uit het combineren van bestaande functionaliteiten/systemen, is het verstandig om te kijken of er al een basis bestaat die verder uitgebreid kan worden. Ook bij de ontwikkelaars van LetsGrow gaat de voorkeur uit naar het gebruiken van een bestaand systeem (indien beschikbaar), zodat er meer tijd over blijft voor het implementeren van functionaliteit. De sprintbacklog bevat deze iteratie: Onderzoekplan opzetten, Onderzoek uitvoeren, Opstart maken functioneel ontwerp, opstart maken technisch ontwerp, onderzoekje aanlevering data uit externe bronnen.

4.1 Onderzoek plan

Het onderzoek is opgedeeld in twee delen, namelijk het onderzoeksplan en het onderzoeksrapport. Het onderzoeksplan is opgezet ter voorbereiding van het onderzoek en in het onderzoeksrapport wordt het daadwerkelijke onderzoek beschreven.

Probleem- en doelstelling

Er zijn verschillende dashboard pakketten te vinden om te implementeren, maar er zal moeten worden bepaald of deze pakketten voldoen aan de wensen van de opdrachtgever. Mocht dit het geval zijn, zal er een bestaand dashboard pakket geïmplementeerd en aangepast gaan worden voor LetsGrow. In het geval dat er geen dashboard pakket beschikbaar is die voldoet aan de wensen van de opdrachtgever zal er een nieuwe dashboard applicatie ontwikkeld dienen te worden.

Het doel van dit onderzoek is om te bepalen of er een dashboard pakket beschikbaar is, dat voldoet aan de wensen van de opdrachtgever.

Probleemeigenaren en doelgroep

Het project kent slechts een probleem eigenaar, namelijk:

- De opdrachtgever

De opdrachtgever is namelijk de persoon die wil weten of er een dashboard pakket beschikbaar is, of dat er een nieuwe dashboard applicatie ontwikkeld dient te worden. Aangezien het onderzoek een onderdeel van een intern project binnen LetsGrow, zal de uitkomst van het onderzoek ook slechts bedoeld zijn voor de medewerkers van LetsGrow.

Onderzoeksmethode

Er zijn diverse pakket selectie methodes beschikbaar, dus er moest gekeken worden welke methode het beste binnen dit onderzoek toegepast kon worden. Er zijn twee veelgenoemde pakketselectie methodes vergeleken en hier is uiteindelijk een keuze uit gemaakt. De twee vergeleken methodes zijn KPMG en Indora.

De keuze is uiteindelijk op de KPMG methode gevallen, aangezien deze methode op grote schaal wordt gebruikt en hier veel informatie over te vinden is. De vergelijking van deze twee methodes wordt

beschreven in hoofdstuk 3.2 van het onderzoeksrapport. Ik heb gekozen voor KPMG, maar heb besloten om wel gebruik te maken van de voorbeeldvragen uit het document “Indora Informatisering – Software- en leverancierselectie: een korte introductie”. Dit document bevat een zeer uitgebreide en complete vragenlijst die mij goed op weg heeft geholpen met het bepalen van de eisen van het dashboard pakket.

Onderzoeksvraag
De onderzoeksvraag die gebruikt is voor het onderzoek luid als volgt: “**Bestaat er een dashboard pakket dat aan de wensen voor de nieuwe situatie kan voldoen?”.**

Om antwoord te kunnen geven op deze vraag, worden er volgens de KPMG methode een aantal pakketten geselecteerd en vergeleken met elkaar. Indien er pakketten beschikbaar zijn die voldoen aan de wensen van de opdrachtgever, kan er gekeken worden welk pakket het beste toe te passen is om de huidige situatie te verbeteren.

4.2 Planning

In de hieronder getoonde tabel is globale planning bepaald van het onderzoekstraject.

Taak	2-Sep	5-Sep	6-Sep	7-Sep	8-Sep
Onderzoeksplan					
Vooronderzoek - **Samenstellen programma wensen en eisen** - **Knockout criteria bepalen** - **Vaststellen scope van het proces**					
Longlist vaststellen					
Shortlist vaststellen					
Opstellen en afsluiten contract					

Tabel 5: Planning onderzoek

4.3 Onderzoeksrapport

Het onderzoek heeft als doel een geschikt dashboard pakket te selecteren uit een grotere lijst met dashboard pakketten. Er is tijdens het onderzoek gebruik gemaakt van de KPMG pakketselectie methode. Ik heb deze methode niet exact gevolgd, maar ik heb de methode als richtlijn aangehouden. De reden hiervoor is dat niet alle onderdelen relevant zijn voor dit onderzoek en niet alles haalbaar is binnen de beperkte tijd die ik beschikbaar heb voor het onderzoek.

Vooronderzoek
Om het onderzoek goed uit te kunnen voeren is het belangrijk om de huidige situatie helder te beschrijven, zodat eventuele misverstanden kunnen worden voorkomen. Dit gebeurt onder andere aan de hand van het bedrijfsmodel, waarin de missie en de visie van het beschrijf worden beschreven. Het vooronderzoek is van belang, aangezien dit mogelijk een rol kan spelen in de keuze van een softwarepakket. Hier is namelijk ook gekeken naar de toekomst van LetsGrow wat betreft mobiele apps of mobiele websites. Dit wordt verder toegelicht onder het kopje “Extra onderzoek” en in hoofdstuk 2 van het bijgeleverde onderzoeksrapport.

Verder worden er in dit hoofdstuk knock-out criteria opgesteld, die bepalen welke pakketten toepasbaar zijn, om tot de gewenste situatie te komen.

Knock-out criteria
Door knock-out criteria op te stellen, kun je de eerste selectie uitvoeren en de pakketten die niet aan de belangrijkste criteria voldoen er direct uit filteren. De lijst met knock-out criteria is tot stand gekomen op basis van de huidige requirements en het vooronderzoek. Deze lijst wordt hieronder weergeven:

Nr.	Criteria
1	**Het dashboard moet responsive zijn** Hierdoor kan het dashboard op zowel een desktop als op een mobiel apparaat gebruikt worden.
2	**Het dashboard moet ontwikkeld zijn met een veelgebruikte programmeertaal** Dit is nodig voor een goede onderhoudbaarheid.
3	**Het dashboard moet eenvoudig te bedienen zijn** Zo kunnen ook de klanten met minder computer ervaring hebben met LetsGrow overweg.
4	**Klanten moeten de mogelijkheid hebben om zelf het dashboard te wijzigen** Dit ontlast de klantsupport en laat de klant een dashboard volledig naar wens inrichten.
5	**Het dashboard moet op een Windows omgeving kunnen draaien** Aangezien LetsGrow draait op een Windows server zal het dashboard dit ook moeten kunnen.
6	**Het moet mogelijk zijn om een nieuw type dashboard tegel toe te voegen aan het dashboard** Zo kunnen nieuwe datavisualisaties e.d. in de toekomst ook gebruikt worden op het dashboard.
7	**Het dashboard pakket moet nog ondersteund en doorontwikkeld worden.** Het pakket moet (beveiliging) updates en bugfixes ontvangen om actueel te blijven.
8	**Indien het dashboard pakket beschikt over een backend moet de database te integreren zijn binnen de database van LetsGrow** Het dashboard wordt onderdeel van LetsGrow, dus dan moet het ook in deze database komen.

Tabel 6: Knock-out criteria

Volgens de bron Pakketselectie: de begin- en de eindfase uitgelicht kunnen de belangrijkste eisen (requirements) gebruikt worden als knock-out criteria. Echter beschrijven deze eisen voornamelijk punten waar ieder dashboard aan voldoet. Hierdoor is gekozen om een aantal specifiekere (technische) eisen mee te nemen.

Een ander mogelijk knock-out criteria had kunnen zijn dat het pakket niet meer dan een x aantal euro mocht kosten. Dit is in dit project echter geen knock-out criteria, aangezien dit soort software eigenlijk nooit meer dan 1000 euro kost en dit voor het bedrijf geen enkel probleem is. Ook had een criteria kunnen zijn dat de applicatie over een backend moest beschikken. Dit is echter geen knock-out criteria geworden, want als het dashboard pakket aan een groot deel van de wensen voldoet is het mogelijk om dit zelf te ontwikkelen.

Verder had een knock-out criteria kunnen zijn dat het dashboard pakket goed gedocumenteerd moest zijn. Ook hier is niet voor gekozen, aangezien er vanuit gegaan wordt dat de support voldoende informatie kan verstrekken indien nodig. Deze aanname is gebaseerd op de voorwaarden die verstrekt worden op de website van de dashboard pakketten.

Extra onderzoek
Voor er begonnen kon worden aan het selecteren van een software pakket, was er de wens om een klein tweede onderzoek uit te voeren. Het doel van dit onderzoek was om te bepalen hoe de toekomst van mobiele apps eruit zag. Het was belangrijk om dit vooraf te bepalen, aangezien de uitkomst van dit onderzoek bepalend kan zijn of een pakket wel of niet zal voldoen in de nieuwe situatie.

De uitkomst van dit onderzoek is besproken met de opdrachtgever en vervolgens vergeleken met de toekomstvisie van LetsGrow. Er is binnen LetsGrow nog geen duidelijkheid wat betreft de vraag of er in de toekomst alleen nog maar met mobiele apps gewerkt ging worden of dat de webapplicatie behouden blijft. Dit onderzoek kon LetsGrow hierin adviseren en zo zijn we tot de conclusie gekomen dat een (mobiele) webapplicatie momenteel de beste keuze is voor een mobile first approach.

Uit dit onderzoekje is namelijk gebleken dat de toekomst van de mobiele apps steeds meer richting de web kant op zal gaan. Dit komt, omdat er steeds meer functionaliteiten worden ontwikkeld die web apps native functionaliteit geven. Hierbij moet gedacht worden aan het cachen van pagina's, het versturen van pushnotificaties, het op de achtergrond ophalen van content en support om de web app offline te kunnen gebruiken.

Indien de keuze was gevallen op een mobiele app, dan zou hier gekozen worden voor Xamarin. LetsGrow heeft namelijk momenteel al een aantal apps uitgebracht die ook met Xamarin zijn ontwikkeld, dus hier zou de voorkeur naar uit gaan. Echter, gezien de groeiende mogelijkheden voor webapplicaties en het feit dat LetsGrow momenteel ook een grote webapplicatie is, is er gekozen om de dashboard applicatie te ontwikkelen als een (mobiele) website. Meer informatie over dit onderzoek is te vinden in hoofdstuk 2.6 van het onderzoeksrapport.

Longlist
Vervolgens is er een lijst opgesteld met pakketten die voldoen aan de knock-out criteria. Volgens de officiële KPMG methode (Pakketselectie: de begin- en de eindfase uitgelicht(C-2002-3-Heijer) blijven hier normaal gesproken 8 á 10 pakketten over. Gezien het beperkte aanbod van pakketten is deze lijst bij mij beperkt gebleven tot 5 pakketten. De belangrijkste reden hiervoor is dat de pakketten veelal niet beschikken over een backend, waardoor ik als ontwikkelaar wel een dashboard kan opzetten en deze vervolgens aan de klanten kan opleveren, maar de klanten hebben dan vervolgens niet de mogelijkheid om deze verder naar wens aan te passen. Een pakket ombouwen, zodat de klant wel zelf dashboards kan opzetten, zou erg veel tijd kosten. Als gebleken was dat er geen dashboard pakket gevonden kon worden, had dit een optie geweest. Echter zijn er toch een aantal pakketten beschikbaar die aan de eisen voldoen, waardoor de pakketten zonder backend afgevallen. In deze fase zijn zoals hierboven beschreven 5 pakketten overgebleven. Het gaat hier om de volgende pakketten:

- JDash Framework for Asp.Net MVC
- JDash Framework for Asp.Net Web Forms
- JSlate
- Freeboard
- Dashku

Al de hierboven benoemde pakketten voldoen aan de knock-out criteria, echter zit er bij sommige pakketten een kleine beperking aan. Om te bepalen welke pakketten zonder aanpassingen het beste voldoen aan de wensen van de opdrachtgever is er een lijst opgesteld om dit goed te kunnen vergelijken. Deze lijst wordt op de volgende pagina getoond. Om te bepalen of de bovenstaande pakketten voldoen aan de knock-out criteria heb ik de pakketten, middels de beschikbare online demo's, informeel getest (om bijvoorbeeld de responsiviteit te controleren) en zijn de voorwaarden en informatie op de website doorgenomen (om de omgeving en het type database e.d. te achterhalen).

Knock-out criteria	JDash WebForms	JDash MVC	JSlate	Freeboard	Dashku
Het dashboard moet responsive zijn					
Het dashboard moet ontwikkeld zijn met een veelgebruikte programmeertaal					
Het dashboard moet eenvoudig te bedienen zijn					
Klanten moeten de mogelijkheid hebben om zelf het dashboard te wijzigen (backend & database)					
De applicatie moet op een Windows omgeving kunnen draaien					
Het dashboard pakket moet nog ondersteund en doorontwikkeld worden.					
Indien het dashboard pakket beschikt over een backend moet de database te integreren zijn binnen de database van LetsGrow					

Tabel 7, Knock-out criteria vergelijking(Onderzoeksrapport, Ron Persoon)

In deze lijst is te zien dat de pakketten van JDash volledig voldoen aan de gestelde knock-out criteria. De pakketten JSlate, Freeboard en Dashku voldoen ook al deze criteria, maar op enkele punten nog wel aandacht nodig hebben. Aangezien de insteek van het dashboard ook grotendeels de eenvoudige bediening is, is het erg belangrijk dat het dashboard pakket hier aan voldoet. Aangezien JSlate en Dashku hier beperkter in zijn dan de overige pakketten blijven er drie pakketten over:

- JDash Framework for Asp.Net MVC
- JDash Framework for Asp.Net Web Forms
- Freeboard

Voor gedetailleerdere uitleg over de pakketten verwijs ik naar hoofdstuk 3 in het onderzoeksrapport.

Shortlist

Officieel zou er bij de KPMG methode nu een lijst overblijven van zo'n 5 pakketten. In mijn geval zijn dit er 3 geworden. Ik heb deze pakketten dieper onderzocht en bepaald welk pakket het beste in de huidige situatie geïmplementeerd kan worden. Het doel van deze stap is het wegstrepen van nog twee pakketten om zo het beste pakket over te houden.

Ten eerste is bepaald in welke ontwikkeltaal de dashboard pakketten ontwikkeld zijn. Het bleek dat Freeboard slechts gebaseerd was op html en css en dat beide JDash frameworks gebaseerd waren op Asp.net.

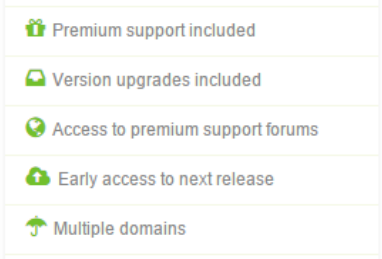
De shortlist bevatte echter twee varianten van het JDash pakket. Ook hiervan is bepaald of een van deze twee pakketten een optie zou kunnen zijn. Het overgrote deel van de applicaties van LetsGrow is ontwikkeld met behulp van Web Forms. Het leek er op dat dit dan ook de beste keuze zou zijn uit de drie overgebleven pakketten. Echter werd na het onderzoekje over de toekomst van Web Forms (Zie hoofdstuk 4.1 van het onderzoeksrapport) vrij snel duidelijk dat de kans groot is dat Web Forms langzaam zal worden uitgefaseerd. In het nieuwe .Net Framework wordt hier al geen support meer op gegeven. Met deze kennis zou het niet verstandig zijn om deze oude techniek te gebruiken.

Er bleven nu nog twee pakketten over, namelijk Freeboard en het JDash Asp.Net MVC pakket. Ik heb hier gekozen om door te gaan met het JDash Asp.Net MVC pakket, aangezien dit ontwikkeld is in C# en hier een volledig backend achter draait. Freeboard is in dat opzicht veel beperkter en de bediening van dit dashboard pakket was ook niet zo eenvoudig als JDash. Tevens zou het veel tijd kosten om hier zelf een backend achter te ontwikkelen, iets dat bij JDash reeds gebeurd was.

Achteraf gezien had een knock-out criteria kunnen zijn dat het pakket niet uit een onstabiel land mag komen. Het geselecteerde dashboard pakket is namelijk ontwikkeld in Turkije. Echter, op basis van de voorwaarden op de site van JDash, zal dit naar verwachting geen problemen opleveren.

Contract

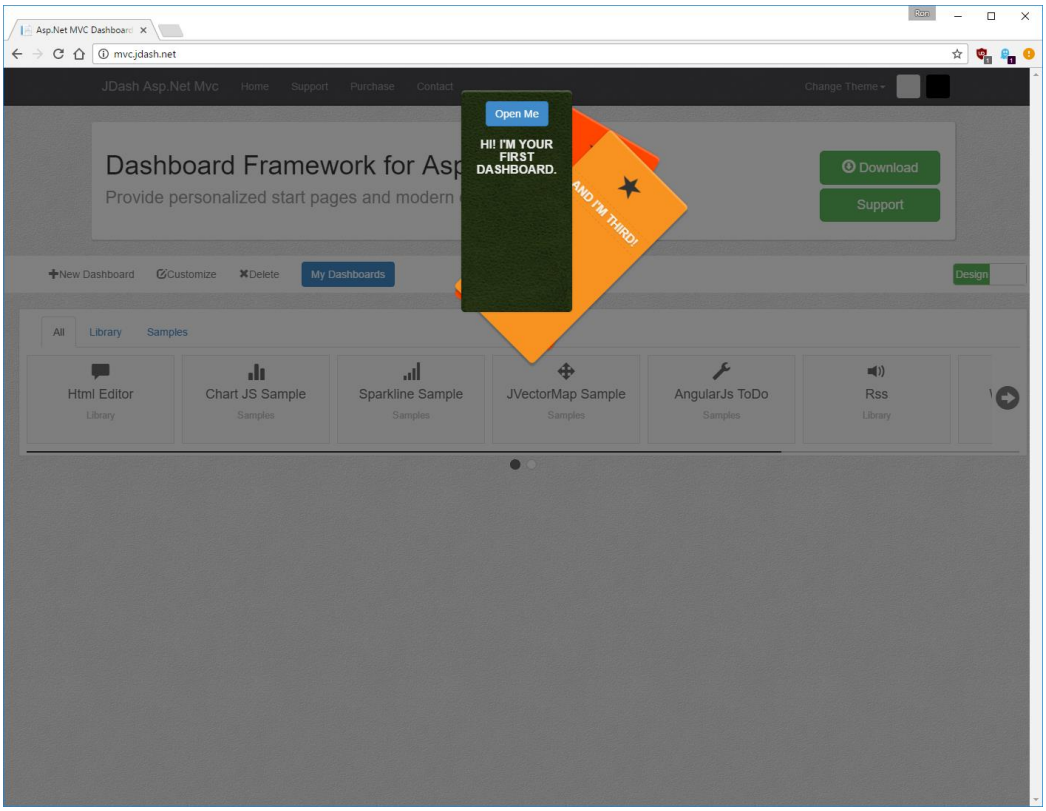
Volgens KPMG wordt er na de pakketkeuze onderhandeld en een contract afgesloten. Dit is bij dit onderzoek niet echt van toepassing, aangezien dit pakket eenmalig wordt aangeschaft en er per jaar betaald wordt voor de diensten die via de website worden aangegeven, namelijk



Afbeelding 8: Beloftes JDash
Bron: www.jdash.net

Ook prijsonderhandelingen zijn niet aan de orde in mijn situatie. JDash hanteert vaste prijzen voor de dashboard pakketten, maar bied tegen extra betaling wel de mogelijkheid om maatwerk te leveren. Aangezien hier geen gebruik gemaakt van gaat worden, is dat in onze situatie niet van toepassing.

Afbeelding 9 toont de lay-out van het geselecteerde dashboard pakket.



Afbeelding 9: JDash dashboardselectie scherm (www.jdash.net)

4.4 Demo

Om uiteindelijk over te gaan op het aanschaffen van het pakket, heb ik aan het gehele LetsGrow team een demo gegeven die de werking van het geselecteerde dashboard pakket demonstreert. Hierbij heb ik zo goed mogelijk proberen uit te leggen welke mogelijkheden dit pakket bied en waar het uiteindelijk in zal resulteren. Dit bood mij de mogelijkheid om onduidelijkheden weg te nemen en het pakket extra toe te lichten.

De reacties op deze demonstratie waren zeer positief en het pakket is direct na deze demonstratie aangeschaft.

4.5 Functioneel ontwerp

In de eerste iteratie was de lijst met requirements opgesteld, maar in deze iteratie is deze lijst requirements uitgewerkt tot functioneel ontwerp. Ik heb hiervoor gekozen, aangezien de use case beschrijvingen en de storyboards konden afhangen van de uitkomst van de pakketselectie.

Gebruikersrollen

De LetsGrow applicatie kent twee gebruikersrollen, namelijk de klant (die LetsGrow gebruikt om zijn/haar meetgegevens te bekijken en de beheerder (de medewerkers van LetsGrow). Zonder tussenkomst van een beheerder heeft de gebruiker de mogelijkheid om het dashboard volledig naar zijn wens in te richten en data in te kijken. LetsGrow heeft een rol als beheerder, waarbij LetsGrow vooraf een geconfigureerd dashboard op zou kunnen zetten wanneer een klant een module koopt.

De energie managers die ook gebruik maken van LetsGrow vallen onder de categorie klanten, aangezien zij, net als klanten, bepaalde rechten hebben om informatie wel of niet in te mogen zien.

Use case

Hieronder wordt de use case van de dashboard applicatie weergeven, die is opgesteld vanuit de requirements. Deze use case zal vermoedelijk tijdens het ontwikkeltraject meerdere keren wijzigen, aangezien er nieuwe requirements bij kunnen komen. Dit betreft de initiële versie van het use case diagram. De laatste versie van dit diagram is opgenomen in het Functioneel ontwerp dat is bijgevoegd als los document. De groen gekleurde use cases zijn de functionaliteiten die (deels) door mij ontwikkeld zullen worden.

De nummers in de use cases refereren naar de functionele requirements tabel.

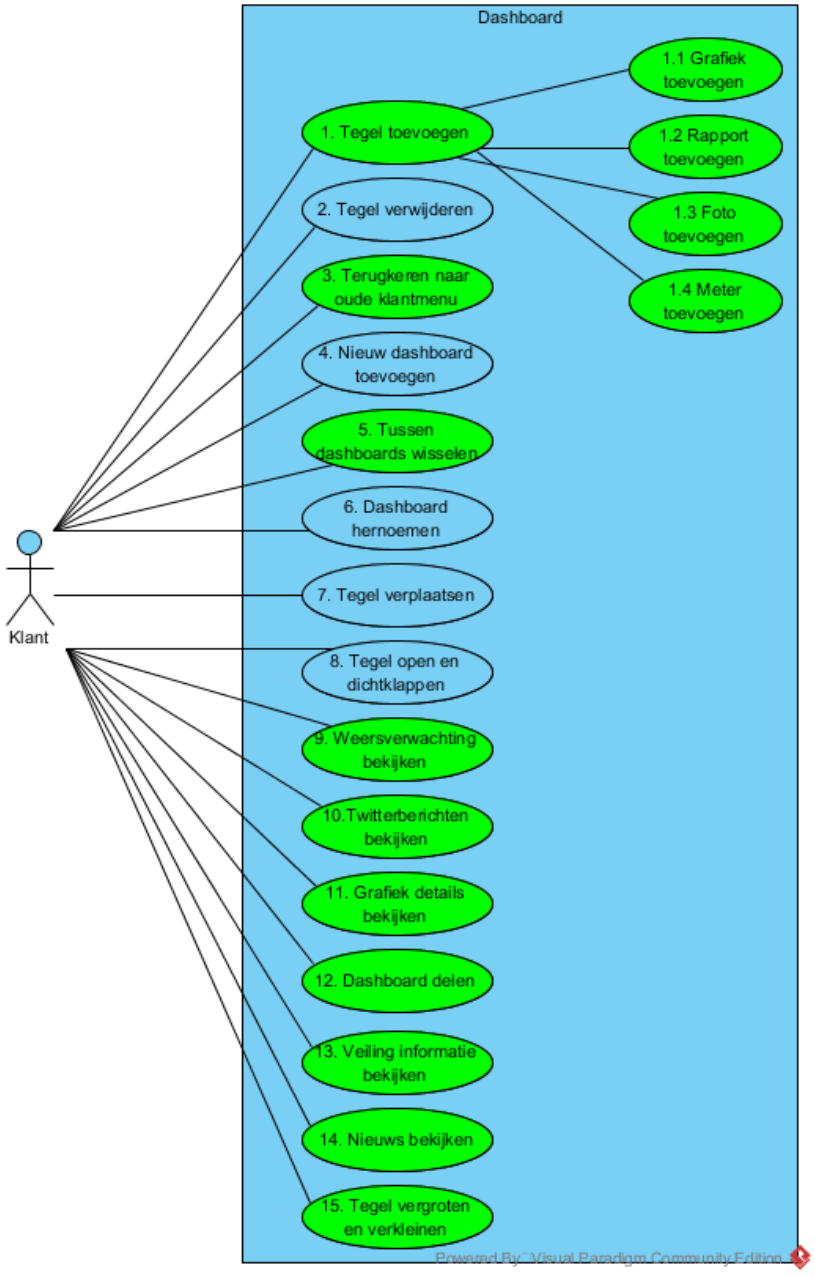


Diagram 1: Use case diagram

Risico's
LetsGrow kent een aantal risico's die de 24/7 beschikbaarheid kunnen doen falen. Een aantal van deze risico's kunnen er buiten mijn macht om voor zorgen dat de applicatie niet of beperkt functioneert.

Externe risico's

Nummer	Risico	Uitleg
1	Geen internet verbinding beschikbaar	De klant zal moeten beschikken over een werkende internetverbinding om het dashboard te kunnen gebruiken.
2	Externe webapplicaties zijn offline	Indien de externe webapplicaties offline zijn, zoals de applicatie die de grafieken genereert, dan kunnen de grafieken niet op het dashboard worden getoond.
3	De database is offline	Mocht het voorkomen dat de database offline gaat, dan zal het dashboard niet bruikbaar zijn. De reden hiervan is dat alle data van rapporten, grafieken en afbeelding worden opgeslagen in de database.

Tabel 8: Externe risico's LetsGrow

Nieuwe risico's
Doordat er een bestaand dashboard pakket is geselecteerd, dat binnen LetsGrow geïmplementeerd kan worden, zijn er een aantal risico's bijgekomen die er voor kunnen zorgen dat het project faalt. Deze risico's zijn

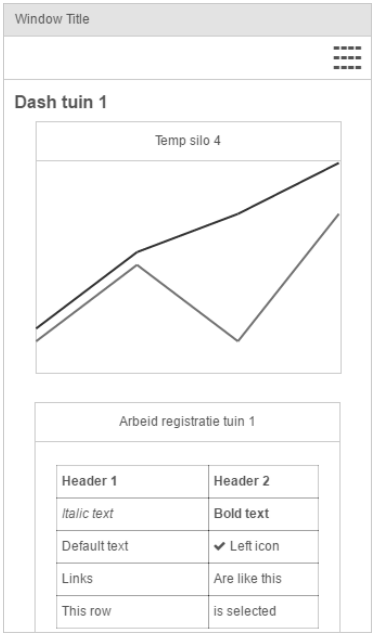
Risico	Kans	Impact	Oplossing
Support dashboard pakket valt weg	Klein	Midden	Indien de support voor het dashboard pakket weg valt kan het project mogelijk vertraging oplopen, aangezien bepaalde functionaliteiten meer tijd kosten om te implementeren. Mocht dit voor dusdanige problemen zorgen zal er contact opgenomen worden met school.
Uitkomst pakketselectie blijkt niet bruikbaar	Klein	Klein	Mocht het geselecteerde dashboard achteraf niet bruikbaar zijn, dan zullen de prioriteiten van de requirements hoogstwaarschijnlijk wijzigen, aangezien er minder functionaliteit gerealiseerd kan worden binnen de beschikbare tijd.

Tabel 9: Nieuwe risico's

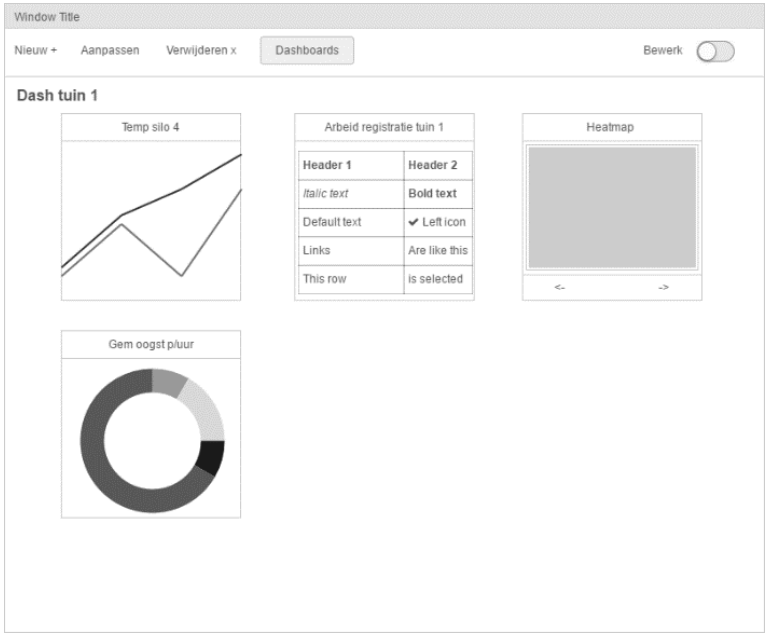
Storyboards
Om de opdrachtgever een goede indruk te geven van het dashboard zijn er een aantal storyboards gemaakt. De reden dat hier geen screenshots van het gekozen dashboard pakket in zijn opgenomen is dat het gekozen dashboard pakket op een aantal vlakken nog wat wijzigingen zal moeten ondergaan om aan de wensen van LetsGrow te voldoen.

Er zijn storyboards gemaakt voor de mobiele weergave, maar ook voor de desktop weergave. Er wordt eerst gefocust op het ontwikkelen voor de mobiele apparaten, maar aangezien het om een responsive webapplicatie gaat zal het waarschijnlijk weinig moeite kosten om de desktop weergave werkend te krijgen.

Deze storyboards zijn gebaseerd op het ontwerp van het geselecteerde dashboard pakket, maar zijn door mij deels aangepast om aan de niet-functionele requirements 2.8 en 2.11 te voldoen. Deze requirements hebben te maken met de lay-out van het dashboard. Voor meer storyboards verwijs ik naar hoofdstuk 9 van het functioneel ontwerp.



Afbeelding 10: Dashboard mobiel



Afbeelding 11: Dashboard desktop

4.6 Technisch ontwerp

Dit hoofdstuk bevat het design klassendiagram en het design sequentie diagram uit iteratie 2. Beide diagrammen bestaan voor een deel uit een of meerdere black boxen. De reden hiervoor is dat het gekozen dashboard pakket gedeeltelijk “closed-source” is en hierdoor dus niet alle klassen aanpasbaar zijn.

Klassendiagrammen

Slechts een beperkt aantal klassen van het dashboard pakket zijn voor mij interessant. Dit zijn bijvoorbeeld de klassen die ik tegenkom als ontwikkelaar en waar ik gebruik en/of wijzigingen aan zal maken. De niet interessante klassen zijn de klassen die diep binnen het dashboard pakket zijn geïmplementeerd en closed-source zijn. Ik heb er voor gekozen om het dashboard pakket (met een deel van de niet aanpasbare klassen) voor de duidelijkheid wel te modelleren. Zo kan ik een beter overzicht krijgen van het dashboard pakket en kan er direct wat documentatie op worden opgebouwd. Tevens helpt dit mee bij het leerproces van de interne werking van het dashboard.

Het dashboard pakket bestaat uit een aantal open-source klassen die hoofdzakelijk te maken hebben met het tonen van het dashboard en een aantal closed-source klassen die voornamelijk de interne werking van het dashboard pakket bepalen. De hieronder beschreven wijzigingen zijn gemaakt in het open-source gedeelte van het dashboard pakket. Dit zal dan naar verwachting dan ook geen problemen opleveren in het geval van een update, aangezien de interne werking van het dashboard pakket niet gewijzigd wordt. Diagram 2 toont het analyse klassendiagram van de dashboard applicatie.

De groene klassen zijn (deels) door mij ontwikkeld en de blauwe klassen behoren tot het dashboard pakket. Alle klassen gemodelleerd binnen de package “Jdash” zijn closed-source klassen.

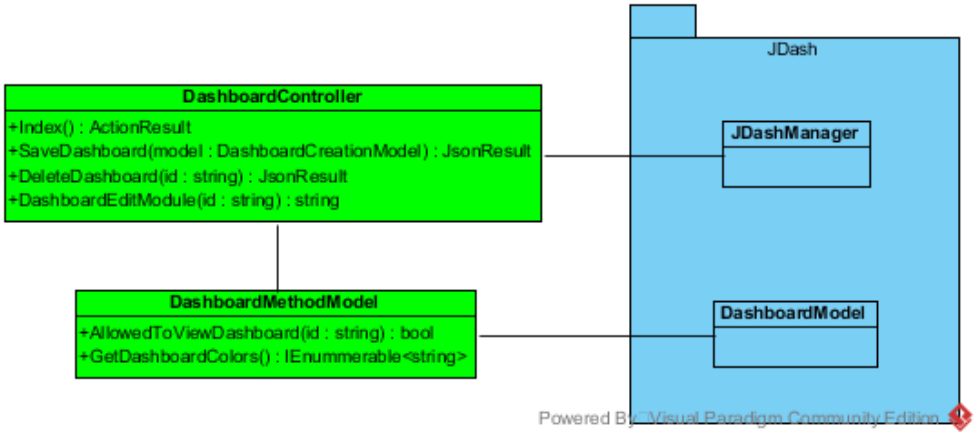


Diagram 2: Analyse klassendiagram dashboard

Dit diagram is deels tot stand gekomen via reverse engineering. Een volledig overzicht van de klassen van het dashboard pakket is te vinden in het technisch ontwerp hoofdstuk 3.

De klasse DashboardMethodModel is toegevoegd naar eigen inzicht, aangezien er naar mijn mening te veel model methodes in de controller werden gebruikt. De klasse DashboardModel is een reeds bestaande klasse uit het JDash pakket, dus hierdoor is er voor een alternatieve naam gekozen. Ik heb gekozen om voor deze klasse gebruik te maken van het singleton pattern, aangezien op deze manier niet steeds een nieuw object van de klasse aangemaakt hoeft te worden en op deze manier wat informatie, zoals de username en het dashboard id tijdelijk kunnen worden opgeslagen.

Een singleton pattern wordt gebruikt om te zorgen dat er maar één instantie van een object wordt gemaakt. Een methode binnen de DashboardMethodModel zorgt dat indien er geen instantie van de klasse bestaat dat er een nieuwe instantie wordt gemaakt. Als de instantie wel bestaat wordt deze geretourneerd. De uitwerking van dit klassendiagram wordt hieronder weergeven. Een grotere weergave van dit diagram is bijgevoegd in bijlage C.

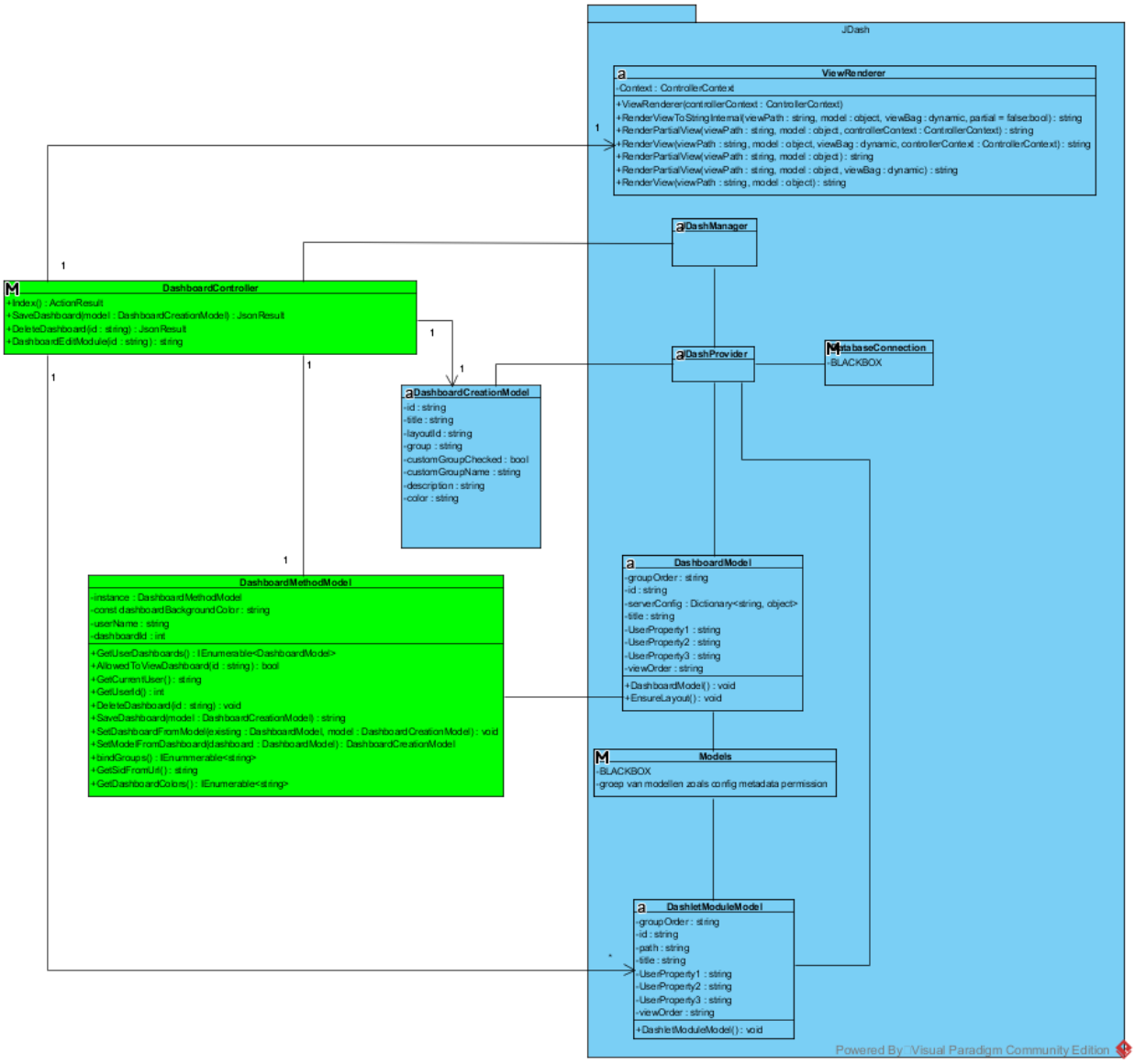


Diagram 3: Design klassendiagram dashboard

Aangezien de individuele tegels in het dashboard pakket, net als het dashboard pakket zelf, volgens de MVC structuur zijn opgebouwd, is besloten om deze in een apart klassendiagram te modelleren. Dit klassendiagram is gebaseerd op de structuur van de tegels in het geselecteerde dashboard pakket. Nagenoeg geen van de tegels uit het dashboard pakket kan gebruikt worden voor LetsGrow, aangezien er vooral LetsGrow applicaties op het dashboard getoond zullen gaan worden. De overige aanwezige tegels uit het dashboard pakket, met uitzondering van het weer, hebben (nog) geen toepassing voor LetsGrow. Hierdoor is dus slechts de structuur van de tegels is overgenomen. De implementatie van de attributen en methodes binnen de tegels zijn door mij zelf bepaald.

In het dashboard pakket gebruiken de model klassen van de dashboard tegels allemaal de zelfde attributen. Hierdoor is de keuze gemaakt om dit attribuut in een abstracte klasse te plaatsen, zodat grote methodes niet steeds opnieuw geïmplementeerd moeten worden. Het klassendiagram van de

dashboard tegels wordt hieronder weergeven. Een grotere versie van dit analysediagram is bijgevoegd in bijlage B.

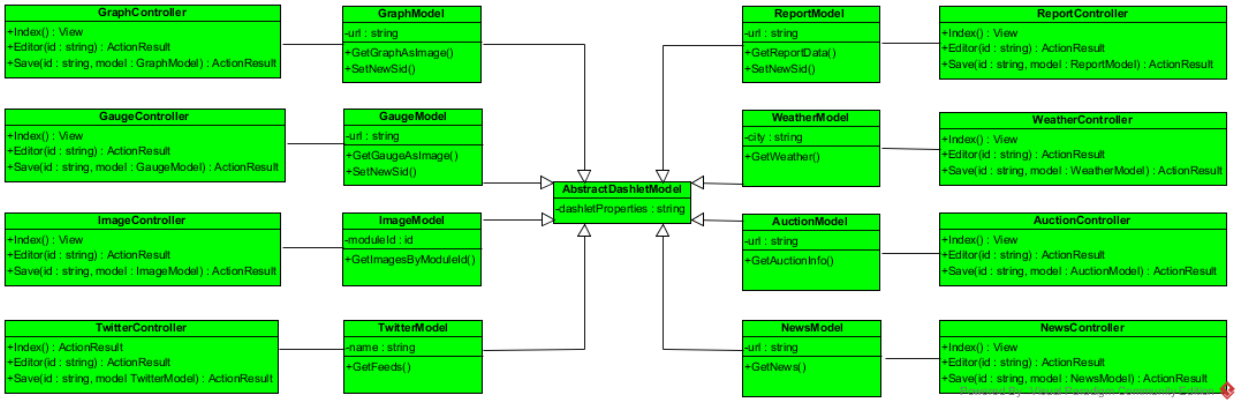


Diagram 4: Analyse klassendiagram dashboard tegels

De koppeling tussen de dashboard tegels en het dashboard verloopt via de DashboardController en de viewpagina van het dashboard. De controller haalt bij het starten van het dashboard een lijst op van alle mogelijke tegels die de klant op het dashboard kan plaatsen. De geconfigureerde dashboard tegels (die de klant al eerder op het dashboard heeft geplaatst) worden pas opgehaald op het moment dat het dashboard is geladen. Dit ophalen gebeurd na het aanroepen van een interne methode van het dashboard pakket via de viewpagina van het dashboard.

Sequentiediagrammen

Om de interne werking van het dashboard beter te modelleren zijn er een aantal sequentie diagrammen gemaakt. Een deel van het sequentiediagram is als black box gemodelleerd. Hier is voor gekozen, aangezien een gedeelte van het dashboard pakket “closed-source is”. Op deze manier is er toch een goed beeld te geven van de werking van het dashboard, zonder diep op het “closed-source” gedeelte in te gaan.

Het volgende diagram geeft de procedure bij het openen van het dashboard weer. Dit diagram is gedeeltelijk tot stand gekomen via reverse engineering en toont de werking van het geselecteerde dashboard pakket. Ook dit diagram is opgezet om het leerproces van het software pakket te versnellen en om documentatie op te bouwen voor LetsGrow. Hierdoor kunnen eventuele aanpassingen in de toekomst sneller en eenvoudiger worden doorgevoerd.

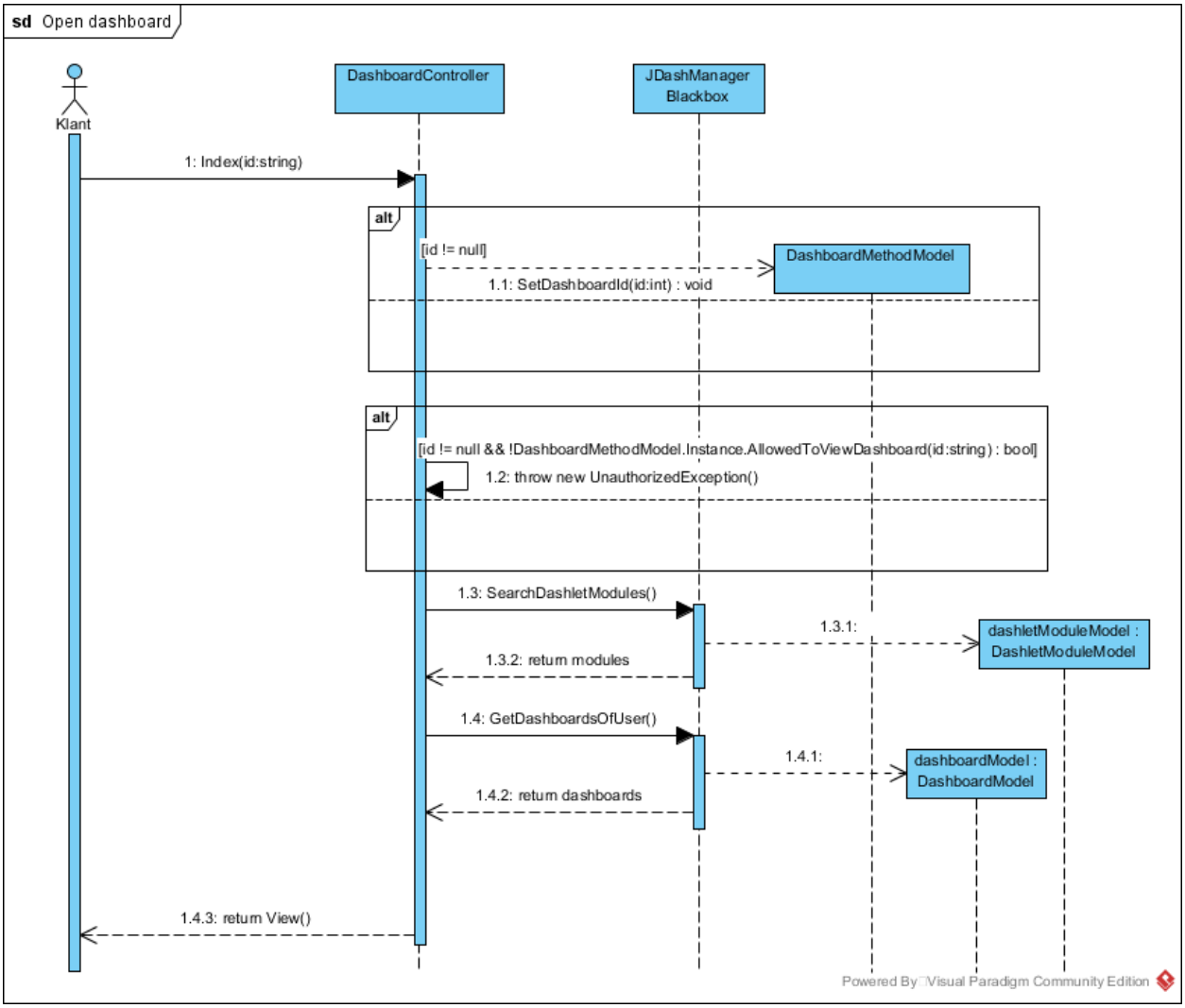


Diagram 5: Sequentie dashboard openen

Na het doorlopen van het bovenstaande sequentie diagram zijn de geconfigureerde dashboard tegels nog niet ingeladen. Dit gebeurt pas zodra de view is geladen, zoals beschreven in hoofdstuk 4.6. Meer sequentiediagrammen betreffende de dashboard applicatie zijn te vinden in het bijgeleverde document ‘Technisch Ontwerp’.

4.7 Testen

Binnen deze iteratie zijn er nog geen daadwerkelijke wijzigingen gemaakt aan het dashboard pakket. Dit zal gebeuren in iteratie 2, waarbij het pakket binnen LetsGrow zal worden geïmplementeerd. Aangezien de wijzigingen, waarbij een aantal controller methodes zijn verplaatst naar de nieuwe model klasse, naar verwachting niet voor problemen zal zorgen, zal deze wijziging op een informele manier getest worden. Dit zal gebeuren door alle functionaliteiten opnieuw uit te proberen en te controleren of deze nog functioneren zoals het deed tijdens de demo uit hoofdstuk 4.4.

4.8 Problemen

Het probleem in deze iteratie waar ik tegenaan liep, was dat het aanbod dashboard geschikte pakketten erg beperkt was. Een eerste inventarisatie leverde een flink aantal pakketten op, maar deze pakketten boden slechts de mogelijkheid om aan klanten een kant en klaar ingericht dashboard op te leveren. Slechts een beperkt aantal pakketten bood de klanten zelf de mogelijkheid om het dashboard naar wens in te richten. Dit is dan ook de reden dat de pakketselectie beperkt is gebleven tot 5 pakketten.

4.9 Wijziging met betrekking tot kerntaken

Aangezien de pakketselectie heeft geresulteerd in een dashboard pakket dat prima geschikt is om te implementeren binnen LetsGrow, zal er een kleine wijziging gemaakt worden wat betreft de kerntaken. Hier had ik gekozen om de kerntaak ontwerpen op niveau 3 te behalen, maar deze kerntaak is minder relevant geworden. Er zijn wel degelijk nieuwe functionaliteiten die ontworpen dienen te worden, maar een ander deel is weggevallen door de keuze van het dashboard pakket.

Hierdoor is ter compensatie gekozen om de kerntaak “3.5 Uitvoeren van en rapporteren over het testproces” niveau 3 te vervullen. Er wordt binnen LetsGrow zelf niet heel diep getest, dus dit kan ik nu ook opvangen binnen mijn opdracht.

4.10 Mijlpalen en evaluatie

In deze iteratie is er een mijlpaal behaald, namelijk:

Nummer	Mijlpaal
3	Resultaat onderzoek pakketselectie

Deze iteratie stond in het teken van het onderzoek en het ontwerpen. Hier is een belangrijke keuze gemaakt, namelijk het dashboard pakket dat gebruikt gaat worden. Aan de hand van een aantal knock-out criteria zijn er een aantal pakketten overgebleven waar dieper onderzoek naar gedaan is. Dit heeft uiteindelijk geresulteerd in een softwarepakket dat geïmplementeerd zal gaan worden.

De klassen- en sequentiediagrammen in de eerste versie van het technisch ontwerp zijn (gedeeltelijk) ontstaan door reverse enginering. Aan het klassendiagram van het dashboard zijn voor nu nog geen wijzigingen toegebracht, maar het klassendiagram van de dashboard tegels is volledig aangepast. Dit diagram bevat nu de klassen voor de dashboard tegels die volgens de requirements in het pakket geïmplementeerd zullen worden. Er is gekozen om het dashboard pakket gedeeltelijk ook te beschrijven en modelleren om zo voor mijzelf en LetsGrow een beter beeld te krijgen van de interne en externe werking van het pakket.

5. Iteratie 2: Opstart ontwikkeling

Iteratie 2 stond in het teken van het opstarten. Deze twee weken zijn besteed aan het implementeren van het dashboard pakket. Hierbij moet er gedacht worden aan het koppelen van de autorisatie van LetsGrow, het koppelen van modules en het inrichten van de testomgeving. De sprintbacklog bestaat deze sprint uit: Implementeren dashboard pakket in huidige situatie (TFS, Database, Autorisatie, Testomgeving) en opstellen mastertestplan.

5.1 Code richtlijnen

Bij LetsGrow wordt er ontwikkeld volgens de IfSQ Level-1 richtlijnen (Graham Bolton, Stuart Johnston, 2008). Met behulp van 'code inspection' kan er bepaald worden of er aan deze richtlijnen is voldaan. IfSQ heeft drie 'coding standards' ontwikkeld, de zogenaamde IfSQ Level-1, IfSQ Level-2 en IfSQ Level-3 richtlijnen. Deze richtlijnen kunnen gebruikt worden om zelf de kwaliteit van je code te evalueren, maar kan ook gebruikt worden als richtlijn voor peer reviews. Dit is ook het geval binnen LetsGrow. De verschillende levels geven de vereiste expertise en tijd aan, die nodig zijn om aan dit standaard te voldoen.

Level-1 beschrijft de meest logische en de meest voorkomende defect indicatoren die door software ontwikkelaars als 'bad practice' wordt gezien.

Meer informatie over de richtlijnen is te vinden op de website van IfSQ (zie bronnen).

5.2 Code reviews

Er wordt binnen LetsGrow ontwikkeld met behulp van code reviews, waarbij de IfSQ Level-1 richtlijn wordt toegepast. Na het ontwikkelen van nieuwe functionaliteit, wordt de code aangeboden ter review en zal er beoordeeld worden of de ontwikkelde code voldoet aan de richtlijnen. De code zal per functionaliteit worden gereviewd en niet per sprint, aangezien reviewen per sprint erg veel code in een keer zou opleveren wat het beoordelen bemoeilijkt. Indien de code wordt goedgekeurd bij de review kan het dashboard worden doorgezet naar de testomgeving. LetsGrow maakt gebruik van de Level-1 richtlijn, aangezien dit voor voldoende structuur zorgt en de overige richtlijnen te veel tijd zouden kosten.

De codereviews en de gemaakte aanpassingen hierop zijn van iedere iteratie beschreven in bijlage D "Codereviews".

5.3 Implementatie

Het implementeren van het dashboard pakket, binnen de huidige situatie, bestaat uit een aantal verschillende onderdelen. Het pakket zal om te beginnen op de team foundation server geplaatst moeten worden tussen de rest van de applicaties van LetsGrow, zoals de grafiek en rapport applicatie. Vervolgens zal de database van het dashboard pakket gekoppeld moeten worden aan de huidige database. Om het pakket uiteindelijk in de huidige situatie toe te kunnen passen is het van belang om de autorisatie goed in te richten. Tot slot zal de testomgeving opgezet moeten worden om de ontwikkelingen aan het dashboard pakket te kunnen testen.

Team Foundation Server

Ik ben begonnen met het plaatsen van het dashboard pakket op de Team Foundation Server (TFS) van LetsGrow. Hier was het belangrijk om te kiezen voor een duidelijke naamgeving. Er is gekozen voor de

naam DashboardUI, aangezien deze naamgeving binnen LetsGrow ook wordt gebruikt voor de andere User Interface applicaties, zoals ChartUI, de grafiek applicatie van LetsGrow en DataEntryUI, de handmatige invoer applicatie van LetsGrow. Deze naam zal tevens gebruikt worden als naamgeving voor de URL waarop het dashboard in de toekomst te bereiken zal zijn.

Database
Om de indeling van de dashboards op te kunnen slaan wordt er gebruik gemaakt van een aantal database tabellen. De tabellen waar het dashboard pakket gebruik van maakt zijn, Dashboard, Dashlet, DashletModule en EntityAuth.

- De **Dashboard** tabel bevat informatie per individueel dashboard.
- De **Dashlet** tabel bevat informatie over welke dashlet (tegel) op welk dashboard getoond wordt met welke configuratie. Deze tabel refereert naar de Dashboard en DashletModule tabel.
- De **DashletModule** bevat informatie over de identieke tegels
- De **EntityAuth** tabel bevat informatie over welke rechten de gebruiker heeft op bepaalde dashboards en dashboard tegels

In hoofdstuk 5.4.3 wordt besproken wat de reden is dat de EntityAuth tabel niet gebruikt gaat worden.

Autorisatie
Een belangrijk onderdeel van LetsGrow is de autorisatie. De meetgegevens mogen uiteraard niet door iedereen bekeken worden, dus het is erg belangrijk dat de gebruiker slechts de data kan zien waar hij/zij recht op heeft.

Het dashboard pakket biedt zelf ook een aantal mogelijkheden voor het autoriseren van de gebruikers. Tijdens het implementeren en informeel testen van de autorisatie bleek dat de autorisatie van het dashboard pakket niet goed werkte (gebruiker met de juiste rechten kreeg bepaalde informatie niet te zien waar hij/zij wel recht op had) en met het oog op de toekomst, wat betreft het delen van dashboards, is deze manier van autorisatie ook niet toepasbaar. Bij LetsGrow wordt er voor het delen van informatie ook gebruik gemaakt van hun eigen autorisatie, dus zal ik hier ook gebruik van moeten maken voor het delen van de dashboards. Tevens zou het aanpassen van de autorisatie van het dashboard pakket zorgen voor onnodig veel werk. Met deze kennis is ervoor gekozen om de autorisatie via de “Authorization module” van LetsGrow af te handelen.

Een ander voordeel van het gebruik van de LetsGrow autorisatie module is dat LetsGrow in een ontwikkelaars scherm handmatig de dashboards kan koppelen aan een bepaalde gebruiker of groep, zonder dat hiervoor direct in de database een aanpassing gemaakt hoeft te worden. In overleg met de software ontwikkelaars van LetsGrow is deze keuze uiteindelijk definitief geworden.

Tijdens het onderzoek naar de huidige situatie (hoofdstuk 3.3) bleek dat er bij LetsGrow gewerkt wordt met definities, zo ook de autorisatie. Er zal hierdoor ook voor de dashboards en dashboard tegels een nieuwe definitie gemaakt moeten worden.

Er was nog even een discussiepunt wat betreft het beheren van de autorisatie van de dashboard tegels. De autorisatie van deze tegels werkt in principe zoals het hoort in het dashboard pakket, maar voor de onderhoudbaarheid is het beter om deze gegevens op een centraal punt te kunnen opslaan en beheren. Dit is dan ook de reden om de dashboard tegels ook in de autorisatie database van LetsGrow op te slaan.

In eerste instantie zal de autorisatie van de dashboard tegels niet gebruikt worden om gebruikers te beperken in welke tegels wel en niet gebruikt mogen worden. Het implementeren van de autorisatie van dashboard tegels is puur toekomst gericht met het oog op het ontwikkelen van dashboard tegels op verzoek van de klant. In eerste instantie zal iedere gebruiker dus alle dashboard tegels kunnen en mogen gebruiken.

Doordat LetsGrow voor het autoriseren gebruik maakt van een sessie id (zie hoofdstuk 3.3), zal dit ook gebruikt moeten worden binnen het dashboard. Hierdoor zal er bij ieder verzoek naar de server vanuit het dashboard het sessie id worden meegegeven.

Inrichten testomgeving
Om de dashboard applicatie te kunnen laten reviewen zal deze ook op de testomgeving geplaatst moeten kunnen worden. Om dit te realiseren wordt het project via de build server van Microsoft Azure gebuild en vervolgens als downloadbare zip aangeboden. Dit zip bestand kan met een speciale tool van LetsGrow op de testomgeving worden geplaatst.

De reden dat LetsGrow een tool heeft om het project op de test server te plaatsen is, omdat er een aantal bestanden moeten worden hernoemd en aangepast om het op de testserver werkend te krijgen. Binnen het project worden er namelijk voor de test en productie server aparte configuratie bestanden aangemaakt. Afhankelijk van de omgeving wordt het juiste configuratie bestand klaargezet.

In deze configuratiebestanden wordt er bijvoorbeeld, afhankelijk van de omgeving, verwezen naar de andere applicaties op de testserver of productieserver. Ook kan dit configuratiebestand omgeving afhankelijke parameters bevatten.

5.4 Testen

Voordat een applicatie opgeleverd kan worden is het belangrijk om de kwaliteit van het dashboard vast te stellen. Dit is de reden dat er testen zijn uitgevoerd. Hiervoor zijn twee documenten opgesteld, namelijk het mastertestplan en het detailtestplan.

Mastertestplan
Het mastertestplan beschrijft het proces dat doorlopen is om de kwaliteit van het project vast te kunnen stellen. Hier wordt onder andere bepaald of het project voldoet aan de vooraf gestelde eisen.

Met behulp van de testrisicoanalyse wordt vastgesteld welke onderdelen problemen kunnen opleveren voor het eindresultaat. Op basis van het mastertestplan (MTP) wordt er een detailtestplan (DTP) opgezet en deze zal vervolgens (indien nodig) iedere iteratie worden bijgewerkt.

Er zal op basis van de testrisicoanalyse en kwaliteitsattributen worden bepaald op welke test ontwerptechniek er gebruikt zal gaan worden.

Definition of Done
De definition of done (DoD) is tot stand gekomen na een gesprek met Joost, de tester van LetsGrow. Hier is bepaald dat de DoD is bereikt op het moment dat de functionaliteit voldoet aan de requirements en de door mij opgezette testen succesvol zijn uitgevoerd. De DoD houdt dus in dat de functionaliteit gereed is en op de testomgeving is geplaatst, zodat Joost vervolgens een acceptatietest kan uitvoeren.

Testboom

Om de risicogebieden vast te stellen is er gebruik gemaakt van een testboom. Een testboom is een soort mind-map waarin iedere belangrijke functionaliteit wordt weergeven als een tak. Indien een tak bestaat uit meerdere belangrijke onderdelen kan deze verder worden opgesplitst. Samen met Edwin is besproken welke kwaliteitsattributen voor LetsGrow belangrijk zijn.

Diagram 6 : Testboom

Testrisicoanalyse

Een testrisicoanalyse (TRA) wordt gebruikt om de test strategie te bepalen. Er is zelden genoeg tijd om alles binnen een project te testen, dus wordt er met behulp van de TRA prioriteiten gesteld om te bepalen op welke activiteiten de focus zal komen te liggen. Zo kan er genoeg tijd gestoken worden in de onderdelen met een hoge prioriteit, zodat er vastgesteld kan worden dat de kwaliteit van het product goed genoeg is zonder alles expliciet getest te hebben.

De bepaling van het relatief belang van de TRA wordt beschreven in hoofdstuk 2 van het mastertestplan. Deze testrisicoanalyse is ingevuld door Edwin, Joost (de tester van LetsGrow) en mijzelf.

Uit deze TRA zijn vijf risicogebieden naar voren gekomen met het risico kritisch en twee risicogebieden met het risico hoog, namelijk:

Nr.	Risico gebied	Kans Edwin	Impact Edwin	Kans Joost	Impact Joost	Kans Ron	Impact Ron	Totaal Kans	Totaal Impact	Totaal	Risico
1	Grafiek tonen	1	9	1	9	3	9	5	27	135	Kritisch
10	Dashboard autoriseren	3	9	1	9	3	9	7	27	189	Kritisch
2	Meter tonen	3	5	5	5	5	5	13	15	195	Kritisch
3	Rapport tonen	3	5	3	9	3	9	9	23	207	Kritisch
9	Dashboard delen	3	9	1	9	3	9	7	27	189	Kritisch
11	Tegel autoriseren	3	5	5	3	3	5	11	12	132	Hoog
12	Terugkeren naar klantmenu	3	9	3	5	1	5	7	19	133	Hoog

Tabel 10: Test risico analyse met kritisch en hoge risico's

De volledige testrisicoanalyse is te vinden in hoofdstuk 2 van het Mastertestplan. Het mastertestplan is bijgevoegd als los document.

Teststrategie

In de teststrategie is bepaald welke testtechnieken er gebruikt gaan worden. Om dit te bepalen is er een kwaliteitsattributen tabel opgezet. Deze kwaliteitsattributen tabel is te vinden in hoofdstuk 3.3 van het mastertestplan.

Uit de kwaliteitsattributen tabel is gebleken dat functionaliteit en bruikbaarheid erg belangrijk zijn binnen LetsGrow. Aangezien de beperkte tijd is het niet mogelijk om beide kwaliteitsattributen te testen, dus is er gekozen om voornamelijk de functionaliteit te testen. De belangrijkste reden hiervoor is dat ik voornamelijk functionaliteit ga ontwikkelen om het dashboard in te richten en hierdoor minder aan de usability ga werken. Dit is uiteraard wel een onderdeel van de opdracht en hieronder vallen onder andere de configuratie schermen voor de dashboard tegels. Echter is dit een kleiner deel van de opdracht. Buiten deze redenen mag er vanuit gegaan worden dat het gekozen dashboard pakket reeds goed getest is op functionality en usability.

Gebruikte testmethodes

De eerste testsoort die gebruikt zal worden is de moduletest. Het dashboard pakket wordt uitgebreid met meerdere tegels die gezien kunnen worden als verschillende modules. Dit zal dus een erg goede methode zijn om deze functionaliteiten te testen. De testen die door mij worden uitgevoerd (met uitzondering van de systeemtest) worden uitgevoerd met behulp van de ontwikkeltools tegen de testomgeving aan. De systeemtest zal op de testomgeving worden uitgevoerd. Joost, de tester van LetsGrow, zal de testen ook op de testomgeving uitvoeren.

Verder zal er in dit project gewerkt worden met erg veel verschillende applicaties die in het dashboard bij elkaar komen, dus de ketentest is hiervoor een goede testmethode. Met een ketentest wordt gecontroleerd of de processen en systemen op de juiste manier zijn geïntegreerd en samen een werkend geheel vormen. De systemen zijn bijvoorbeeld de autorisatie applicatie, grafiek applicatie, meter applicatie en dergelijke bedoeld. Kortom, alle verschillende applicaties van LetsGrow waar het dashboard mee zal communiceren.

Ook zal er een systeemtest worden uitgevoerd. Dit systeemtest zal aan het eind van het proces worden opgesteld, aangezien pas op dat moment bekend is welke functionaliteit de applicatie bevat en wat er dus getest zal moeten worden. De tester van LetsGrow zal de losse onderdelen van de applicatie testen. De systeemtest zal door mijzelf worden opgesteld en uitgevoerd om te bepalen of het dashboard voldoet aan de requirements.

Niet gebruikte testmethodes
De testmethodes die ik niet ga uitvoeren zijn de acceptatietesten. Deze zullen namelijk worden uitgevoerd door Joost, de tester van LetsGrow.

Verdere uitleg over de gekozen testmethodes is te vinden in hoofdstuk 2.3.2 van het mastertestplan.

Strategie matrix
In de strategie matrix is bepaald op welke diepte er per testmethodiek getest zal kunnen worden. Echter zal zoals eerder beschreven niet iedere test worden uitgevoerd in verband met het gebrek aan tijd.

Op basis van deze strategie matrix worden de strategie matrixen in het detailtestplan opgezet. Deze strategie matrix is te vinden in hoofdstuk 3.3.3 van het mastertestplan.

Testomgeving
De ontwikkelstraat van LetsGrow bestaat, zoals beschreven in hoofdstuk 3.1, uit 3 omgevingen. De ontwikkelomgeving draait lokaal op de pc van de ontwikkelaar. Indien de functionaliteit af is wordt deze op de TFS als shelveset aangeboden voor een code review. Zodra de codereview is goedgekeurd zal de code worden ingecheckt en op de testomgeving worden geplaatst. Meer informatie over de codereviews zijn te vinden in hoofdstuk 5.2 en 5.3.

Vrijgave advies
Na afloop van het project zal er een vrijgave advies worden uitgegeven. Dit rapport beschrijft de status waarin de applicatie zich na afloop bevindt en geeft advies of de applicatie geheel klaar is voor een laatste gebruikersacceptatietest door Joost, de tester van LetsGrow, of dat nog niet alle overeengekomen functionaliteit is gerealiseerd.

Een vrijgave advies kan resulteren in een van de volgende adviezen: “Vrijgeven”, “Conditioneel” en “Niet vrijgeven”. Om het vrijgave advies te kunnen geven zal er ook gekeken moeten worden naar een aantal acceptatiecriteria.

Acceptatiecriteria
De acceptatiecriteria voor dit project zijn als volgt:

- Het eindproduct moet voldoen aan alle functionele requirements die geprioriteerd zijn op “Must have” en “Should have”
- Alle testen moeten geslaagd zijn. Testen die niet slagen door een fout in het dashboard pakket, maar waar wel een work-around voor beschikbaar is, kunnen in overleg worden toegestaan.
- De applicatie moet functioneel zijn op zowel een desktop omgeving als op een mobiel apparaat.

Indien blijkt dat het systeem nog niet voldoet aan de acceptatiecriteria, dan zal deze nog niet in zijn geheel worden opgeleverd voor de gebruikersacceptatietest.

Aangezien de applicatie wordt ontwikkeld in iteraties zal er steeds een werkend product worden opgeleverd. Dit werkende product zal eerder al opgeleverd worden voor een selecte groep klanten. Voor deze releases zal er geen vrijgave advies worden opgesteld, omdat het nog niet om een publieke release gaat. Het vrijgave advies zal aan het eind van het traject worden opgesteld en zal uiteindelijk het advies geven of de applicatie klaar is om publiekelijk uit te brengen.

5.5 Planning

Zoals in de planning hieronder is af te lezen, zullen eerst de testen ontworpen worden en vervolgens na het ontwikkelen pas uitgevoerd worden. De acceptatietesten zijn niet in de planning opgenomen, aangezien deze niet door mijzelf worden uitgevoerd maar door de tester van LetsGrow.

Het ontwerpen van de module- en ketentest zal aan het begin van de sprint worden opgezet. Na de ontwikkeling zullen de module testen worden uitgevoerd. De systeemtest zal aan het eind van het ontwikkelproces worden opgezet en uitgevoerd, aangezien pas op dat moment de te testen functionaliteit bekend is. Hierna zullen de ketentesten uitgevoerd worden.

Detailtestplan
Het detailtestplan is opgezet op basis van het mastertestplan. Er worden vervolgens testontwerpen opgezet voor de verschillende testsoorten. Ook zal er een systeemtest worden uitgevoerd. De keuzes van de toegepaste testsoorten zullen onderbouwd worden in de iteraties bij het kopje “Testen”.

Testbasis
Als testbasis wordt er gebruik gemaakt van verschillende documenten, namelijk:

- Functioneel ontwerp
- Technisch ontwerp

Verder zullen ook de rest van de documenten, waar de eisen uit af te leiden zijn, worden opgenomen als testbasis.

Teststrategie
Uit de kwaliteitsattributen tabel zijn de kwaliteitsattributen beoordeeld op hun relatief belang en hier zijn een aantal kwaliteitsattributen uit voortgekomen met het relatief belang hoog. Deze belangen zijn onderdeel van het kwaliteitsattribuut “functionaliteit”. De overige belangen met een hoog relatief belang kwamen voornamelijk uit het kwaliteitsattribuut “bruikbaarheid”, maar deze worden zoals beschreven in hoofdstuk 5 van het detailtestplan buiten beschouwing gelaten. De volledige kwaliteitsattributen tabel wordt beschreven in hoofdstuk 3.3.1 van het mastertestplan.

Strategie matrix
In deze strategiematrix is de testdiepte bepaald voor de belangrijkste kwaliteitsattributen. Deze strategiematrix is opgezet met behulp van de testboom en de kwaliteitsattributen tabel. Aan de hand van deze matrix wordt bepaald met welke testdiepte de onderdelen getest moeten worden. De totstandkoming van deze matrix is te vinden in hoofdstuk 5.1 van het detailtestplan. Het detailtestplan is bijgevoegd als los document. Deze tabel is door Joost, Edwin en mijzelf ingevuld en dit heeft vervolgens tot de onderstaande strategie matrix geleid.

Risicogebied	Testrisicoanalyse	Suitability (Geschiktheid)	Interoperability (interoperabiliteit)	Security (Beveiliging)
		H	H	H
Grafiek tonen	K	K	K	L
Dashboard autoriseren	K	-	K	K
Meter tonen	H	K	K	K
Rapport tonen	H	K	K	L
Dashboard delen	M	H	H	K
Tegel autoriseren	M	L	L	H
Terugkeren naar klantmenu	M	-	M	-
Afbeelding tonen	L	M	H	H
Weer tonen	L	L	M	-
Veiling informatie tonen	L	L	M	M
Tegel details bekijken	L	-	L	L
Twitter tonen	L	L	M	-
Nieuws tonen	L	L	M	-

Tabel 11 : Strategiematrix
L = Laag, M = Middel, H = Hoog, K = Kritisch

Strategie matrix – testsoorten

In dit hoofdstuk is opnieuw de strategie matrix opgezet, maar dit maal wordt er ook weergeven welk testsoort er gebruikt zal gaan worden om het risicogebied te testen. Doordat er voor het dashboard gebruik gemaakt zal gaan worden van de LetsGrow autorisatie is het niet nodig om de autorisatie nog expliciet te testen. Tijdens het ontwikkelen zal namelijk informeel getest worden op ongeldige data.

Risicogebied	Suitability (Geschiktheid)	Interoperability (interoperabiliteit)	Security (Beveiliging)
Grafiek tonen	AT (testmaat 2)	PCT	
Dashboard autoriseren			
Meter tonen	AT	PCT	
Rapport tonen		PCT	
Dashboard delen	EG		
Tegel autoriseren			
Terugkeren naar klantmenu			
Afbeelding tonen			
Weer tonen			
Veiling informatie tonen			
Tegel details bekijken	EG		
Twitter tonen			
Nieuws tonen			

Tabel 12 : Strategie matrix – testsoorten
ST = System Test, AT = Algoritm Test, EG = Error Guessing, PCT = Process Cyclus Test

De risicogebieden zonder gespecificeerde testmethode zullen informeel worden getest.

5.6 Problemen

De problemen die in deze iteratie optraden zaten deels in de functionaliteit van het gekozen software pakket en deels in het testgedeelte. Tijdens het implementeren bleek dat het dashboard niet volledig naar behoren functioneerde, terwijl dit in de demo op de site van het dashboard pakket wel goed functioneerde. Deze problemen traden op toen het dashboard was geïmplementeerd en ik de functionaliteiten informeel heb getest om te controleren of deze nog allemaal werkte. Zo werden bijvoorbeeld de dashboard tegels niet meer getoond nadat de pagina werd ververst, of het keuzescherm om een ander dashboard te selecteren functioneerde niet meer. Na wat mailwisseling met de ontwikkelaars van het dashboard pakket werd het probleem gelokaliseerd en opgelost.

Verder bleek er tijdens het implementeren van de autorisatie (zie hoofdstuk 5.3) dat dit niet volledig naar behoren functioneerde en mede hierdoor is er in overeenstemming met LetsGrow besloten om de autorisatie via de autorisatie service van LetsGrow te laten verlopen. Dit zorgt dat alles via een centraal punt te beheren zal zijn en met het oog op de toekomst zal dit het delen van dashboards vereenvoudigen.

Verder was er wat twijfel wat betreft het testen, maar na contact opgenomen te hebben met mevrouw Lousberg heb ik wat waardevolle tips ontvangen en kon ik verder.

Verder heeft tijdens deze sprint een computer fout voor wat problemen gezorgd. Na een update van Windows wilde het systeem niet meer opstarten. Aangezien de documentatie werd opgeslagen in de Cloud en het project op de Team Foundation Server werd opgeslagen, was er geen data verlies. Helaas kostte dit wel wat tijd, omdat geregeld moest worden dat mijn systeem opnieuw geïnstalleerd werd en nadat deze opnieuw geïnstalleerd was moest het systeem weer opnieuw worden configureert.

5.7 Mijlpalen en evaluatie

Deze iteratie was gebruikt voor het implementeren van het dashboard pakket. Dit bestond uit verschillende onderdelen, zoals het inrichten van de TFS, implementeren van de database, implementeren van de autorisatie en het inrichten van de testomgeving.

Er zijn deze iteratie verschillende belangrijke keuzes gemaakt, namelijk de keuze om de LetsGrow autorisatie in te bouwen en de keuze voor de te gebruiken testmethodieken.

Tijdens de afspraak met mevrouw Lousberg heb ik wat waardevolle feedback ontvangen op mijn reeds gemaakte documenten. Naar aanleiding van dit gesprek is ook mijn planning verbeterd, aangezien deze te globaal beschreven was.

6. Iteratie 3: Grafieken en rapporten

Deze iteratie staat in het teken van het tonen van de grafieken en rapporten op het dashboard. Hiervoor is het technisch ontwerp aangepast en zijn er een aantal testen opgesteld. De sprintbacklog van deze iteratie bestaat uit: Selecteren grafiek uit teeltgroep, selecteren rapport uit teeltgroep, tonen grafiek op dashboard, tonen rapport op dashboard. Dit omvat requirements: 1.1.1 en 1.1.2

6.1 Ontwerpen

In eerste instantie was het de bedoeling dat de grafieken, rapporten en meters op het dashboard als afbeelding getoond zouden gaan worden. Echter nadat ik de demo had gegeven (zie hoofdstuk 4.4), werd de vraag gesteld of de grafieken nu ook interactief waren, zodat ze op dezelfde manier gebruikt kunnen worden als de huidige situatie. Dit is echter in het geval van afbeeldingen niet mogelijk, dus is de keuze gemaakt om de grafiek niet als afbeelding te tonen. In plaats daarvan is de keuze gemaakt om de grafiek applicatie te tonen in een iframe binnen een dashboard tegel. Om dit goed te laten functioneren zullen er ook een aantal aanpassingen moeten worden doorgevoerd in de grafiek en rapport applicatie. In bijlage H: “Wijzigingen benodigd voor grafiek en rapport applicatie” is beschreven welke wijzigingen aan de grafiek en rapport applicatie nodig zijn om volledig te kunnen laten functioneren op het dashboard.

Er bestaan drie verschillende soorten categorieën waaruit een grafiek of rapport gekozen kan worden. In overleg met Leon is gekozen om te beginnen met het ophalen van grafieken en rapporten uit de teeltgroepen, aangezien dit voldoende complexiteit bevat en deze teeltgroepen voldoende grafieken en rapporten bevatten om de door mij ontwikkelde functionaliteit te kunnen testen. Zo zal bijvoorbeeld het ophalen van grafieken en rapporten uit modules een stuk lastiger worden, aangezien hier een extra abstractie laag tussen zit en dit meer kennis van het systeem vereist. Het ophalen van overige definities zal daarin tegen wat eenvoudiger zijn, aangezien dit één op één verbindingen zijn in de database.

Functioneel ontwerp

Het functioneel ontwerp is voor deze iteratie niet aangepast, aangezien deze functionaliteit reeds als requirement is opgenomen en dit voldoende beschreven is.

Na deze iteratie is de gebruiker in staat om een grafiek of rapport tegel naar het dashboard te slepen, om vervolgens met behulp van het te ontwikkelen selectie proces een bestaande grafiek of rapport te kunnen selecteren.

Technisch ontwerp

Het technisch ontwerp is hier wel voor aangepast, mede doordat er besloten is om de grafieken niet meer weer te geven als afbeelding, maar om de grafiekapplicatie binnen een iframe te tonen.

Om de keuzenlijsten te vullen, die gebruikt worden om een categorie en grafiek of rapport te selecteren, is gekozen om een nieuwe model klasse aan te maken met de twee benodigde attributen. In eerste instantie maak ik gebruik van een anonymous type om deze keuzelijsten te vullen. Echter is dit voor de onderhoudbaarheid en het debuggen niet erg geschikt, dus heb ik hier een nieuw object (ObjectTypeModel) voor gemaakt die de twee benodigde velden bevat.

De controllers van de grafieken en rapporten bevatten beide voor een deel dezelfde methodes. De belangrijkste reden hiervoor is dat het dashboard pakket deze methodes binnen een controller verwacht

en zonder deze methodes niet kan functioneren. Voor de gedeelde methodes zijn de abstracte klassen aangemaakt. Het klassendiagram is groen om aan te geven dat dit volledig door mij is gemaakt.

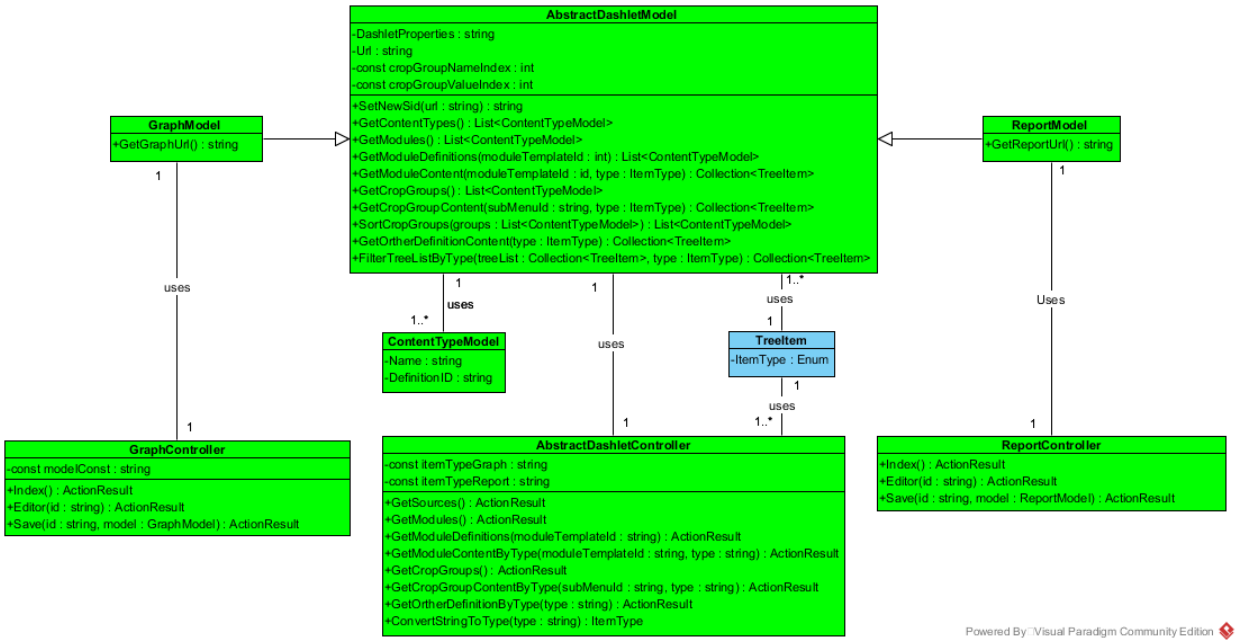


Diagram 7 : Grafiek en rapport controllers en modellen

Voor de “id” parameters wordt er gebruik gemaakt van een string in plaats van een integer. Dit is overgenomen uit het huidige systeem, aangezien deze een id als string verwachten. De reden hiervoor is dat het als string uit de database komt en verder weinig noodzaak was om het te parsen naar een integer. Mocht dit toch nodig zijn kon dit altijd nog.

6.2 Ontwikkelen

Buiten de logica in de controller en model klassen wordt er binnen de views veel gebruik gemaakt van javascript, Ajax en jQuery. Het selectieproces van een grafiek of rapport bestaat voor de gebruiker uit meerdere schermen, maar op de achtergrond is dit een pagina waar verschillende elementen zijn verborgen. Zodra de gebruiker op ‘Volgende’ klikt, wordt het huidige element verborgen en wordt het andere element getoond. Zo kan de gebruiker middels een soort wizard eenvoudig een grafiek of rapport tegel configureren.

Verder wordt er voor het vullen van de keuzelijsten, waarmee de gebruiker de te tonen grafiek of rapport kiest, gebruik gemaakt van Ajax calls. Ik heb de Ajax calls op zo’n manier opgebouwd dat dit zelfde bestand gebruikt kan worden voor het ophalen van zowel grafieken, rapporten als de nog te ontwikkelen meters. Om, op het moment van selecteren van bijvoorbeeld een module, een lijst met bijbehorende elementen op te halen, wordt er gebruik gemaakt van jQuery en Ajax. Hierbij wordt de gekozen optie als attribuut meegeven aan de Ajax call en deze zorgt op zijn beurt dat de volgende keuzelijst wordt gevuld met de juiste velden. Door deze request met Ajax af te handelen hoeft de pagina niet ververst te worden en krijgt de gebruiker niet steeds een knipperend beeld.

Afbeelding 12 toont het view bestand van de grafiek tegel. De verborgen span elementen worden gebruikt om de dynamische Ajax call de juiste request te laten maken. Het is nodig om deze Actions in

de specifieke index pagina te definiëren, zodat er niet onnodig html elementen worden gegenereerd en op deze manier de uitbereiding van een nieuw type weergave vrij eenvoudig te realiseren is.

```
<div>
    @Html.Partial("SharedEditPanel")

    <span id="getmodulecontent" class="hidden">@Url.Action("GetModuleContentByType", "AbstractDashlet", new { type = "graph" })</span>
    <span id="getcropgroupcontent" class="hidden">@Url.Action("GetCropGroupContentByType", "AbstractDashlet", new { type = "graph" })</span>
    <span id="getortherdefinitions" class="hidden">@Url.Action("GetOrtherDefinitionContentByType", "AbstractDashlet", new { type = "graph" })</span>
</div>
```

Afbeelding 12: Selectie view grafiek tegel

Afbeelding 13 geeft de dynamische Ajax request weer die gebruik maakt van de span elementen uit afbeelding 12. Met behulp van de “fillOptions” methode wordt de nieuwe selectie lijst gevuld.

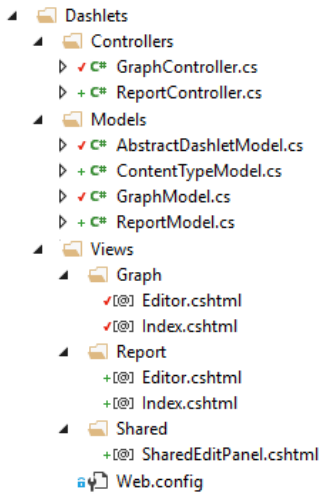
```
function fillOptions(data, optionId, additionalParam) {
    var json = JSON.parse(data);
    var options = $("#selectbox_" + optionId);
    options.empty();
    options.append($("<option />").text($("#firstinputitemtext").text()).val("0"));
    $.each(json, function () {
        options.append($("<option " + additionalParam + "/>").val(this.DefinitionID).text(this.Name));
    });
}

function getOptions(source) {

    if (source != "0") {
        $.ajax({
            type: "GET",
            url: $("#get" + source + "s").text(),
            dataType: "text",
            success: function (data) {

                fillOptions(data, source);
            },
            error: function () { }
        });
    }
}
```

Afbeelding 13: Dynamische Ajax call



Afbeelding 14: Mappenstructuur

Het leek in eerste instantie niet mogelijk om een javascript bestand te delen binnen dit dashboard pakket, door een voor mij onbekende reden, maar dit bleek om een bug te gaan binnen de html view bestanden. Meer hierover in hoofdstuk 6.4.

Iedere dashboard tegel heeft in dit dashboard pakket zijn eigen view, dus zo ook de grafiek en rapport tegel. Aangezien het selectieproces van grafieken en rapporten nagenoeg gelijk is, zal de selectiepagina

ook grotendeels hetzelfde zijn. Hierdoor is gekozen voor een gedeelde view, waar de grafiek view en rapport view beide gebruik van maken.

Ten slotte kreeg ik als feedback uit de codereview dat er wat vaste teksten binnen de code en views stonden en dat deze het beste zo snel mogelijk in een “Resource bestand” geplaatst moesten worden, zodat de applicatie direct met verschillende talen overweg kan. Ook dit is verbeterd.

Doordat de grafieken en rapporten binnen een iframe op het dashboard getoond gaan worden, heb ik wijzigingen moeten maken aan de grafiek applicatie. Deze was namelijk niet responsive genoeg. Hierdoor is met behulp van bootstrap het menu van de grafiek aangepast en is de legenda (die door het schalen onleesbaar werd) gewijzigd. De legenda is nu minder gedetailleerd, zodat deze weer leesbaar wordt. Een bijkomend voordeel is dat de grafiek applicatie, buiten het dashboard om, nu ook goed bruikbaar is op een mobiel apparaat.

6.3 Testen

De onderdelen die gemaakt zijn in deze iteratie zijn, zijn ook getest. De volgende functionaliteiten zijn getest: Grafiek tonen, grafiek selecteren, rapport tonen, rapport selecteren.

Grafiek tonen: Algoritmetest (testmaat2)
Hier wordt een voorbeeld gegeven van een kleine test, namelijk het tonen van de grafiek. Het bijbehorende diagram is te vinden in hoofdstuk 6.1 van het detailtestplan. Er is gekozen voor een algoritmetest, zodat de functionaliteit op alle verschillende manieren wordt doorlopen. Hiermee kan bepaald worden of deze verschillende manieren de verwachtte uitkomst opleveren.

In en uitgaande paden

Beslispunt	Paden in	Paden uit
A	1	2, 4
B	3	3, 5, 6

Tabel 13: In en uitgaande paden – Grafiek tonen: Algoritmetest(testmaat2)

Paden combinaties

Beslispunt	Combinaties
A	1-2, 1-4
B	2-3, 2-5, 2-6

Tabel 14: Paden combinaties – Grafiek tonen: Algoritmetest(testmaat2)

Logische testgevallen

Logisch testgeval	Beschrijving	Pad
At.l.grafiek.tonen.1	Succesvol grafiek tonen.	1-2-3
At.l.grafiek.tonen.2	Het dashboard bevat geen grafieken.	1-4
At.l.grafiek.tonen.3	De grafiek tegel op het dashboard is niet geconfigureerd.	1-2-5
At.l.grafiek.tonen.4	De grafiekapplicatie is offline	1-2-6

Tabel 15: logische testgevallen – Grafiek tonen: Algoritmetest(testmaat2)

Fysieke testgevallen

Fysiek testgeval	Logisch testgeval	Beschrijving	Handelingen	Verwachte uitkomst
At.f. grafiek.	At.l. grafiek.	Succesvol grafiek tonen.	Eerste poging:	Eerste poging:

tonen.1	tonen.1		- Gebruiker opent het dashboard.	- De grafiek tegel wordt getoond.
At.f. grafiek. tonen.2	At.l. grafiek. tonen.2	Het dashboard bevat geen grafieken.	Eerste poging: - Gebruiker opent het dashboard.	Eerste poging: - Er wordt geen grafiek tegel getoond.
At.f. grafiek. tonen.3	At.l .grafiek. tonen.3	De grafiek tegel op het dashboard is niet geconfigureerd.	Eerste poging: - Gebruiker opent het dashboard.	Eerste poging: - Er wordt een lege grafiek tegel getoond.
At.f. grafiek.t onen.4	At.l. grafiek. tonen.4	De grafiekapplicatie is offline	Eerste poging: - Gebruiker opent het dashboard.	Eerste poging: - Er wordt een grafiek tegel getoond met een foutmelding.

Tabel 16: Fysieke testgevallen – Grafiek tonen: Algoritmetest(testmaat2)

6.3 Problemen

Het ontwikkelen van de grafieken en rapporten op het dashboard leverde een klein probleem op. Dit probleem kwam door een bug in het dashboard pakket. Het bleek niet mogelijk te zijn om op de traditionele manier een javascript bestand toe te voegen aan een html pagina. Na contact opgenomen te hebben met de ontwikkelaars van het dashboard pakket werd mij verteld dat dit inderdaad om een bug in het pakket ging en werd ik geïnformeerd over een work-around. De work-around voor deze bug was om met behulp van javascript een nieuwe script tag aan te maken, hier de script source aan mee te geven en vervolgens deze tag aan de body van het html element toe te voegen.

Verder zijn drie van de 9 testen niet geslaagd. Dit zijn de testen waar gebruik gemaakt wordt van het automatisch openen van het configuratie scherm van de dashboard tegels. Doordat er een work-around beschikbaar is en deze functionaliteit onderdeel is van het dashboard pakket, wordt hier nu even niks mee gedaan en is de bug gemeld bij de ontwikkelaars.

6.4 Complexiteit

De complexiteit van deze iteratie zit hem niet zo zeer in het tonen van de tegels op het dashboard, maar vooral in de communicatie tussen vele externe systemen en verschillende programmeertalen. Voornamelijk het selectieproces kostte veel tijd. Hier zat namelijk veel uitzoekwerk in, gezien de grafieken e.d. opgehaald moeten worden vanuit verschillende applicaties. Vervolgens moet deze informatie op een zo gebruiksvriendelijke manier aan de gebruikers worden getoond, zodat iedereen eenvoudig de juiste grafiek of het juiste rapport kan selecteren.

Ook was het belangrijk om zo veel mogelijk redundantie te voorkomen, aangezien het selectieproces van grafieken, rapporten en de nog te ontwikkelen meters grotendeels overeen komt en het niet de bedoeling is dat hier drie maal exact dezelfde code voor gebruikt wordt.

Verder zijn er aan de bestaande grafiekapplicatie wijzigingen aangebracht om deze beter op mobiele apparaten te laten werken. Dit is gerealiseerd met behulp van bootstrap en aangezien dit voor mij nieuw was heb ik hier wel wat tijd in moeten steken. Aangezien iedere actie binnen LetsGrow geautoriseerd is, is het belangrijk dat bij iedere verzoek het sessie id wordt meegegeven. Aangezien dit id verloopt na 40 minuten zal bij het tonen van de grafiek iedere keer het sessie id opgevraagd moeten

worden en zal de URL opnieuw moeten worden opgebouwd. Om dit te realiseren is een handige constructie bedacht en geïmplementeerd.

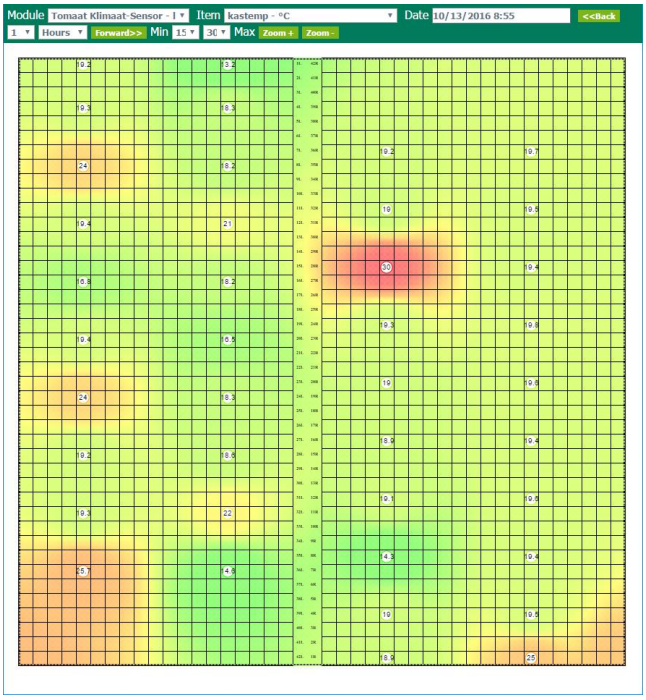
6.5 Mijlpalen en evaluatie

Er zijn, buiten het probleem beschreven in hoofdstuk 6.4, weinig problemen opgetreden in deze iteratie. Echter was het werken met bootstrap voor mij wel nagenoeg nieuw, dus ik heb dit wel even moeten uitzoeken. Aangezien het in eerste instantie de bedoeling was om de grafieken en dergelijke als afbeelding te tonen had ik niet gerekend op het extra werk om de grafiek applicatie om te bouwen met bootstrap. LetsGrow en ik zijn toch erg positief over de keuze van deze verandering, aangezien het de lay-out een goede draai heeft gegeven en de grafiek nu ook, buiten het dashboard om, goed te bekijken is op een mobiel apparaat.

7. Iteratie 4: Meters en plattegrond

Deze iteratie staat in het teken van het kunnen aanmaken en op het dashboard kunnen tonen van meters. Deze functionaliteit is nog niet beschikbaar binnen LetsGrow, dus hier moet ik een nieuwe applicatie voor ontwikkelen. Dit omvat requirement 1.1.4 en 1.17

Verder is er deze iteratie een nieuwe requirement bijgekomen, namelijk het toevoegen van een plattegrond op het dashboard. Een ontwikkelaar van LetsGrow is, op moment van schrijven, bezig met het ontwikkelen van deze functionaliteit. Op deze plattegrond kan de klant in een oogopslag de temperatuur van verschillende sensoren in de kas bekijken en direct zien zodra er iets fout gaat. De manager van LetsGrow had het idee van de plattegrond besproken met een klant en aangezien deze erg enthousiast was moest deze functionaliteit, met voorrang op de andere requirements, bij het dashboard worden ingebouwd. Hierdoor is deze nieuwe functionaliteit direct meegenomen in deze sprint en zijn een aantal andere requirements, zoals het kunnen tonen van afbeeldingen, beoordeeld met een lagere prioriteit. Deze requirement is tot stand gekomen door een samenwerking tussen het sensor bedrijf “Technolution” en LetsGrow. De hittepunten representeren de sensoren van Technolution.



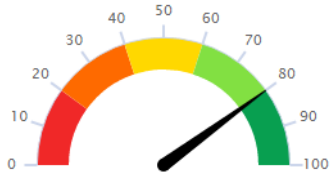
Afbeelding 15: Plattegrond LetsGrow

De sprintbacklog bestaat deze iteratie uit: Onderzoekje naar bestaande meters, ontwikkelen meterapplicatie binnen LetsGrow, tonen meter op dashboard, tonen plattegrond op dashboard.

7.1 Ontwerpen

Voor er begonnen kon worden met het ontwikkelen van de nieuwe meter applicatie, is er een document opgesteld met verschillende meter mogelijkheden. In dit document is er voor verschillende type meters informatie verzameld, zoals de benodigde data om de meter te kunnen realiseren, waar de meter voor gebruikt kan worden en hoe de meter er uit kan komen te zien. Dit document genaamd: ‘Onderzoekje verschillende meters’ is te vinden in bijlage I.

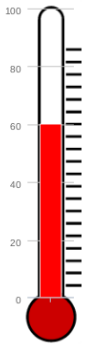
Hieronder worden een paar voorbeelden uit de bijlage getoond. Ook worden hier een aantal toepassingen voor de meter beschreven.



Afbeelding 16: Meter

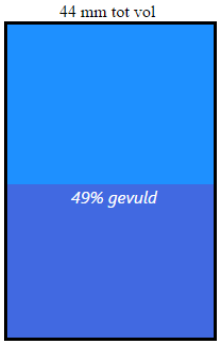
Gebruikt voor:

- Planten opgepot per uur
- Gemiddelde temperatuur over bepaalde periode
- Windsnelheid
- Windrichting
- Stroomproductie



Gebruikt voor:
- Temperatuur buiten/ bepaalde afdeling

Afbeelding 17: Temperatuur meter



Gebruikt voor:
- Delfland project
- Vulling bassin / silo

Afbeelding 18: Niveau meter

Voor een aantal van deze meters wordt gebruik gemaakt van de Highcharts library. Deze library wordt momenteel ook gebruikt voor de grafiek applicatie en hier is dus momenteel al een licentie voor aanwezig. Highcharts is slechts gratis voor non-commercial projecten, dus zodra het gebruikt wordt voor een commercial project zal er voor het gebruik betaald moeten worden.

Het onderzoekje naar verschillende type meters is door de ontwikkelaars, tester en manager van LetsGrow doorgelezen en ze hebben hier vervolgens feedback op gegeven. Deze feedback is opgenomen in bijlage I: “Onderzoekje verschillende meters”.

Een van de ideeën voor een meter was een niveau meting van een waterbassin. Deze meter is voor een nieuw project van LetsGrow dat door Delfland is opgezet. Dit is een project, waar met behulp van een weersverwachting, wordt bepaald of er regen zal vallen. Mocht dit het geval zijn, dan kunnen tuinders hun waterbassin van tevoren voor een deel leeg laten lopen zodat de pompen en gemalen het eerste deel van het water vooraf kunnen wegpompen. Op deze manier kunnen de pompen voor een deel ontlast worden op het moment dat het daadwerkelijk gaat regenen. Dit idee heeft geleid tot de niveau meter. Verder zal de meter applicatie een normale meter bevatten, een temperatuur meter en een statusmeter, waarbij er met behulp van een soort stoplicht wordt weergeven of iets goed gaat of niet. Aangezien de niveaumetingen een nieuw idee zijn, is hier nog geen implementatie voor binnen LetsGrow.

Verder zal in deze iteratie aandacht geschonken worden aan het toevoegen van plattegronden aan het dashboard. Deze functionaliteit is er nog wel tussen te passen, aangezien de selectieprocedure van een plattegrond sterk lijkt op die van een meter en het toevoegen van een plattegrond. Het toevoegen van deze tegel zal naar verwachting relatief weinig tijd in beslag zal nemen.

Functioneel ontwerp

Om de meter applicatie en de plattegrond tegel te kunnen realiseren is het ook nodig om wijzigingen aan het functioneel ontwerp te maken. Het functioneel ontwerp mistte namelijk nog het aanmaken van een nieuwe meter en het toevoegen van een plattegrond. Deze zijn toegevoegd aan het meegeleverde document 'Functioneel ontwerp' onder andere als requirements. Verder zijn er wijzigingen gemaakt aan het use case diagram (zie hoofdstuk 5 van het bijgeleverde document "functioneel ontwerp") en voor het toevoegen van een meter is vervolgens ook een use case tabel opgezet, omdat deze nieuwe functionaliteit duidelijk moet worden beschreven. Het toevoegen van een plattegrond valt onder het toevoegen van een nieuwe tegel en is beschreven in use case diagram 6.1 in het functioneel ontwerp.

Technisch ontwerp

Voor de nieuwe requirement zijn er ook wijzigingen gemaakt aan het technisch ontwerp. Hier is het technisch ontwerp van de meter applicatie en de bijbehorende dashboard tegel aan toegevoegd. Om de meter applicatie te ontwikkelen is er voor gekozen om dit binnen de grafiek applicatie te doen, aangezien het op die manier niet nodig was om een nieuw project te starten en alle referenties die door de grafiek applicatie gebruikt worden opnieuw te definiëren. De meter applicatie zal namelijk ongeveer dezelfde LetsGrow applicaties gebruiken (zoals autorisatie e.d.) als de grafiek applicatie.

In het klassendiagram hieronder wordt de meter applicatie beschreven Dit is een nieuw klassendiagram voor nog niet bestaande functionaliteit.

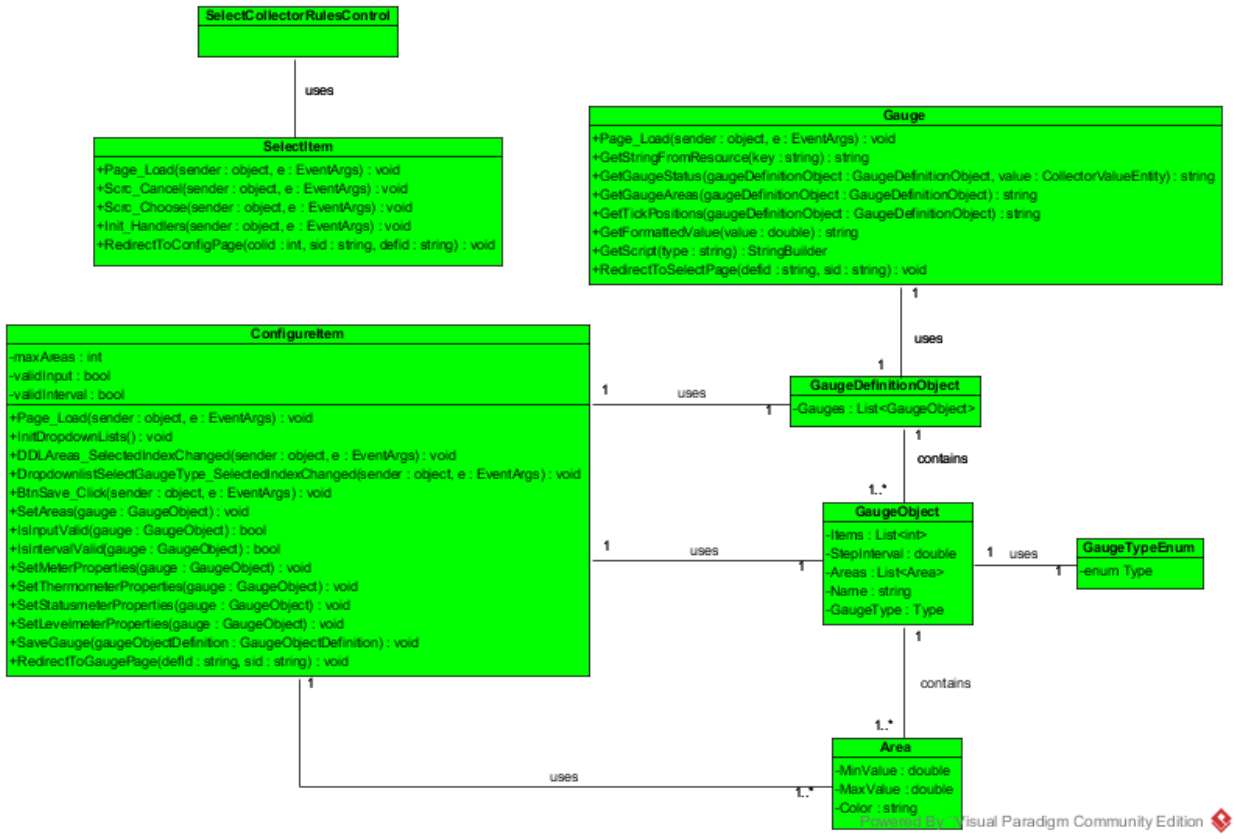


Diagram 8: Klassendiagram meter applicatie

Aangezien de grafiek applicatie is gebouwd in C# Web Forms is er voor gekozen om de meter applicatie ook op dezelfde manier te ontwikkelen. Dit voorkomt dat de grafiek applicatie een mix wordt tussen Web Forms en MVC. Het bovenstaande klassendiagram geeft de 3 schermen weer waar de gebruiker doorheen zal klikken tijdens het configureren van een meter. Het eerste scherm “Gauge” controleert of de meter reeds geconfigureerd is. Als dit het geval is, dan wordt de meter getoond. Is de meter nog niet geconfigureerd zal het “SelectItem” scherm getoond worden. In dit scherm zal de gebruiker een item kiezen waar een meter voor gemaakt moet worden. In het derde scherm “ConfigureItem” heeft de klant de mogelijkheid om het type meter te kiezen en deze vervolgens naar wens te configureren. Door de meter applicatie op te delen in meerdere pagina’s zijn de taken duidelijk gescheiden en dit verbeterd de onderhoudbaarheid.

De keuzes, uitleg en vergroting van dit klassendiagram worden, samen met de overige diagrammen, besproken in hoofdstuk 3.1.3 in het bijgeleverde technisch ontwerp.

7.2 Ontwikkelen

Zoals eerder besproken is de meter applicatie ontwikkeld met Web Forms. De reden hiervoor was dat de huidige grafiek applicatie hier ook mee gebouwd is en de meter applicatie binnen de grafiek applicatie wordt gedefinieerd. De meter applicatie moet gezien worden als een applicatie die los staat van de grafiek applicatie.

Klanten kunnen de nieuwe meter via het oude klant menu aanmaken door uit de selectielijst de keuze ‘meter’ te selecteren en vervolgens op opslaan te klikken. Er wordt nu een leeg definitie object aangemaakt. Door op deze definitie te klikken wordt de meter applicatie geopend en kan de meter geconfigureerd worden. Persoonlijk lijkt het mij mooier als automatisch de meter configuratie pagina wordt geopend zodra er een nieuwe definitie wordt aangemaakt via het oude klantmenu, maar aangezien de huidige manier van werken anders is, zal deze manier worden aangehouden om het voor de klanten zo duidelijk mogelijk te houden.

Om de grafiek aan de gebruiker te tonen wordt er gebruik gemaakt van javascript. De manier om javascript aan een pagina toe te voegen binnen Web Forms is door gebruik te maken van Client scripts. Hieronder wordt een verkorte versie weergeven van de Client script dat gebruikt wordt om de meter te tonen:

```
Type cstype = this.GetType();
ClientScriptManager cs = Page.ClientScript;
string csname1 = "MethodScript";

if (!cs.IsClientScriptBlockRegistered(cstype, csname1))
{
    StringBuilder cstext2 = new StringBuilder();
    cstext2.Append("<script type=\"text/javascript\">");
    cstext2.Append("function getName() { return '" + gaugeDefinitionObject.Gauges[firstGaugeIndex].Name + "' };");
    cstext2.Append("function getValue() { return [" + lastVal + "] };");
    cstext2.Append("</script>");
    cs.RegisterClientScriptBlock(cstype, csname1, cstext2.ToString(), false);
}
```

Afbeelding 19: Voorbeeld Client script

Zoals in afbeelding 19 hierboven wordt weergegeven, wordt er eerst gecontroleerd of het script al is ingeladen. De reden dat deze check wordt gebruikt is omdat Web Forms werkt via postbacks. Op het moment dat er op een knop geklikt wordt, zal de Page_Load methode als eerst worden uitgevoerd waarna pas de button click methode wordt uitgevoerd. In de Page_Load methode kan er met behulp van een IsPostBack attribuut gekeken worden of de pagina initieel geladen wordt of vanuit een PostBack, maar indien het Client script binnen bijvoorbeeld de button click methode is gedefinieerd

bestaat de kans dat deze meerdere keren wordt uitgevoerd. Hiervoor dient de IsClientScriptBlockRegistered methode.

Tot slot is er ook voor gezorgd dat de aangemaakte meter op het dashboard kan worden toegevoegd.

7.3 Feedback klant

Zoals eerder besproken is er deze iteratie speciaal voor een klant de functionaliteit ontwikkeld om een plattegrond toe te kunnen voegen aan het dashboard. Nadat de manager bij de klant was langs geweest en de huidige stand van zaken had laten zien hebben we wat feedback ontvangen wat betreft het dashboard en de plattegrond. Zo waren er een aantal opmerkingen wat betreft de groottes van de dashboard tegels en de indeling hiervan. Verder kwam er uit dit gesprek dat er toch onnodig veel ruimte wordt ingenomen door verschillende elementen en dit vereist dus nog wat aandacht om dit beter te kunnen weergeven.

Aan de hand van deze feedback hebben we een stand-up meeting gehouden om even met elkaar door te spreken hoe deze punten kunnen worden opgelost. Ik vond het erg nuttig dat het dashboard door een klant is beoordeeld, aangezien de klant een goed beeld geeft van hoe de eindgebruiker tegen het dashboard aan zal kijken. Deze feedback is genoteerd om hier de volgende iteratie aandacht aan te kunnen besteden. Met deze informatie zal een document worden opgesteld met wijzigingen die het dashboard pakket zal moeten ondergaan, bijvoorbeeld de huisstijl van LetsGrow implementeren, om beter te integreren binnen de LetsGrow applicaties.

7.4 Demo

In principe wordt iedere eerste woensdag van de maand een demo gegeven door Leon. Hierbij worden de nieuwste functies en veranderingen van de afgelopen maand aan het team van LetsGrow gepresenteerd. Tijdens deze demonstratie zijn ook het dashboard en de plattegrond aan de orde gekomen. De reacties hierover waren zeer positief en ook hier waren er wat ideeën over het verbeteren van het dashboard, zoals het beter benutten van de beschikbare ruimte en het toevoegen van een achtergrond afbeelding op het dashboard.

7.5 Testen

Aangezien er deze iteratie twee nieuwe requirements bij gekomen zijn, zal het testplan worden uitgebreid. Er zal namelijk ook getest worden op het aanmaken van een nieuwe meter, aangezien dit belangrijke nieuwe functionaliteiten heeft gebracht en dit dus ook getest zal moeten worden. Verder heb ik ervoor gekozen om het dashboard ook te testen middels een systeemtest, zodat het hele proces wordt doorlopen en bepaald kan worden of alles aan de eisen voldoet.

Het toevoegen van een plattegrond aan het dashboard heb ik slechts informeel getest, aangezien de selectieprocedure van een plattegrond sterk lijkt op die van een meter en dus is het testen van een meter voor nu voldoende zekerheid geeft voor wat betreft de werking van de plattegrond.

Ik heb ervoor gekozen om het toevoegen van een meter te testen volgens de algoritmetest testmaat 1, aangezien deze procedure redelijk overeen komt met het toevoegen van de grafieken en rapporten aan het dashboard. Aangezien de manier van testen nagenoeg hetzelfde is als dat van de vorige iteratie, zijn deze testen enkel opgenomen in het testrapportage en niet in het afstudeerverslag. Deze testen zijn te vinden vanaf hoofdstuk 3 in het meegeleverde testontwerp document.

Verder is het aanmaken van een meter ook getest met grenswaardeanalyse. Een meter bestaat uit verschillende gebieden met ieder hun eigen grenswaarde, dus een grenswaardeanalyse kan hier goed gebruikt worden. Deze grenswaardeanalyse is te vinden in hoofdstuk 3.3 van het bijgeleverde testontwerpen document.

7.6 Problemen

Deze iteratie is naar mijn mening vrij vlot verlopen en ik ben dan ook niet veel echte problemen tegengekomen. Het grootste probleem had eigenlijk wel te maken met de plattegrond. De ontwikkelaar die bezig is geweest met de plattegrond had zich nog niet veel bezig gehouden met het schalen van de plattegrond, aangezien dit nog wat problemen opleverde. De plattegrond moest toch op het dashboard om het aan de klant te kunnen demonsteren, dus is voor een tijdelijke oplossing gekozen om de plattegrond met behulp van CSS uit te zoomen. Dit is geen nette oplossing, maar voor de presentatie aan de klant was dit tijdelijk voldoende. Doordat de plattegrond uitgezoomd wordt binnen de tegel, worden ook de configuratie opties zoals de datum en tijd selectie en dergelijke verkleind, waardoor de plattegrond niet meer te configureren is via het dashboard.

7.7 Complexiteit

De complexiteit van deze iteratie lag vooral bij het bedenken en selecteren van verschillende weergavemogelijkheden en vervolgens het tonen hiervan. Hierbij is er nagedacht over de herbruikbaarheid van diverse methodes en de uitbreidbaarheid, zodat in de toekomst eenvoudig nieuwe meters kunnen worden toegevoegd aan de door mij ontwikkelde meter applicatie.

7.8 Mijlpalen en evaluatie

Deze iteratie is een klein onderzoekje gedaan naar de verschillende beschikbare meters (zie bijlage I) en is een document opgesteld dat door verschillende werknemers binnen het bedrijf is bekeken. Hierop heb ik feedback gehad en kon aan de hand van deze feedback een keuze maken welke meters er in de eerste versie opgeleverd zouden worden. Bij het ontwikkelen van de meters is rekening gehouden met eventuele uitbreidbaarheid, zodat zonder al te veel moeite nieuwe weergaves in de vorm van een meter toegevoegd kunnen worden aan het dashboard. Ook is er, zoals eerder besproken, deze iteratie een nieuwe requirement bijgekomen, namelijk het tonen van een plattegrond. Hiervoor was het nodig om een nieuwe dashboard tegel te ontwikkelen, waar de klant de mogelijkheid heeft om een plattegrond te selecteren om deze vervolgens te kunnen weergeven op het dashboard.

Het was jammer dat het een beetje tegen zat met de plattegrond, maar door de tijdelijke oplossing heeft dit verder geen grote gevolgen gehad. Dit probleem zal in een volgende iteratie worden meegenomen en opgelost.

Deze iteratie heeft goede eerste versie van de meter applicatie opgeleverd die in de toekomst eenvoudig kan worden uitgebreid.

8. Iteratie 5: Dashboard grafische interface

Deze iteratie ligt de nadruk op de grafische interface van de applicatie. Aangezien Peter, de manager van LetsGrow, het dashboard bij steeds meer klanten wil demonstreren, is het steeds belangrijker om de interface van het dashboard onder handen te nemen. Het is belangrijk om een goede interface te hebben, zodat het dashboard eenvoudig en intuïtief te bedienen is en de klanten een goede indruk kunnen krijgen van het eindresultaat.

Aangezien het dashboard al eerder in gebruik genomen gaat worden bij een selecte groep klanten, is het noodzaak om wat zaken aan het dashboard aan te passen, zodat deze beter overeen komt met de huisstijl. Het is belangrijk dat het dashboard een vertrouwde lay-out heeft, zodat klanten niet verward worden door verschillende interfaces en LetsGrow intuïtief kunnen blijven gebruiken en snel gewend raken aan het dashboard.

Ook is deze iteratie tijd besteed aan het tonen van de plattegrond op het dashboard. Dit leverde in de vorige iteratie een aantal problemen op en is toen met een tijdelijke fix opgelost. Deze iteratie is in overleg met Eray, de ontwikkelaar van het dashboard, dit probleem opgelost. De sprintbacklog van deze iteratie bestaat uit: opstellen lijst benodigde wijzigingen dashboard, aanbrengen benodigde wijzigingen dashboard. Dit omvat requirements: 1.3, 1.5, 1.11, 2.2, 2.5, 2.6, 2.8, 2.9, 2.11, 2.12, 2.14. Requirement 2.14 heeft deze iteratie een hogere prioriteit gekregen. Dit zorgt voor een rustige overgang.

8.1 Ontwerpen

Aangezien het dashboard pakket op een hoop punten nog aanpassingen nodig had, ben ik begonnen met het opstellen van een lijst met benodigde wijzigingen. Deze lijst is opgesteld naar eigen inzicht en ik heb deze vervolgens met Leon en Peter besproken. De lijst, inclusief de besproken feedback van Leon en Peter is te vinden in bijlage J genaamd: “Wijzigingen t.b.v. dashboard stijl”.

De grootste wijzigingen betreffende het ontwerp waren:

- **Optimaal gebruik maken van de beschikbare ruimte (overbodige elementen verwijderen, ruimte tussen tegels minimaliseren)**
  Door optimaal gebruik te maken van de ruimte kan er zo veel mogelijk informatie op het scherm worden getoond.
- **Huisstijl toepassen**
  LetsGrow heeft een eigen huisstijl, waar het dashboard ook aan zal moeten voldoen.
- **Intuïtieve werking**
  Het is erg belangrijk dat de klanten direct begrijpen hoe het dashboard te bedienen is. Zo waren de dashboard tegel bewerk pagina’s wel functioneel, maar nog niet erg intuïtief. Ook hier zijn wijzigingen gemaakt, zodat hier geen fouten gemaakt kunnen worden. Er is tijdens het ontwikkelen in de vorige iteraties geen rekening gehouden met de intuïtieve werking, aangezien de functionaliteit even belangrijker was dan het ontwerp en we hadden al bedacht dat er een iteratie gewijd zou worden aan de lay-out.

Verder is deze iteratie een manier bedacht hoe het dashboard goed te integreren zou zijn, zonder dat klanten van de een op de andere dag een compleet nieuwe interface krijgen. Er is gekozen om de startpagina van het dashboard te gebruiken om het huidige klantmenu te tonen. Hierdoor kunnen de klanten zelf de keuze maken om via het oude klantmenu te blijven werken, of om een dashboard te

openen. Deze oplossing is bedacht samen met Leon en zal gebruikt worden voor een rustige overgang. Mogelijk zal dit wijzigen in de toekomst en zal het dashboard de functionaliteiten van het oude klantmenu volledig overnemen.

8.2 Ontwikkelen

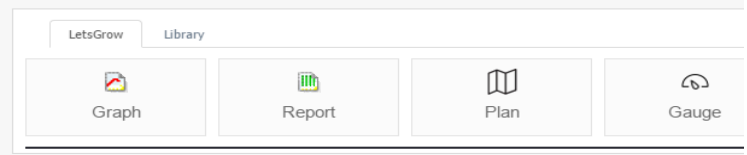
Zoals in de vorige iteratie is besproken was het naar verwachting vrij eenvoudig om de plattegrond tegel te implementeren. Echter had dit nog wel wat voeten in de aarde. Ik heb samen met Eray diverse mogelijkheden bedacht om het dashboard met de plattegrond te kunnen laten communiceren, zodat de hoogte van de plattegrond bepalend was voor de hoogte van de dashboard tegel. Aangezien er de plattegrond, net een aantal van de overige tegels, in een iframe wordt weergeven liepen we al snel tegen het fenomeen “cross site scripting” aan. Dit houdt in dat een pagina een pagina in een ander domein probeert te wijzigen met behulp van javascript en dit is uit veiligheid niet zomaar toegestaan. Dit hebben we kunnen oplossen door een tip van Edwin, door via javascript het domein van de pagina in te stellen. Zodra beide applicaties in hetzelfde domein zitten

Ook is deze iteratie requirement 1.11 ontwikkeld, het tonen van een grafiek in detail. De tegel zal nu ook geopend worden in een nieuw scherm. Hiervoor is met JavaScript en jQuery nieuwe functionaliteit ontwikkelt die deze extra knop aan de dashboard tegel toevoegt. Hierbij was het nodig om een dynamisch gegenereerd element aan te passen. Dit is gerealiseerd doormiddel van onderstaande code:

```
var url = $("#appUrl_" + context["id"]).attr('value');
$(dashletId + " > .d-dashlet-header > .d-dashlet-commands").append("<a target='_blank' href="
+ url + " title='fullscreen' class='dddicon' id='fullscreenanker'></a>");
$(dashletId + " > .d-dashlet-header > .d-dashlet-commands > #fullscreenanker").append("<i
class='d-icon d-icon-fullscreen'>&#xf0db;</i>");
```

Dit was de enige mogelijkheid om een extra optie aan een dashboard tegel toe te voegen, aangezien het dashboard dit officieel niet ondersteunt.

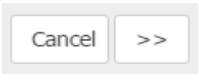
Het dashboard beschikte verder over veel “hardcoded” teksten in de pagina. Deze teksten zouden dus niet vertaald worden op het moment dat de gebruiker van taal wijzigt. Deze teksten zijn allemaal in de resource bestanden geplaatst, om te zorgen dat de klant het dashboard in zijn/haar eigen taal kan bekijken.

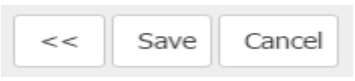


Afbeelding 20: Dashboard tegels

De titels van de dashboard tegels, die bovenin de pagina getoond worden als de gebruiker de bewerk modus heeft ingeschakeld, stonden “hardcoded” in de database. Voor deze titels heb ik een HtmlExtension klasse aangemaakt die een Translate methode bevat. Met behulp van deze methode wordt, via de view, aan de hand van de tegel titel in het resource bestand gezocht naar de vertaling. Er zal dus handmatig een vertaling toegevoegd moeten worden aan het resource bestand zodra er een nieuwe dashboard tegel wordt toegevoegd.

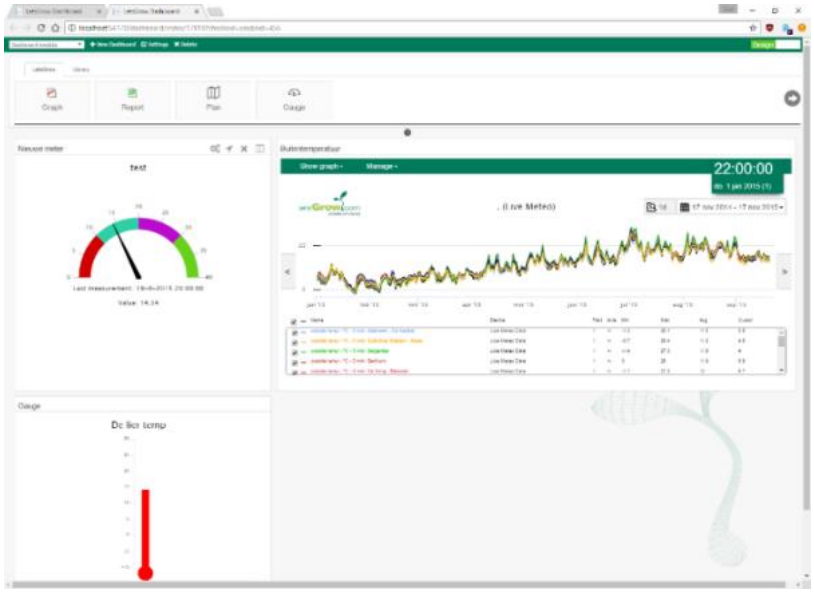
Verder is de navigatie bij de configuratie van de dashboard tegels intuïtiever geworden. Het dashboard pakket genereerd in het configuratie scherm van de tegels automatisch drie knoppen, namelijk Save, Apply en Cancel. Aangezien ik de configuratie procedure heb onderverdeeld in meerdere schermen moest hier ook tussen gewisseld kunnen worden.





Afbeelding 21: Config pagina 1 *Afbeelding 22: Config pagina 2*

Aangezien het dashboard pakket officieel niet de mogelijkheid bied om deze knoppen te wijzigen heb ik dit client side via javascript moeten doen. In plaats van twee losse klikbare teksten “Vorige” en “volgende” om tussen deze pagina’s te wisselen, zijn deze knoppen nu duidelijk geïntegreerd tussen de rest van de knoppen en is de bediening veel eenvoudiger. Voor deze navigatieknoppen is er algemeen javascript bestand. Iedere dashboard tegel maakt hier gebruik van om redundantie te voorkomen.



Afbeelding 23: Nieuwe interface dashboard met LetsGrow huisstijl

8.3 Testen

Deze iteratie is er een grote wijziging gemaakt wat betreft de testmethodiek. Tijdens het 60% concept gesprek met mevrouw Lousberg ben ik tot de conclusie gekomen dat er te diep getest werd. Er werden weinig fouten gevonden en de fouten die werden gevonden hadden ook via een lichtere testmethodiek gevonden kunnen worden. Hierdoor is besloten om de algoritmetesten in deze en volgende iteraties te vervangen door beslissingstabellen testen. Hierdoor blijft er tevens meer tijd over voor het ontwikkelen.

Tijdens het testen is gebleken dat er nog wat problemen waren zodra er meerdere dezelfde tegels aan het dashboard werden toegevoegd. Zodra er bijvoorbeeld 3 plattegrond tegels aan het dashboard werden toegevoegd, werd alleen de eerst toegevoegde tegel in hoogte aangepast en de overige twee niet. Doordat deze drie tegels hetzelfde id hadden werd 3 keer het zelfde element in grootte gewijzigd, maar niet de overige twee. Dit probleem is opgelost door het frame waarin de plattegrond in draait een uniek id te geven, waar het tegel id een onderdeel van is. Eenzelfde probleem trad op bij de nieuwe functionaliteit om de tegel te openen in een nieuw scherm. Ook hier werd slechts aan het eerst element de nieuwe knop toegevoegd, maar dit is met een zelfde manier opgelost.

Doordat het testen te zwaar bleek is het dashboard getest op een lichtere methode. Hiervoor heb ik gekozen voor de beslissingstabellentest. In de onderstaande test wordt het ophalen van grafieken uit modules getest. Dit is een goede test om verbanden weer te geven. Dat kan in deze situatie goed gebruikt worden voor de combinatie van modules en definities. De volledige beslissingstabellentest is te vinden in hoofdstuk 8.1 van het detailtestplan.

Eerst zijn de condities vastgesteld:

Conditie	Naam
C1	Gebruiker beschikt over modules
C2	Module bevat meerdere definities
C3	Module bevat grafieken

Tabel 17: condities beslissingstabellentest

Vervolgens worden de mogelijke acties bepaald:

Actie	Naam
A1	Haal modules op
A2	Haal module definities op
A3	Haal grafieken op

Tabel 18: Acties beslissingstabellentest

Deze acties en condities worden vervolgens opgenomen in een tabel, waarbij alle mogelijkheden worden beschreven. Hier wordt vervolgens bepaald welke mogelijkheden niet kunnen voorkomen en deze worden dan verwijderd uit de tabel:

Condities / acties	1	2	3	4	5
Gebruiker heeft modules	0	1	1	1	1
Module bevat meerdere definities	0	0	0	1	1
Module bevat grafieken	0	0	1	0	1
Haal modules op		X	X	X	X
Haal module definities op			X	X	X
Haal grafieken op			X		X

Tabel 19: Mogelijke condities en acties beslissingstabellentest

Aan de hand van de overgebleven mogelijkheden worden de logische testgevallen opgesteld:

Logisch testgeval	Beschrijving
Lt1.grafiek.ophalen	De klant opent het tegel configuratie scherm en kiest in de bovenste selectie lijst voor modules, maar de gebruiker heeft geen modules.
Lt2.grafiek.ophalen	De klant opent het tegel configuratie scherm en kiest in de bovenste selectie lijst voor modules en de selecteert een van de modules.
Lt3.grafiek.ophalen	De klant selecteert in de selectie lijst onder de gekozen module de enigste definitie.
Lt4.grafiek.ophalen	De klant selecteert in de selectie lijst onder de gekozen module de gewenste definitie.
Lt5.grafiek.ophalen	De klant selecteert in de selectie lijst onder de gekozen module definitie de grafiek

Tabel 20: Logische testgevallen beslissingstabellentest

8.4 Problemen

Deze iteratie ben ik tegen wat kleine beperkingen gelopen van het dashboard pakket, zoals dat het niet mogelijk is om de opties van de dashboard tegel (zoals het vergroten hiervan) toe te voegen. Ook de niet aanpasbare knoppen in het bewerk scherm van een tegel zorgde voor een beperking. Echter was dit met behulp van javascript en jQuery prima op te lossen. Ook de plattegrond en het schalen hiervan zorgde voor de nodige problemen, maar door de goede samenwerking tussen Eray en mij is het uiteindelijk wel gelukt.

8.5 Evaluatie en mijlpalen

Deze iteratie is naar mijn mening prima verlopen. Ik heb een hoop dingen besproken met Leon en Peter over hoe de LetsGrow huisstijl het beste toegepast kan worden en dit vervolgens toegepast. Het dashboard heeft een flinke metamorfose ondergaan en past nu veel beter tussen de overige LetsGrow applicaties. Ik ben erg blij met het resultaat.

9. Iteratie 6: Samenwerking dashboard tegels

Deze iteratie is gebruikt om de samenwerking tussen dashboard tegels te realiseren. Deze wens is naar voren gekomen tijdens een van de demonstraties die Peter heeft gegeven van het dashboard. De klant was erg enthousiast wat betreft het dashboard en bedacht dat het mooi zou zijn als de dashboard tegels onderling met elkaar kunnen communiceren. Dit houdt in dat wanneer de gebruiker bijvoorbeeld zijn muis over een grafiek beweegt, dat de overige tegels hierop reageren en data van dezelfde datum en tijd tonen. Voor de eerste versie zullen alleen de grafiek en de plattegrond met elkaar communiceren.

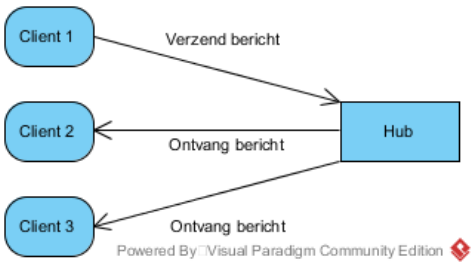
De sprintbacklog van deze iteratie bestaat uit: bepalen benodigde techniek voor tegel communicatie, aanpassingen maken aan grafiek applicatie, aanpassingen maken aan plattegrond applicatie en ontwikkelen SignalR server. Deze functionaliteit behoort bij requirement 1.18

Tevens zijn er deze iteratie wat wijzigingen gemaakt aan de requirement prioritering. Requirements 1.9, 1.10 en 1.12 hebben de prioriteit “Could have” gekregen, aangezien ze plaats moeten maken voor de nieuwe requirement 1.18. De opdrachtgever heeft namelijk liever requirement 1.18 dan de andere hier boven genoemde requirements en er is hoogstwaarschijnlijk niet voldoende tijd om al deze requirements te implementeren. Ook de requirements 1.1.3 en 1.12 zullen door deze nieuwe functionaliteit niet meer geïmplementeerd worden. Om deze requirements te kunnen realiseren zal er waarschijnlijk een volledige iteratie van twee weken nodig zijn, maar dit is niet meer mogelijk. Hierdoor hebben deze requirements de prioriteit “Won’t have” gekregen.

9.1 Ontwerpen

Om dit te kunnen realiseren wordt er gebruik gemaakt van een techniek genaamd SignalR. De keuze om SignalR te gebruiken is tot stand gekomen na een tip van Edwin. Ik heb naar aanleiding van zijn tip meer informatie over SignalR en de werking hiervan opgezocht en dit bleek precies te zijn wat we nodig hadden. SignalR is een techniek van Microsoft gebaseerd op C# en Javascript, twee programmeertalen die voor het dashboard ook gebruikt worden. Er zijn erg veel voorbeelden en documentatie over te vinden en het lijkt vrij eenvoudig te implementeren binnen een bestaand project.

Om gebruik te maken van SignalR dient er een server te worden opgezet, waar een client (webpagina) zich op kan registeren. De server houdt een lijst bij van alle geregistreerde clients en beschikt over een methode om een broadcast bericht te versturen naar al deze clients. Dit is erg dynamisch, aangezien er constant nieuwe clients kunnen worden toegevoegd aan de lijst op de server. Dit is belangrijk voor het gebruik in combinatie met het dashboard, aangezien er steeds nieuwe tegels toegevoegd kunnen worden.



Afbeelding 24: Werking SignalR client server

Deze methode is geschikt om de dashboard tegels samen te laten werken, aangezien het dashboard gebruik maakt van iframes. Iedere iframe is een aparte webpagina, waardoor de iframes individueel geregistreerd kunnen worden als clients en dus individueel berichten kunnen ontvangen. Hierdoor is SignalR een goede techniek voor de dashboard applicatie.

In afbeelding 24 verstuurt Client 1 (de grafiek applicatie) zijn huidige tijd en datum door naar de SignalR server. De server broadcast deze tijd en datum naar client 2 en 3 (de plattegronden) die vervolgens de bijbehorende data zullen tonen.

9.2 Ontwikkeling

Voor het opzetten van de SignalR server moest er een keuze gemaakt worden tussen twee mogelijkheden. De eerste mogelijkheid was om voor ieder dashboard een eigen server te starten. De tweede mogelijkheid was om één server te starten waar alle dashboards gebruik van kunnen maken.

De keuze is gevallen op de tweede optie, namelijk het gebruik van één server voor alle dashboards. Hier is voor gekozen, omdat dit in de toekomst wat extra mogelijkheden geeft, zoals het real-time uitwisselen van informatie tussen verschillende dashboards. Hier kan bijvoorbeeld gedacht worden aan een chatapplicatie.

Zoals in hoofdstuk 9.1 is besproken is SignalR gebaseerd op C# en javascript. C# wordt gebruikt voor de server kant en javascript wordt gebruikt voor de client kant. De groene klassen zijn door mij zelf gemaakt.

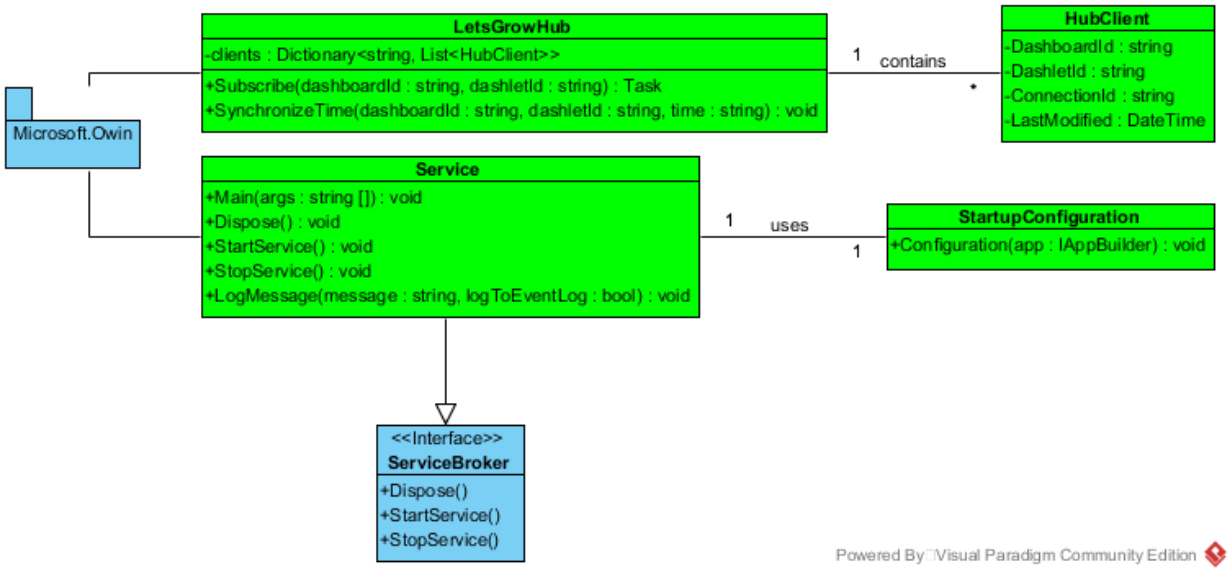


Diagram 9: Klassendiagram SignalR Server

Afbeelding 25 toont hoe een dashboard tegel (client) zich registreert bij de SignalR server:

```
signalrHubConnection = $.hubConnection(signalRServerUrl);
signalrHubConnectionProxy = signalrHubConnection.createHubProxy('LetsGrowHub');

signalrHubConnection.start()
    .done(function () {
        signalrHubConnectionProxy.invoke('Subscribe', getDashboardId(), getDashletId());
    })
    .fail();
```

Afbeelding 25: Client subscribe methode

Door deze code aan te roepen wordt de 'Subscribe' methode in de LetsGrowHub klasse aangeroepen en wordt de client toegevoegd aan de lijst met clients.

9.3 Testen

Voor het testen van de nieuwe SignalR functionaliteit is gebruik gemaakt van beslissingstabellentesten. Deze testen zijn volgens dezelfde richtlijnen opgezet als de testen in iteratie 5 en zijn te vinden in het meegeleverde document “Testontwerpen” in hoofdstuk 5. Er is gekozen voor beslissingstabellen, aangezien de werking van SignalR afhankelijk is van een aantal condities en dit goed getest kan worden met een beslissingstabellentest. De beslissingstabellentest wordt uitgevoerd in de laatste drie weken.

Om het testen te vereenvoudigen is er een extra dashboard tegel gemaakt. Deze tegel doet niks anders dan de geselecteerde tijd weergeven. Op dit moment reageren alleen de plattegrond tegels op het signaal van de grafieken, maar hier wordt nog niet dagelijks data voor opgeslagen. Dit maak het lastig om te controleren of de juiste tijd wordt doorgegeven. Hierdoor is er speciaal voor het testen een testtegel gemaakt. Deze tegel zal zich, net als de plattegrond, registeren bij de SignalR server en zal net als de plattegrond het broadcast bericht ontvangen.

Verder was het hier belangrijk om de grafiek en plattegrond opnieuw te testen, aangezien hier ook wijzigingen in aangebracht zijn. Hiervoor zijn de reeds opgezette testen gebruikt als regressietest en opnieuw uitgevoerd.

9.4 Problemen

De problemen die zijn opgetreden in deze iteratie zitten vooral in het server gedeelte van SignalR. Lokaal kan er een console applicatie gedraaid worden die dient als server. Hier kunnen de dashboard tegels zich zonder problemen op registeren. Deze console applicatie is vervolgens verplaatst naar de server van LetsGrow en dit werkt ook goed. Echter werkt LetsGrow met een plug-in structuur op de server, zodat het eenvoudiger is om dit soort applicaties te kunnen beheren. De console applicatie is hierdoor omgezet naar een plug-in. SignalR luistert continue op een bepaalde poort naar binnenkomende berichten, maar dit is met de rechten die de plug-in krijgt niet toegestaan. Doordat het oplossen van dit probleem erg veel tijd kost ben ik niet toegekomen aan het uitvoeren van de testen.

9.5 Evaluatie en mijlpalen

Deze iteratie zijn er drie mijlpalen behaald, namelijk:

Nummer	Mijlpaal
4	Definitieve versie functioneel ontwerp
5	Definitieve versie technisch ontwerp
6	Definitieve versie testontwerp

Het was de bedoeling om deze iteratie ook mijlpaal 7 af te ronden, de definitieve versie van het testrapportage. Aangezien ik, zoals beschreven in hoofdstuk 9.4, niet voldoende tijd had om de testen uit te voeren, is deze mijlpaal doorgeschoven naar de laatste 3 weken. Deze laatste drie weken waren al ingepland voor uitloop en afronding van het afstudeerverslag, dus deze uitloop qua testen zal geen problemen opleveren.

Deze iteratie was bepaald dat SignalR gebruikt zou worden om de tegels met elkaar te laten communiceren, omdat deze techniek precies bood wat we nodig hadden. Vervolgens had ik de keus gemaakt om te werken met één server waar alle dashboards gebruik van maken, zodat de SignalR functionaliteit eventueel in de toekomst ook gebruikt kan gaan worden voor andere functionaliteiten, zoals bijvoorbeeld een chatapplicatie.

10. Afronding

In dit hoofdstuk worden de laatste drie weken besproken van het afstudeertraject. De laatste drie weken gaan grotendeels besteed worden aan het afronden en verbeteren van het afstudeerverslag. Er zullen hier dan ook geen nieuwe requirements meer worden gerealiseerd.

10.1 Usability feedback

Aan de hand van de testen die Joost tussentijds heeft uitgevoerd van de dashboard applicatie, heb ik wat kleine wijzigingen doorgevoerd aan het dashboard. Zo kon de gebruiker bij het aanmaken van een dashboard een aantal opties invullen waar niks mee gedaan werd (omschrijving, dashboard categorie). Om verwarring te voorkomen zijn deze eruit gehaald.

10.2 Vrijgave advies

Verder is in deze iteratie het vrijgave advies opgesteld. Een vrijgave advies kan advies geven over het in productie nemen van een applicatie, maar ook over het doorzetten van een applicatie naar de volgende (laatste) testfase. Aangezien Joost na de testen van de ontwikkelaars altijd een gebruikersacceptatietest uit voert op het gemaakte werk, is dit ook voor het dashboard het geval. Het vrijgave advies dat ik opgezet heb is dan ook gericht op het doorzetten naar de volgende (laatste) testfase, namelijk de gebruikersacceptatietest. Joost zal na deze test in overeenstemming met Peter bepalen of de functionaliteit doorgezet gaat worden naar de ontwikkelomgeving.

Het vrijgave advies luidt dan ook: **Vrijgeven**

De overeengekomen functionaliteiten zijn gerealiseerd en door mij getest met positief resultaat. Hierdoor is het product gereed voor een laatste gebruikersacceptatietest en kan, indien deze test slaagt, worden doorgezet naar de productieomgeving.

10.3 Het dashboard

Doordat het dashboard eerst nog één testfase moet ondergaan voordat deze in productie genomen kan worden, heb ik met behulp van afbeeldingen de flow van het dashboard doorlopen. Dit is te vinden in bijlage K: “Werking dashboard applicatie in afbeeldingen”. De planning is dat het dashboard eind Januari op de productieomgeving zal verschijnen, zodat deze gepresenteerd kan worden tijdens de tuinbouw relatiedagen. Dit is de grootste vakbeurs op het gebied van glastuinbouw en vind plaats in Gorinchem.

10.4 Evaluatie en mijlpalen

In deze iteratie zijn er een aantal mijlpalen behaald, namelijk:

Nummer	Mijlpaal
7	Definitieve versie testrapportage
8	Definitieve versie afstudeerverslag

Mijlpaal 7 stond ingepland voor iteratie 6, maar aangezien ik niet aan het testen ben toegekomen deze iteratie, is de testrapportage in de laatste iteratie afgerond. Zoals besproken is deze iteratie ook gebruikt om het afstudeerverslag af te ronden.

Ik ben uiteindelijk zeer tevreden over het eindresultaat. Niet alleen qua opdracht maar ook qua verslag.

11. Evaluatie

Deze evaluatie is opgedeeld in twee delen, de proces evaluatie en de product evaluatie. In de proces evaluatie kijk ik terug op het gehele proces en in de product evaluatie kijk ik terug op de realisatie van het dashboard pakket.

11.1 Proces evaluatie

Als ik terug kijk op het hele proces ben ik erg tevreden. Het was een goede keus om in iteraties van twee weken te werken, aangezien er zo voldoende momenten waren om de requirements naast elkaar te leggen en te bepalen of de prioritering nog steeds juist was. Zo zijn er tijdens het proces dus een aantal requirements bijgekomen waardoor de andere requirements een lagere prioriteit kregen.

In het begin heb ik helaas veel te veel aandacht besteed aan het testen. Testen begon een doel op zich te worden, maar dat was niet te bedoeling. Gelukkig heb ik dit tijdens de conceptbespreking met mevrouw Lousberg besproken en ben ik een andere weg ingeslagen.

Door de nieuwe requirement van de plattegrond tegel heb ik veel samen gewerkt met Eray, de ontwikkelaar van de plattegrond applicatie. Dit is erg goed verlopen.

Tot slot zijn de laatste drie weken voornamelijk gebruikt voor het afronden van het afstudeerverslag.

11.2 Product evaluatie

Ik ben zeer tevreden over het eindresultaat. De doelstelling is in de loop van het project wel iets gewijzigd, waardoor de data uit externe bronnen niet meer geïmplementeerd zou worden binnen het dashboard. In plaats daarvan is de meter applicatie, plattegrond en meter tegel, en tegel communicatie binnen het project geïmplementeerd. Deze functionaliteiten zijn door de opdrachtgever beoordeeld met een hogere prioriteit, waardoor de requirements met betrekking tot data uit externe bronnen zijn komen te vervallen. Dit is echter erg goed uitgepakt en dit alles bij elkaar heeft een mooi dashboard opgeleverd.

Op basis van het onderzoek naar een geschikt dashboard pakket is de opdrachtgever geadviseerd over de mogelijkheden. Het advies heeft geleid tot de daadwerkelijke aanschaf van het pakket.

De complexiteit van de opdracht zat hem voornamelijk in het bedenken van diverse weergave mogelijkheden, het combineren van verschillende programmeertalen en het koppelen van de verschillende applicaties aan de nieuwe dashboard applicatie.

11.3 Beroepstaken

In dit hoofdstuk wordt per beroepstaak beschreven wat er gedaan is om aan de beroepstaak te voldoen.

1.3 Selecteren standaard Software -- niveau 3

Tijdens het afstudeertraject is er onderzoek gedaan naar een geschikt dashboard pakket voor LetsGrow.com (Iteratie 1). Het onderzoek is volgens de richtlijnen van KPMG uitgevoerd en zo is het onderzoek opgesplitst in verschillende delen. Het eerste onderdeel is het vooronderzoek, waar de requirements en visie van het bedrijf worden bekeken. Op basis hiervan zijn een aantal knock-out criteria opgesteld waar het pakket minimaal aan moet voldoen.

Op basis van deze knock-out criteria is er een longlist tot stand gekomen. De pakketten in deze longlist voldoen allemaal aan de minimale eisen, waardoor de keuze al een stuk beperkter is geworden. Op basis van deze longlist is er dieper ingegaan op de geselecteerde pakketten om te bepalen welke van deze pakketten het beste geschikt is. Dit resulteerde uiteindelijk in een dashboard pakket gebaseerd op C#, ASP.Net, HTML5 en Javascript. Doordat veel LetsGrow applicaties ook zijn opgebouwd met C#, ASP.Net Javascript was dit pakket erg geschikt.

1.4 Uitvoeren analyse door definitie van requirements -- niveau 3

Om te achterhalen waar het dashboard aan moest voldoen, was het eerst noodzaak om de requirements te achterhalen. Dit heb ik gedaan door een aantal gesprekken te houden met de opdrachtgever en mijn begeleider. Ter voorbereiding van deze gesprekken zijn een aantal vragen opgesteld, die vervolgens besproken zijn en geleid hebben tot een aantal requirements. Deze gesprekken zijn bijgevoegd in bijlage E. Op basis van deze eerste gesprekken is een initiële lijst requirements voortgekomen. Deze lijst diende als basis om het onderzoek en de eerste iteratie uit te voeren.

Tijdens het proces zijn er meerdere keren nieuwe requirements bijgekomen, waardoor er opnieuw geprioriteerd moest worden.

3.2 Ontwerpen systeemdeel -- niveau 3

Voor het ontwerpen van een systeemdeel is de meter applicatie ontwikkeld. Deze functionaliteit was nog niet beschikbaar binnen LetsGrow. Aangezien de opdrachtgever ook graag meters op het dashboard zou willen tonen, was het noodzaak dat deze meters eerst aangemaakt konden worden. De klant kan nu naar eigen wens een meter kiezen en configureren. Deze geconfigureerde meter kan vervolgens op het dashboard worden getoond.

Ook is er een server applicatie gemaakt. Deze applicatie is nodig om de dashboard tegels onderling met elkaar te kunnen laten communiceren. De dashboard tegels registeren zich bij deze applicatie en kunnen vervolgens een tijd naar de server sturen of een broadcast bericht van de server ontvangen.

3.3 Bouwen applicatie -- niveau 4

Een grote uitdaging bij het ontwikkelen was het combineren en implementeren van de verschillende applicaties van LetsGrow op het dashboard. De complexiteit lag dan ook niet zo zeer op het tonen van de tegels op het dashboard, maar vooral het selecteren en configureren van een dashboard tegel.

Er is ontwikkeld in Visual Studio 2015 met de bijbehorende versiebeheertool TFS. Het dashboard pakket, opgebouwd met het MVC framework, is volledig naar de wens van de opdrachtgever aangepast. Om het dashboard pakket goed te kunnen integreren met de huidige situatie zijn er ook een aantal wijzigingen

gemaakt aan de huidige situatie. Zo zijn er bijvoorbeeld wijzigingen gemaakt aan de grafiek applicatie, aangezien deze niet goed bruikbaar was op het dashboard. Voor de ontwikkeling is gebruik gemaakt van diverse programmeertalen, zoals C#, ASP.Net, Javascript, HTML, CSS

3.5 Uitvoeren van en rapporteren over het testproces -- niveau 3
Deze requirement is toegevoegd als compensatie voor beroepstaak 3.2. Echter is deze beroepstaak niet volledig weggevallen en is 3.5 een extra behaalde beroepstaak geworden. In eerste instantie werd er door mij te zwaar getest (zie hoofdstuk 8.3), want een minder zware testmethode bleek ook te voldoen. Ik heb toen besloten om af te stappen van de zware algoritmetesten en verder te gaan met een beslissingstabellentest.

Voor het testen zijn diverse documenten opgezet, zoals het mastertestplan, het detailtestplan, de testontwerpen en de testrapportage. De laatste drie zijn steeds (indien nodig) uitgebreid tijdens de iteraties. In het mastertestplan is er met behulp van een testrisicoanalyse bepaald welke onderdelen het hoogste risico vormden en dus het zwaarst getest moesten worden.

In het detailtestplan is er bepaald welke testmethodieken bij de risicogebieden konden worden toegepast en hiermee zijn vervolgens de testen opgesteld en beschreven in het meegeleverde document genaamd testontwerpen. In het testrapport zijn de resultaten van de testen beschreven.

Begrippen

Begrip	Betekenis
Klimaatcomputer	Computer die wordt gebruikt om alle meetgegevens van de verschillende meetboxen te verzamelen en te verwerken.
Meet box	Apparaat dat wordt opgehangen in de kas en zijn meetgegevens verstuurd naar de klimaatcomputer.
URL	Uniform Resource Locator – een url verwijst naar een bepaalde webpagina en wordt gebruikt om data op te halen.
Team Foundation Server (TFS)	TFS is een versiebeheertool dat wordt meegeleverd met Visual Studio. Met deze software wordt er onderscheid gemaakt tussen verschillende versies van het ontwikkelde programma en kan eenvoudig code gedeeld worden.
Shelveset	Met behulp van een shelveset kunnen wijzigingen in de code worden gedeeld (met iedereen die toegang heeft tot de TFS) zonder de code daadwerkelijk op de server in te checken.
Cloud	De Cloud wordt gebruikt voor het opslaan van gegevens, software en bestanden. Dit wordt de Cloud genoemd, aangezien niet exact bekend is waar deze content wordt opgeslagen.
Iframe	Html element dat gebruikt wordt om een webpagina in te tonen
Work-around	Alternatieve methoden om een bepaalde taak uit te voeren, maar in de zelfde uitkomt resulteert.
Resource bestand	Bestand dat wordt gebruikt om teksten van een applicatie op een centraal punt te kunnen beheren. Verder worden deze bestanden gebruikt om LetsGrow en het dashboard in meerdere talen te laten werken, door bijvoorbeeld gebruik te maken van resource bestanden genaamd: Resources.nl.resx(Nederlands) of Resources.es.resx(Spaans). Automatisch wordt dan het juiste bestand ingeladen en wordt de gehele applicatie in de juiste taal weergeven. Indien er geen vertaling beschikbaar is in de gewenste taal, zal de vertaling uit het standaard resource bestand gebruikt worden. Binnen LetsGrow gaat dit om de taal Engels. De naam van dit standaard resource bestand is Resource.aspx.
Library	Een (javascript) bestand dat binnen de code gebruikt kan worden voor het toevoegen van extra opties en functionaliteiten.
Definitie	Een definitie is een object dat binnen LetsGrow wordt gebruikt. Een definitie bevat wat configuratie attributen en linkt dit aan bepaalde data. In principe is alles binnen LetsGrow een definitie, zo zijn grafieken een definitie, maar ook de items in bijvoorbeeld de menuboom.
Tegel	Met een tegel wordt een individueel item op het dashboard bedoeld. Dit kan bijvoorbeeld een grafiek, rapport of meter zijn.
Amsterdam Power Exchange (APX)	APX is een beurs waar energie verhandeld wordt.
Energie-managers	Energie managers regelen de in- en verkoop van energie voor de klanten van LetsGrow.

Bronnen

LetsGrow.com (2000)
Geraadpleegd op:
http://www.letsgrow.com

Indora - Software- en leverancierselectie een korte introductie (Indora 2004)
Geraadpleegd op:
http://www.ictaccountancy.nl/downloads/INDORA_Software_en_leverancierselectie.PDF

Factsheet – Selectie van Standaardsoftware
Geraadpleegd op:
http://itinbedrijf.info/0208_KPMG.pdf

Pakketselectie: de begin- en de eindfase uitgelicht(C-2002-3-Heijer)
Geraadpleegd op:
https://www.compact.nl/articles/pakketselectie-de-begin-en-de-eindfase-uitgelicht/

What are Progressive Web Apps? (2016)
Geraadpleegd op:
http://blog.ionic.io/what-is-a-progressive-web-app/

Are Progressive Web Apps the Future? (2012)
Geraadpleegd op:
http://developer.telerik.com/featured/are-progressive-web-apps-future/

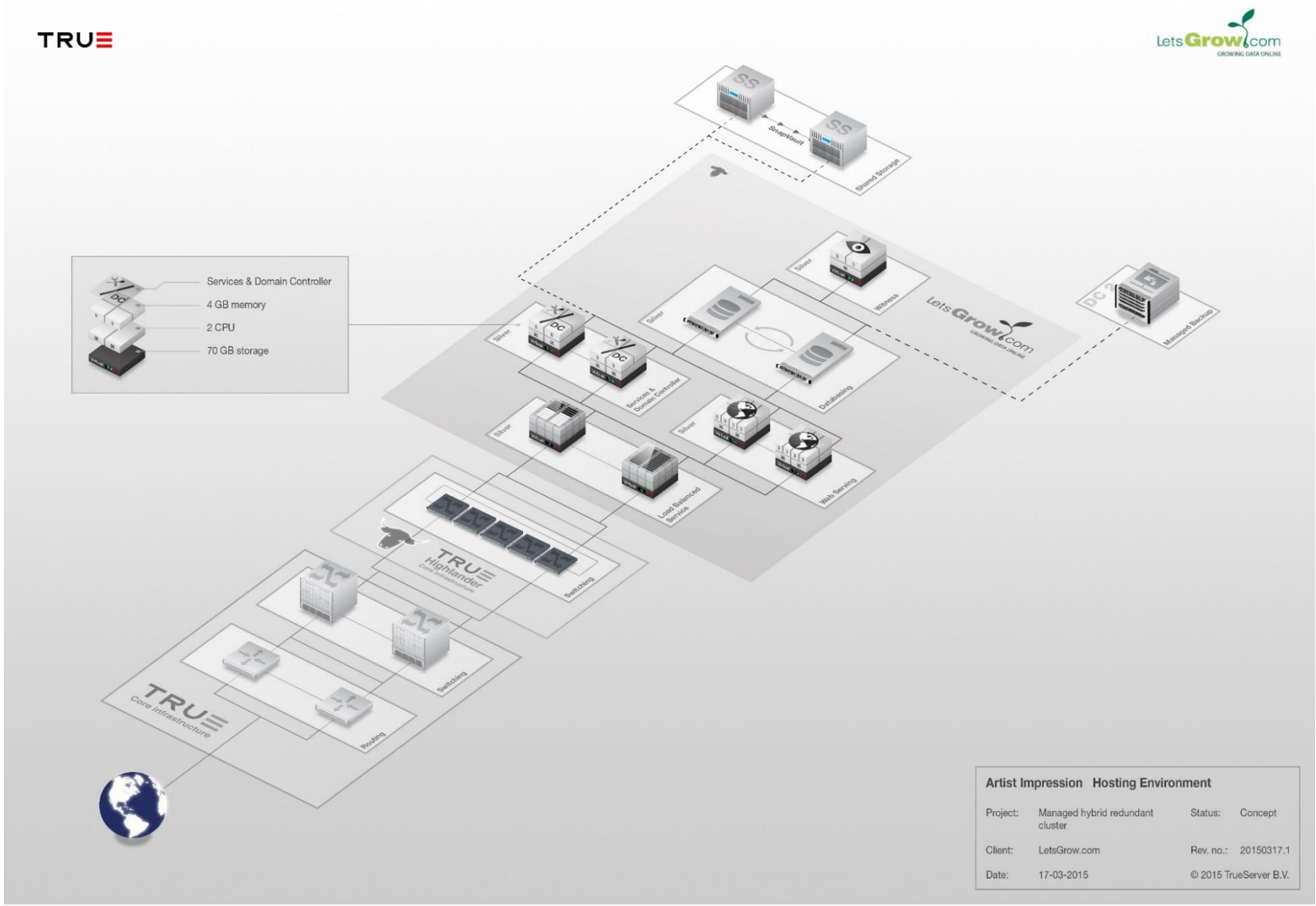
Progressieve Web Apps: de toekomst van mobiele apps? (2016)
Geraadpleegd op:
http://www.emerce.nl/achtergrond/progressive-web-apps-de-toekomst-van-mobiele-apps

IfSQ Level-1: An Entry-Level Standard (Graham Bolton, Stuart Johnston, 2008)
Geraadpleegd op:
http://www.ifsq.org/level-1.html

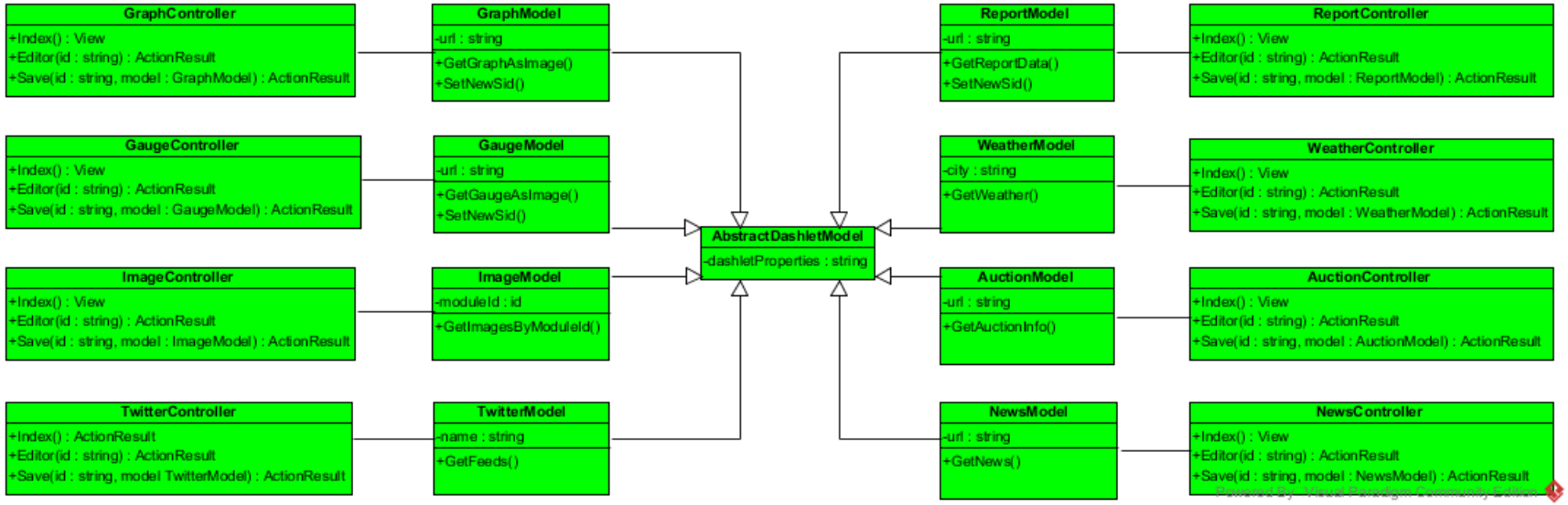
De overige bronnen met betrekking op de inhoud van het onderzoek, worden vermeld in het onderzoeksrapport.

Bijlages

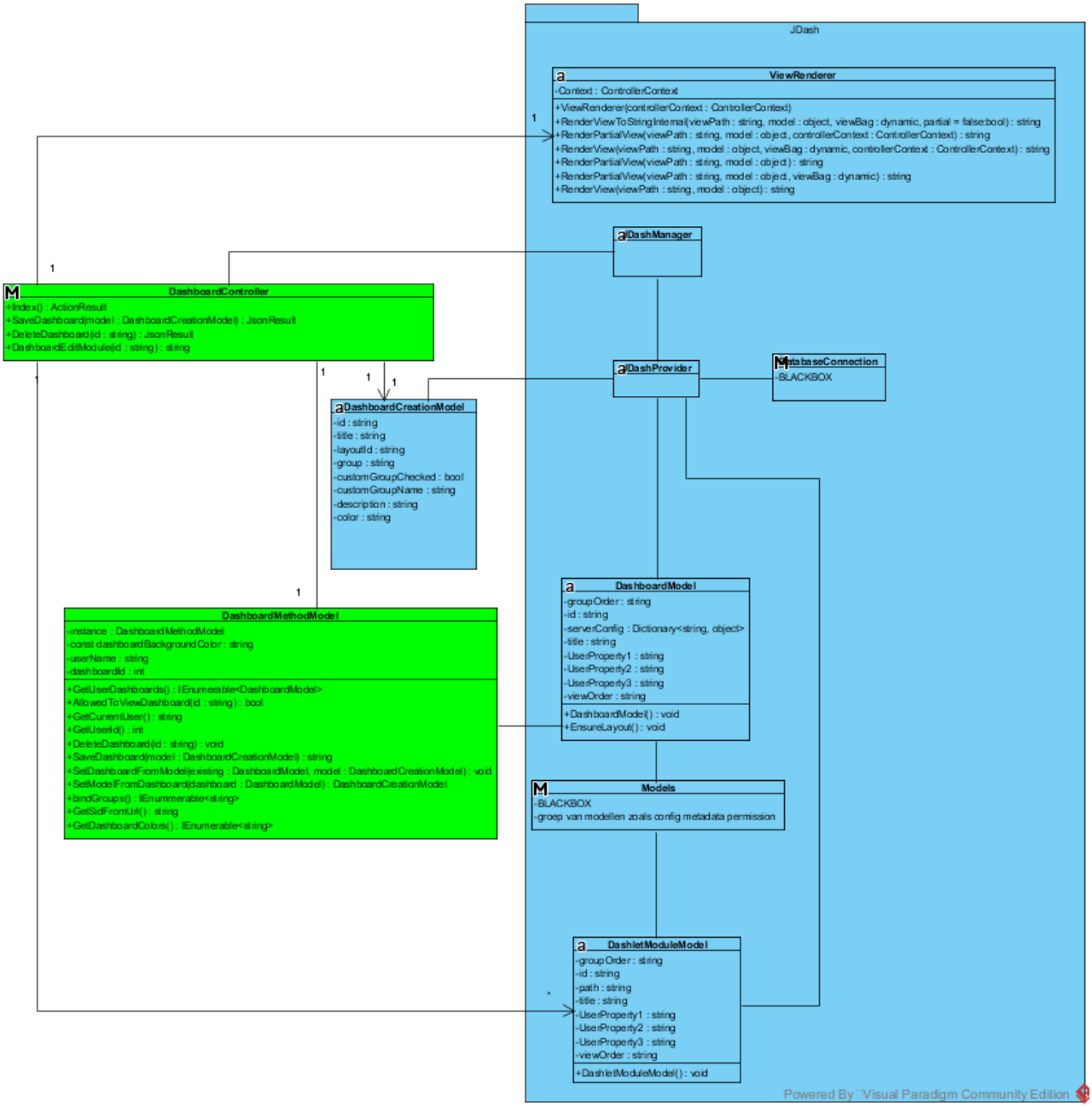
A. Bijlage Infrastructuur LetsGrow



B. Bijlage Analyse dashboard tegel klassendiagram



C. Bijlage Design klassendiagram



D. Bijlage Codereviews

Iteratie 2: Opstart ontwikkeling

Uitkomst codereview

Hieronder de feedback die ik heb ontvangen Edwin Vis met betrekking tot de implementatie van het dashboard.

Algemeen
Boven classes commentaar toevoegen; geef aan wat de functie is van de class.

DashboardMethodModel.cs
Regel 115: Methods/Functions met initiele hoofdletter.
Is het wel een best practice om hier logica als het Saven en Getten van dashboards in onder te brengen. Ik verwacht dat meer in de controller….?

DashboardController.cs
Regel 80, 101: Methods/Functions met initiele hoofdletter.

Verder heb ik eigenlijk niets op te merken behalve dat je de AuthorizationModule voorlopig niet in moet checken.

Oplossing

Algemeen:
Hier had ik niet meer aan gedacht voordat ik de wijzigingen had ingecheckt, dus na dit toegevoegd te hebben was dit puntje opgelost.

DashboardMethodModel.cs
Er is gekozen om deze methodes in de model klasse te plaatsen, aangezien er interactie is met de database. Door deze reden heb ik het saven e.d. in de model klassen laten staan.

DashboardController.cs
Deze methodes zijn onderdeel van het dashboard pakket, maar de methodenamen zijn aangepast om het overeen te laten komen met de rest van de structuur.

Wat betreft de AuthorizationModule, hier heb ik een wijziging moeten maken om te voorkomen dat het controleren van de rechten van de gebruiker wordt overgeslagen. Echter maken veel verschillende applicaties binnen LetsGrow gebruik van deze module, dus dient er bepaald te worden of deze wijziging kwaad kan voor overige projecten. Tot die tijd wordt deze module binnen het dashboard project handmatig bijgewerkt op de server.

Iteratie 3: Grafieken en rapporten

Uitkomst codereview algemeen

Het eerste deel van de codereview is gedaan door Leon Batta en het tweede deel van de review is gedaan door Edwin Vis.

Models\DashboardCreationModel.cs

- Geen wijzigingen: niet inchecken

Models\DashboardMethodModel.cs

- Ln 50 lege catch: kennelijk heb je rechten op een dashboard dat niet bestaat: moet je die niet opruimen?

Scripts\dashlets\Report\index.js

- Lege file: is deze nodig?

Scripts\dashlets\Shared\sharededitorpanel.js

- Ln 27: selectie op strings: beter met id's
- Ln 45: selectie op tekststring "Maak uw keuze": reeds besproken
- Algemeen: GrowingGroup: zie commentaar Edwin
- Algemeen: RedSphere zou ik meer omschrijven als "OtherGraphs" ("ModuleGraps", "CropGroupGraphs")

Web.config

- Debug staat op true

GraphController.cs
Naamgeving: met GrowingGroup bedoel je waarschijnlijk CropGroup?
Het is beter om de ingeburgerde namen te gebruiken anders wordt het al snel verwarrend.

ReportController.cs
Naamgeving: met GrowingGroupbedoel je waarschijnlijk CropGroup?
Het is beter om de ingeburgerde namen te gebruiken anders wordt het al snel verwarrend.

AbstractDashletModel.cs
Regel 44: Volgens mij gaat dit mis als je sid ip SID gebruikt. Equals is namelijk ordinal (case-sensitive).
Regel 67: teksten dienen uit resource files te komen.

GraphModel.cs
Regel 14: Haal dit soort urls op uit de environmentsettings.config (omgeving specifiek).

ReportModel.cs
Regel 14: Haal dit soort urls op uit de environmentsettings.config (omgeving specifiek).

Oplossing

Models\DashboardCreationModel.cs
Geen wijzigingen benodigd.

Models\DashboardMethodModel.cs
De reden dat hier gebruik gemaakt was van een try/catch waren wat problemen met de database. Wat oude autorisatie informatie bleef bewaard wanneer er een dashboard werd verwijderd. Dit probleem was al opgelost, alleen is deze try/catch blijven staan. Deze kon dus verwijderd worden.

Scripts\dashlets\Report\index.js
De dashboard applicatie is zo opgebouwd dat iedere dashboard tegel een index.cshtml en een editor.cshtml pagina heeft. Om javascript gescheiden te houden van de html, dient er per html pagina een javascript bestand te worden aangemaakt waarin de benodigde scripts geplaatst kunnen worden. Indien dit bestand niet wordt aangemaakt zal er een javascript foutmelding gegeven worden dat het gezochte bestand niet kan worden gevonden. Het is dus nodig om dit bestand aan te maken.

Scripts\dashlets\Shared\sharededitorpanel.js
Hier maakte ik gebruik van een string om te controleren of het eerste item van de dropdown lijst was geselecteerd. Echter is dit niet de meest optimale methode, dus deze code is gewijzigd zodat er op een integer wordt gecontroleerd in plaats van een string.

RedSpheres is hernoemd naar OrtherDefinitions. OrtherGraphs is geen goede naam, aangezien ook de rapport en meter applicatie gebruik zal maken van “RedSpheres”. Ik had in eerste instantie gekozen voor deze naam, aangezien deze items binnen het bedrijf “Rode bol items” genoemd worden. Aangezien dit slechts binnen het bedrijf zo genoemd wordt is er in overleg met Leon Batta voor gekozen om de “RedSpheres” te hernoemen naar OrtherDefinitions.

Web.config
Wanneer de debug op yes staat in de web.config en het project wordt ingecheckt, dan worden er allerlei extra bestanden aangemaakt die het debuggen mogelijk maken. Dit is voor een release versie op de testomgeving natuurlijk niet nodig (aangezien het testen door de programmeur op de lokale machine gebeuren). Deze is dus op false gezet voordat het project werd ingecheckt.

GraphController.cs en ReportController.cs
Binnen LetsGrow wordt er gebruik gemaakt van de term CropGroup wanneer er gesproken wordt over een teeltgroep (groep met tuinders/adviseurs).

AbstractDashletModel.cs
Hier had ik nog geen problemen mee ondervonden, maar om te voorkomen dat het een keer fout kan gaan is dit ook opgelost.

GraphModel.cs en ReportModel.cs
Aangezien de ontwikkel en testomgeving beide tegen de testdatabase aanpraten had ik in eerste instantie een deel van de op te bouwen URL als vaste tekst had neergezet. Bij het builden van de applicatie worden er verschillende configuratiebetanden samengevoegd, zodat de applicatie ook op de productieomgeving geplaatst kan worden. Op de oude manier was dit niet mogelijk, maar ik heb nu deze URL in configuratie betand (de EnvironmentSettings) geplaatst en zo kan het project ook gebuild worden voor de productieomgeving.

Uitkomst codereview grafiekapplicatie

Deze codereview is uitgevoerd door Leon Batta

Default.aspx.cs

- Ln 571 en 579: Deze methodes zijn bijna gelijk. Als je aan “CreatListItem” “QueryString.GetQuery(queryStringCategory)” mee geeft, ipv “queryStringCategory”, dan is hij gelijk aan “CreateListItemNotification”.
- Zijn de iconen uit het menu weggevallen?

Web.config

- Niet inchecken met debug=true

Versie ophogen

HighChartsCustomJavaScriptFunctions.js

- Ln 121: uitgecommentarieerde code weg
- Ln 1025: Ik heb de indruk dat “addCustomMinimalLegend” een kopie is van “addCustomLegend”. Dat lijkt me niet de goede weg. Je moet een style zetten op de kolommen die niet getoond moeten worden.

Oplossing

Default.aspx.cs

Ik had een kleine methode gemaakt voor het aanmaken van een list item. Voor het notificatie listitem was er een iets andere opmaak nodig, dus had ik dezelfde methode gebruikt en een klein beetje aangepast. Door de tip van Leon is hier een betere versie voor gekomen.

De iconen waren niet weggevallen.

Web.Config

Zie opmerking bij web.config oplossing 3.2

Versie ophogen

Aangezien de grafiekapplicatie een flinke wijziging had ondergaan was het nodig het versienummer op te hogen. Hierdoor is het bij klanten ook duidelijk dat er een grote wijziging gemaakt is.

HighChartsCustomJavascriptFunctions.js

Hier had ik de bestaande methode voor het tekenen van de legenda gebruikt en hier de elementen uit verwijderd die niet getoond hoefde te worden indien het om een smalle weergave van de grafiek ging.

Na deze opmerking heb ik de elementen een style gegeven en vervolgens aan de hand van de scherm breedte bepaald of een element wel of niet zichtbaar was.

Iteratie 4: Meters en plattegronden

Uitkomst codereview

Deze codereview is uitgevoerd door Edwin Vis

Algemeen:
In aspx prefix de variabele naam met het type control; een DropDownList met de naam Areas -> DropDownListAreas
Gebruik dan ook deze naam in de events: DropDownListAreas_SelectedIndexChanged

Gebruik constants voor nummers en strings.

SelectItem.aspx.nl.res:
- Selecteer èèn item om als meter te weergeven -> “èèn” met streepjes de andere kant op.
-

Statusmeter.js:
- houd de naamgeving consistent; bv. lightdiv -> divLight

ConfigureItem.aspx.cs:
- regel 313 commentaar verwijderen

Default.aspx.*, style.css
- geen wijzigingen

EnvironmentSettings.config:
- Er zijn een tweetal nieuwe settings toegevoegd. Deze zie ik niet terug in Test- of ProductionSettings.config

Gauge.aspx:
- Heb je al deze javascripts wel nodig?

Gauge.aspx.cs:
- Page_Load is behoorlijk lang; beter is om de functie op te splitsen in kleinere functies.
- Regel 142: Functie GetStringFromResource is wat overkill. Gebruik gewoon de GetGlobalResourceObject.

SelectItem.aspx:
- verwijder commentaar

Oplossing

Algemeen:
Een foutje dat er tussendoor is geschoten na het wijzigen van de namen van de velden. DDLAreas hernoemd naar DropDownListAreas

Aangezien er in de codebehind veel html elementen worden aangesproken wordt er aan de hand van het id van het element het te bewerken element opgehaald en vervolgens kan het element aangesproken worden. Aangezien op een aantal plekken hetzelfde element wordt aangesproken kan het id van het benodigde veld beter als constante worden gedefinieerd voor het geval dat het id ooit mocht wijzigen.

Statusmeter.js
De overige elementen zijn gedefinieerd in camel case, maar bij deze variabele had ik dit niet gedaan. De naam lightdiv is gewijzigd naar divLight en is nu weer conform de rest van de elementen.

EnviromentSettings.config
Dit bestand wordt door de deploy tool gevuld met content uit de test of productiesettings, afhankelijk van waar het project naartoe gedeployed wordt. Deze linkjes zijn voor de productieomgeving en voor de testomgeving verschillend, dus vandaar dat deze in beide documenten gedefinieerd moeten worden.

De EnviromentSettings bevat de instellingen voor de testomgeving, dus vandaar dat ik hier tot nu toe geen last van gehad had.

Gauge.aspx
Er stonden hier inderdaad wat meer scripts gedefinieerd dan er gebruikt werden, dus een aantal van deze scripts zijn hier weggehaald. Deze scripts kwamen uit een voorbeeld van een highcharts grafiek die later toch niet gebruikt is, maar deze referenties naar de script bestanden waren blijven staan.

Gauge.aspx.cs
Deze methode is nu onderverdeeld in meerdere methodes voor de overzichtelijkheid. Ook hier zijn er een aantal constanten aangemaakt.

Ik maakte gebruik van een methode voor het ophalen van de resource bestanden, omdat de GetGlobalResourceObject methode naar mijn mening iets meer werkt vereist. Hier moet namelijk de naam van het resource bestand aan meegegeven worden, de key van de resource en vervolgens moet het object geparsed worden naar een string. De methode die ik had gemaakt retourneerde direct de string. De naam van het resource bestand is altijd hetzelfde, dus ik kon hier altijd dezelfde naam gebruiken. Aangezien er verder in de code overal gebruik gemaakt wordt van de GetGlobalResourceObject methode heb ik ervoor gekozen om deze methode verder ook te gebruiken.

Iteratie 7: Dashboard grafische interface

Uitkomst codereview

AbstractDashletMethod.cs:
Regel 118: Je haalt de naam van een template op maar doet er niets mee. Beetje jammer van het uitstapje naar de database….

Regel 150: Volgens mij kan dit makkelijker en generieker via HttpUtility.HtmlDecode

Sharededitorpanel.js
Zoals net ook bekeken werkt de

```
$(document).ready(function () {
```

functie niet. Kan het zijn dat deze functie onderdeel moet zijn van een constructie als:

```
define(["require"], function (require) {
    return {

        initialize: function (context, node) {
```

zoals ook geldt voor scripts als editor.js…?

Oplossing

AbstractDashletMethod.cs
Deze methode bleek achteraf niet nodig, maar is per ongeluk blijven staan.

Ik deed een string replace om de vreemde tekens uit de string te halen, maar dit bleek veel eenvoudiger te kunnen met de ingebouwde methode HtmlDecode.

Sharededitorpanel.js
Er waren wat problemen met de $(document).ready( ){ } en dit is op een aantal plekken weg gehaald.

Iteratie 6: Tegel communicatie

Uitcomst codereview

AuthorizationModule
- Geen opmerkingen; deze kan inderdaad ook mee.

HighCharts
HighChartsCustomJavascriptFunctions.js
- Is het de bedoeling dat voorlopig de test date wordt gebruik?
- De functies subscriber en sendTime hebben een wat vage naam. Ik zou er iets van maken als subscribeToSignalRHost en synchronizeTime.

Default.aspx
- 11: Localhost:8080 -> waar is dat voor? Moet naar een config file of kan er helemaal uit.
-

DashboardController.cs
- 24: Commentaar verwijderen

DashboardUI
- Geen opmerkingen (behalve dan dat niet alles in de shelfset daadwerkelijk is aangepast).

GreenHousePlanUI
ViewPlan.js
- 100 en verder: hou de implementatie van de SignalR subscription gelijk aan die van ChartsUI. Je doet hier hetzelfde. Wellicht kun je de code ook in een aparte js onderbrengen.

ViewPlan.aspx
- Gebruik de minimized version van ViewPlanScript.js

Packages
- Exclude alle packages als Newtonsoft ed. Dit willen we allemaal niet in TFS hebben. De build server download de benodigde packages zelf.

Wacht even met inchecken van de (aangepaste) code zodat we eerst een versie kunnen maken voor productie waar de volgende aanpassingen in zitten:
- Bootstrap responsive menu
- Aanpassing van Leon mbt sorteren punten
- Aanpassing Rik tbv navigatie images

Voor de plattegrond geldt iets dergelijks.

Als je wilt testen met de aangepaste code dan moet je dit lokaal even bouwen. Dan zetten we dat op test.

Oplossing

HighChartsCustomJavascriptFunctions.js
Aangezien er slechts op een specifieke tijd demodata beschikbaar is voor de plattegrond, namelijk op 13 oktober 2016 om 07:55, wordt deze tijd steeds gebroadcast tijdens het communiceren tussen de tegels. Dit wordt voornamelijk gebruikt voor het testen en zal op het moment dat de plattegrond meer data beschikbaar heeft worden omgezet zodat de juiste tijd wordt meegegeven.

Methodenamen zijn inderdaad wat vaag, deze zijn aangepast

Default.aspx
Deze regel is er doorheen geslopen tijdens het klaarmaken voor de testomgeving. Deze regel is aangepast zodat er afhankelijk van de omgeving waar het op gebruikt wordt de juiste URL krijgt.

GreenHousePlanUI
Het is inderdaad netter om de code gescheiden te houden van de rest. Dit is aangepast.

ViewPlan.aspx
In de plattegrond werd een geminimaliseerde versie van dit javascript bestand gebruikt. Aangezien ik hierin geen wijzigingen kan maken ivm de leesbaarheid van een geminimaliseerd javascript bestand heb ik wijzigingen aangebracht in de normale viewplanscript.js. Dit document dient echter nog om te worden gezet naar viewplanscript.min.js en dan dient dit bestand gebruikt te worden.

E. Bijlage Gesprekken

Gesprek 1:

Dit gesprek is gehouden om een eerste indruk te krijgen van wat er van het dashboard wordt verwacht. Voor dit gesprek zijn een aantal vragen opgesteld en deze zijn beantwoord en besproken tijdens het gesprek. Dit gesprek heeft plaatsgevonden op 30-08-16 om 10:00.

Voor er echt begonnen werd over het dashboard heb ik een aantal vragen gesteld over de huidige situatie en de manier van werken. Hierna heb ik een aantal vragen gesteld over het te ontwikkelen dashboard.

Hoe steekt de huidige situatie in elkaar?
Klantenportal draait in een Joomla webframe

Het openen van grafieken, rapport en dergelijke is een aparte applicatie.

Bij joomla login wordt er een Sessie ID(SID) aangevraagd. Er wordt een call gemaakt naar een data webservice en deze geeft een SID terug. Joomla start met de SID de klantenportal. Bij elke call wordt de SID meegegeven om zo de authenticatie te regelen bij aanvragen van grafieken en overzichten.
De SID verloopt normaalgesproken na 40 minuten, maar deze wordt bij iedere call weer ververst.

2 mirror db, 2 webservers, 2 active directories, 2 loadbalancers
Service op 1 v/d 2 databases (op de mirror)
Test omgeving is 1 omgeving waar alles draait op een virtual machine

Wordt er gewerkt met een ontwikkelstraat binnen het bedrijf?
De grafiek applicatie wordt ontwikkelt op de pc van de software ontwikkelaar en daarbij wordt gebruik gemaakt van de test omgeving. Er wordt dus wel gebruik gemaakt van verschillende omgevingen, maar deze draaien niet allemaal op aparte servers en de ontwikkel en de testomgeving overlappen elkaar deels, aangezien de data van de testomgeving ook gebruikt wordt voor de ontwikkelomgeving. Geen acceptatieomgeving, aangezien de kosten en tijd niet opwegen tegen het gebruik hiervan.

Hoe worden de ontwikkelde functionaliteiten getest?
Globaal getest geen testplan en niet oude functionaliteit opnieuw testen. Bewuste keus risico ingeschat in verband met imagoschade.

Wat voor ontwikkelmethodiek wordt er binnen het bedrijf gebruikt?
Zelf bedenken hoe ik ontwikkelmethodiek kan toepassen. LetsGrow gebruikt eigen interpretatie van scrum.

Wat zien jullie voor je bij een dashboard pakket?

- Opgedeeld in delen
- Informatie laten zien, niet te ingewikkeld
- Eenvoudig te bedienen
- Uitnodigend

Wat zal er op het dashboard getoond gaan worden?

- Om te beginnen rapporten grafieken

Wat heeft de gebruiker voor mogelijkheden op het dashboard? Volledig aanpasbaar of een kant en klaar dashboard met wat opties?

Klant moet zelf dashboard kunnen inrichten, geen extra werk voor servicedesk

- Item toevoegen
- Item verwijderen
- Item inklappen voor beter overzicht
- Item vergroten en verkleinen

Hebben jullie voor nu nog opmerkingen?

- Nadenken wat er met huidig klantmenu gaat gebeuren en hoe deze in het nieuwe plaatje past. moet voorlopig blijven.
- Direct dashboard tonen na inloggen een mogelijkheid?

Aan de hand van dit gesprek zijn de eerste paar requirements opgesteld, onderverdeeld in functionele- en niet-functionele requirements.

Functionele requirements

Nr.	Requirement
1	Het moet mogelijk zijn om een nieuwe tegel toe te voegen aan het dashboard
2	Het moet mogelijk zijn om een rapport toe te voegen aan het dashboard
3	Het moet mogelijk zijn om een grafiek toe te voegen aan het dashboard
4	Het moet mogelijk zijn om een tegel te verwijderen van het dashboard
5	Het dashboard moet de mogelijkheid bieden om terug te keren naar het oude klantmenu
6	De tegels op het dashboard moeten kunnen worden verplaatst
7	De tegels op het dashboard moeten kunnen worden open en dichtgeklapt
8	De tegels op het dashboard moeten kunnen worden vergroot en verkleind

Niet-functionele requirements

Nr.	Requirement
9	Het dashboard moet direct getoond worden nadat de klant in logt
10	Het dashboard moet responsive zijn
11	Het dashboard moet in weinig klikken te bedienen zijn
12	Een dashboard tegel moet functioneel beperkt zijn, mag maar een ding doen
13	Het dashboard moet er uitnodigend uit zien
14	Het dashboard moet eenvoudig te bedienen zijn
15	De content op het dashboard moet worden onderverdeeld in tegels
16	Het dashboard moet de stijl van LetsGrow implementeren.
17	Het dashboard moet in meerdere talen beschikbaar zijn
18	Het dashboard moet geïntegreerd worden met het huidige klantmenu

Gesprek 2:

Bij gesprek twee is de lijst requirements uit gesprek 1 meegenomen en geprioriteerd in overleg. Dit gesprek is gevoerd met Leon Batta.

Dit gesprek heeft plaatsgevonden op 31 augustus, een dag na gesprek 1. Zo had ik genoeg tijd om het gesprek uit te werken als requirements en deze met dit gesprek mee te nemen. Dit gesprek is gehouden om 13:00 uur.

Is de naam "Tegels" een goede benaming voor de items op het dashboard?

- Ja goed

Buiten de genoemde tegel nog meer mogelijkheden?

- Afbeeldingen van de warmtecamera (foto's)

Als een klant veel dashboard tegels heeft is het veel scrollen en onoverzichtelijk. Handig om meerdere dashboards te kunnen maken?

- Ja, en dan kunnen kiezen welk dashboard je wilt openen. Naam geven voor de duidelijkheid

Ik heb de grafieken bekeken op LetsGrow en hier zitten soms heel veel details in die waarschijnlijk op het dashboard niet te lezen zijn. Is het handig als je op de grafiek kan klikken en dan de huidige grafiekapplicatie wordt geopend waar de grafiek te zien is zoals het nu is, met alle details?

- Ja

Functionele requirements

Nr.	Requirement
1	Het moet mogelijk zijn om een afbeelding toe te voegen aan het dashboard
2	Het moet mogelijk zijn om meerdere dashboards aan te maken
3	Het moet mogelijk zijn om tussen meerdere dashboards te wisselen
4	Het moet mogelijk zijn om een dashboard een naam te geven
5	Indien er op een grafiek geklikt wordt op het dashboard wordt deze vergroot en kan de gebruiker hier op inzoomen

Gesprek 3:

De input voor dit gesprek zijn de twee voorgaande gesprekken met de requirements die tot op heden zijn vastgesteld. Dit gesprek is gebruikt om de nieuw beschreven requirements te controleren en te prioriteren. Dit gesprek is gevoerd met Leon Batta. Op 6 september om 12:00 uur.

Ook zijn er een aantal nieuwe vragen opgesteld die kunnen nieuwe requirements kunnen opleveren.

Ik weet vanuit het intake gesprek dat het dashboard waarschijnlijk ontwikkelt zal gaan worden in C#. Zijn er verder nog punten waar het dashboard aan moet voldoen op dit gebied?

- Veelgebruikte programmeertaal gebruiken, dus inderdaad c#, zoals veel binnen LetsGrow met C# wordt ontwikkelt.
- Het dashboard zal ontwikkelt moeten worden volgens de laatste technieken, zodat het dashboard nog vele jaren mee kan. (Geen oude frameworks)
- Het dashboard moet ontworpen worden volgens design standaarden

Tijdens mijn intake is er ook gesproken over data uit externe bronnen, wat is hier de bedoeling?

- Klanten informatie kunnen tonen via twitter, het weerbericht laten zien, een tegel maken waarop de klant het laatste nieuws kan zien, bijvoorbeeld actuele veilingprijzen.

Zijn er verder nog belangrijke punten?

- Misschien is een app een mogelijkheid, dit zal later nog moeten blijken.

Functionele requirements

Nr.	Requirement
1	Het dashboard moet de mogelijkheid bieden om het weer te kunnen tonen
2	Het dashboard moet de mogelijkheid bieden om twitterberichten te kunnen tonen
3	Het dashboard moet de mogelijkheid bieden om informatie van de veiling te kunnen tonen
4	Het dashboard moet de mogelijkheid bieden om het laatste nieuws te kunnen tonen.

Niet-functionele requirements

Nr.	Requirement
5	Het dashboard moet ontwikkeld worden met een veelgebruikte programmeertaal
6	Het dashboard moet ontwikkeld worden met behulp van de laatste technieken
7	Het dashboard moet ontworpen zijn/worden volgens de design standaarden
8	Het dashboard moet uitgebracht worden in app vorm

Gesprek 4:

De input voor dit gesprek zijn de drie voorgaande gesprekken. In dit gesprek is de lijst met requirements tot nu toe voorgelegd aan Peter Henriks met het verzoek of dit ook voldoet aan zijn wensen.
Dit gesprek heeft een dag na gesprek 3 plaatsgevonden op 7 september om 13:00.

Peter heeft deze lijst doorgenomen en was het verder eens met wat er tot nu toe was besloten. Echter waren er nog twee requirements die hij mistte, namelijk een meter kunnen tonen op het dashboard (nieuwe functionaliteit) en de mogelijkheid om dashboards te delen.

Uit de feedback zijn de onderstaande requirements voortgekomen.

Functionele requirements

Nr.	Requirement
1	Het moet mogelijk zijn om een meter toe te voegen aan het dashboard
2	Het moet mogelijk zijn om een dashboard te delen met andere gebruikers

F. Bijlage Productbacklog

Week 1 & 2

- Opstellen plan van aanpak
- Achterhalen eerste requirements
    - Functionele requirements
    - Niet-functionele requirements
- Onderzoek huidige situatie

De eerste twee weken zijn gebruikt om een beeld te vormen van de huidige situatie om vervolgens het plan van aanpak op te stellen. Aangezien de eerste week toch voornamelijk wordt gebruikt voor kennismaking en het inwinnen van informatie e.d. waren deze taken voldoende voor de eerste twee weken.

Iteratie 1

- Onderzoek bestaande dashboard pakketten
    - Onderzoekplan
    - Onderzoekrapport
- Opstart functioneel ontwerp
- Opstart technisch ontwerp
- Onderzoekje aanlevering data uit externe bronnen

De eerste iteratie is besteed aan het onderzoek. Aangezien het onderzoek uit verschillende onderdelen bestaat en er goed ingelezen dient te worden voordat het onderzoek uitgevoerd kan worden, heb ik twee weken uitgetrokken voor alleen het onderzoeksplan en het onderzoek.

Iteratie 2

- Implementeren dashboard huidige situatie
    - TFS
    - Database
    - Autorisatie
    - Testomgeving
- Opstellen mastertestplan

Bij het implementeren van een pakket binnen een bestaande situatie zal je naar verwachting altijd tegen wat onvoorziene problemen aanlopen. Verder zal er ook nog verder uitgezocht moeten worden hoe de huidige situatie in elkaar steekt voordat bepaalde onderdelen geïmplementeerd kunnen worden. Ook is deze iteratie tijd gestoken in het inrichten van de werkomgeving. Dit vergt ook wat tijd.

Iteratie 3

- Selectie grafiek (definitie) uit teeltgroep
- Selectie rapport (definitie) uit teeltgroep
- Tonen grafiek op dashboard
- Tonen rapport op dashboard

De moeilijkheid in deze iteratie zal voor het selecteren van een grafiek of het selecteren van een rapport niet ver uit elkaar liggen. Indien een van de twee mogelijk is zal het naar verwachting niet moeilijk zijn om de andere ook werkend te krijgen. Ditzelfde geld voor het tonen van de grafiek of rapport op het dashboard. De eerste nieuwe dashboard tegel die aangemaakt moet worden zal wat tijd kosten, maar indien dit gelukt is zal het naar verwachting bij de overige tegels een stuk sneller gaan.

Hierdoor heb ik gekozen om deze twee functionaliteiten in deze sprint op te nemen.

Iteratie 4

- Onderzoekje bestaande meters
- Ontwikkelen meterapplicatie binnen LetsGrow
- Tonen meter op dashboard
- Tonen plattegrond op dashboard

De meter applicatie is een volledig nieuwe functionaliteit dat momenteel niet beschikbaar is binnen LetsGrow. Hierdoor zal hier het grootste deel van de beschikbare uren aan besteed worden. Zoals beschreven in iteratie 3 is er reed al een dashboard tegel gemaakt, dus het toevoegen van de meter en de plattegrond op het dashboard zal naar verwachting niet erg veel tijd kosten.

Iteratie 5

- Lay-out wijzigingen aanbrengen
    - Opstellen lijst benodigde wijzigingen dashboard
    - Doorvoeren benodigde wijzigingen dashboard

Om het dashboard goed te laten integreren binnen LetsGrow is het belangrijk dat deze de juiste lay-out krijgt. Tijdens de verschillende gesprekken en demo’s zijn er aantekeningen bijgehouden met opmerkingen over het dashboard. Die heb ik deze iteratie uitgewerkt en besproken, om op die manier een volledig lijst met wijzigingen op te kunnen stellen. Op basis van de aantekeningen kon ik in grove lijnen wel inschatten dat een iteratie van twee weken voldoende was om deze wijzigingen door te voeren

Iteratie 6

- Bepalen benodigde techniek voor communicatie tussen tegels.
- Ontwikkelen communicatie tussen tegels
    - Wijzigingen maken aan grafiek applicatie
    - Wijzigingen maken aan plattegrond applicatie
    - SignalR server ontwikkelen.

De nieuwe functionaliteit om tegels met elkaar te kunnen laten communiceren is volledig nieuw en dit maakt het ook lastig om een goede inschatting te maken qua tijd. Doordat dit nog vrij onzeker was en er ook nog uitgezocht moest worden welke techniek er voor gebruikt zou gaan worden, is er gekozen om hier de volledige twee weken aan te besteden.

G. Bijlage Aanlevering data externe bronnen

Weer

In dit hoofdstuk worden de mogelijkheden besproken voor het tonen van het weerbericht op het dashboard.

Mogelijkheden

Om het weer te tonen zijn er verschillende mogelijkheden. De eerste mogelijkheid is om data uit de meteo module van LetsGrow op te halen. Een tweede oplossing is het ophalen van weer data uit een externe bronnen.

Er zijn diverse partijen die weerdata aanleveren tegen verschillende kosten. Een aantal van deze partijen zijn:

- Yahoo (gratis)
- OpenWeatherMap (gratis/betaald)
- AccuWeather (betaald)
- Forecast.io (gratis/betaald)
- Weatherbug (betaald)

OpenWeatherMap en Forecast.io bieden hun dienst gratis aan tot een maximum van 60 calls per minuut bij OpenWeatherMap en 1.000 calls per dag bij Forecast.io. forecast $0.0001 per call ($1 per 10.000 requests)

De twee beste opties lijken hier Yahoo en OpenWeatherMap, aangezien Yahoo gratis is, OpenWeatherMap genoeg calls per dag kan aanbieden en deze diensten niet onder doen ten opzichte van de betaalde diensten.

OpenWeatherMap bied tegen betaling ook de mogelijkheid om het weer van de afgelopen tijd te bekijken. Hier heb je de optie voor data van 1 maand terug, 1 jaar terug en 5 jaar terug. Deze opties kosten $150, $950 en $3000.

OpenWeatherMap en Yahoo bieden beide een zeer uitgebreide uitleg over het gebruik van de API.
De data die OpenWeatherMap aanlevert aan de hand van de API request is in JSON formaat. Yahoo geeft de keus om JSON terug te krijgen of XML. Ook bied Yahoo de mogelijkheid om het weer via RSS op te halen.

Conclusie

Het weer zal in een later stadium specifiekere informatie moeten kunnen tonen, zoals de straling e.d., maar voor nu volstaat het normale weerbericht. Aangezien Yahoo gratis is en heeft erg veel locaties beschikbaar waar weerinformatie van opgevraagd kan worden voldoet deze het beste.

Twitter

In dit hoofdstuk worden de mogelijkheden besproken voor het tonen van twitterberichten op het dashboard.

Mogelijkheden

Twitter maakt sinds API 1.1 gebruik van oAuth. Dit is echter alleen nodig wanneer je ook wilt kunnen posten via de widget. Als het doel alleen is om feeds te tonen is het voldoende om een stukje html op je webpagina te plaatsen.

Als het voldoende is om alleen feeds te tonen kan er gebruik gemaakt worden van een widget script van twitter. https://publish.twitter.com/#

```
<a class="twitter-timeline" href="https://twitter.com/TwitterDev">Tweets by TwitterDev</a>
<script async src="//platform.twitter.com/widgets.js" charset="utf-8"></script>
```

Indien alleen het tonen van tweets niet voldoende is en de restricties van de officiële methode via oAuth (beperkt aantal calls) niet gewenst zijn, is het mogelijk om via 3th party scripts de twitter feeds op te halen. Het is echter wel belangrijk dat deze methodes up-to-date blijven. Om dit te kunnen realiseren wordt de html content van de twitter pagina opgehaald en wordt deze geparsed. Dit werkt goed, maar zodra twitter de pagina structuur wijzigt werken deze methodes niet meer. Twee mogelijkheden die gebruikt kunnen worden:
https://github.com/jasonmayes/Twitter-Post-Fetcher
https://github.com/robinbonnes/Custom-Twitter-PHP-API

Conclusie

Voor nu is het voldoende om alleen wat feeds te tonen zonder de mogelijkheid om direct te kunnen tweeten en dergelijke. Daardoor is het niet nodig om een token en key van twitter te gebruiken en hoeft er ook geen gebruik gemaakt te worden van een 3th party script.

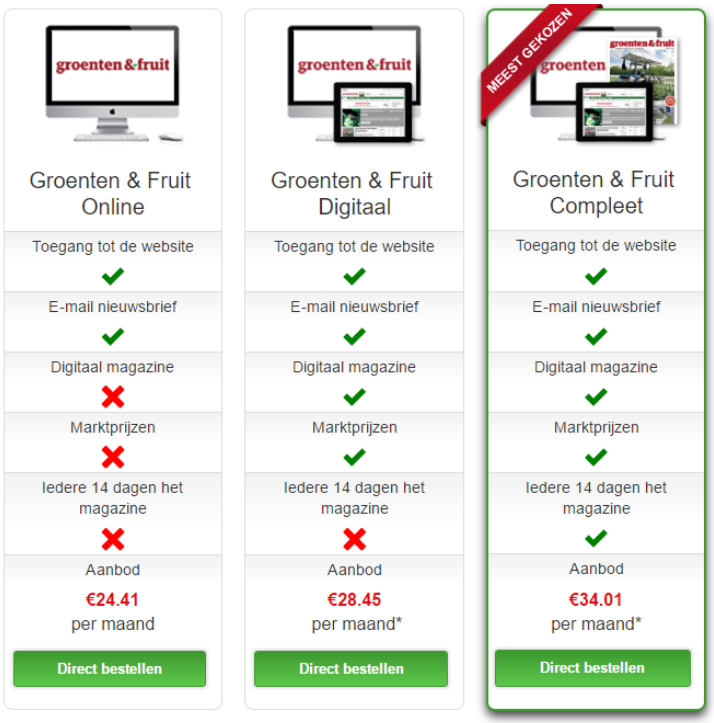
Veiling

In dit hoofdstuk worden de mogelijkheden besproken voor het tonen van data van de veiling

Mogelijkheden

Sinds een paar jaar heeft Royal Floraholland besloten zijn klokprijzen niet meer openbaar te maken met het oog op concurrentie. Voorheen had deze data gebruikt kunnen worden, maar deze mogelijkheid is er niet meer.

Het 2 wekelijkse blad Groente & Fruit bied tegen betaling nog wel de mogelijkheid om actuele groente en fruit prijzen te bekijken. Om toegang te krijgen tot deze informatie dient er een abonnement afgesloten te worden.



Zoals op onderstaande afbeelding te zien is kost dit abonnement minimaal €28,45 per maand.

Prijs informatie van groene planten en bloemen is helaas niet beschikbaar, dus deze mogelijkheid valt af.

Conclusie

Het is nog onbekend hoe de data daadwerkelijk opgehaald kan worden, maar bekend is dat Groente & fruit wel data aanbiedt. Hier zal mogelijk een tool voor geschreven moeten worden.

http://www.sierteeltnet.nl/bloemen-planten/nieuws/floraholland-stopt-met-info-klokprijzen/
http://www.gfactueel.nl/Abonnementen/

H. Bijlage Wijzigingen benodigd voor grafiek en rapport applicatie

Deze wijzingen aan de grafiek en overzicht applicatie zijn nodig om ze te optimaliseren voor het gebruik op het dashboard. Hieronder wordt per applicatie beschreven welke aanpassingen nodig zijn.

Grafiek:

- Bootstrap menu maken waarin de menu items zullen worden verborgen indien het scherm te klein wordt.
- LetsGrow logo verwijderen of verkleinen
- Optie in het bootstrap menu om grafiek in detail te kunnen bekijken in een nieuw scherm
- Legenda strippen. Alleen naam (+ cursor?) wanneer de grafiek te smal is, wel tonen wanneer de dashboard tegel groot wordt weergeven.
- Periode selecteren van grafiek verplaatsen (naar navigatiebar?)
- Titel grafiek verwijderen, wordt namelijk al getoond in de tegel titel.

Rapporten:

- Menu aan de linkerkant verbreden en of ombouwen naar een bootstrap menu
- Dropdown lijsten bovenin beter uitlijnen
- Error “Cannot read property ‘style’ of null” oplossen in zoomFactorPrint.js (deze error komt voor wanneer het rapport wordt geopend, zowel op het dashboard als via de normale manier)

I. Bijlage Onderzoek verschillende meters

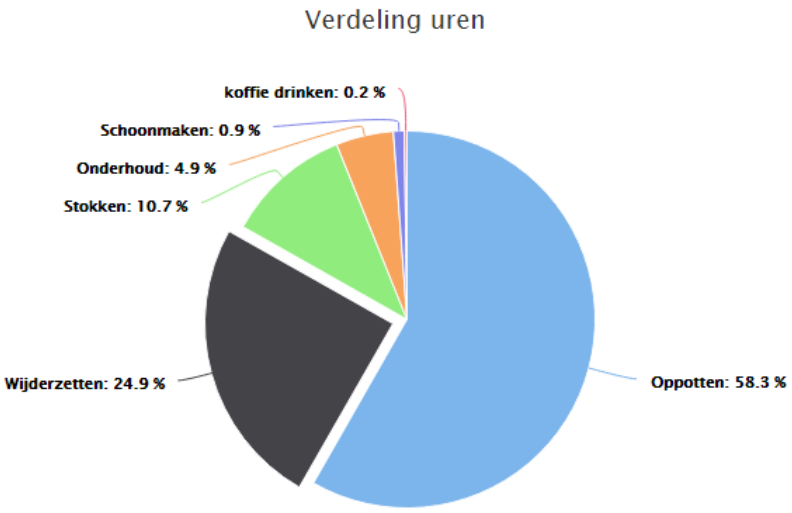
In dit document worden een aantal meters besproken met daarbij uitleg waar de meter eventueel voor gebruikt kan worden. Verder wordt hier ook bij vermeld welke data nodig is om de meter te kunnen tonen.

Pie chart

Benodigde data (Voor bijvoorbeeld arbeidsoverzicht):

- Metingen uit arbeidsregistratiemodule zoals
  - Oppotten
  - Wijderzetten
  - Stokken
  - Enz.

- Hoeveel uur in welke categorie
- Roos lengtes



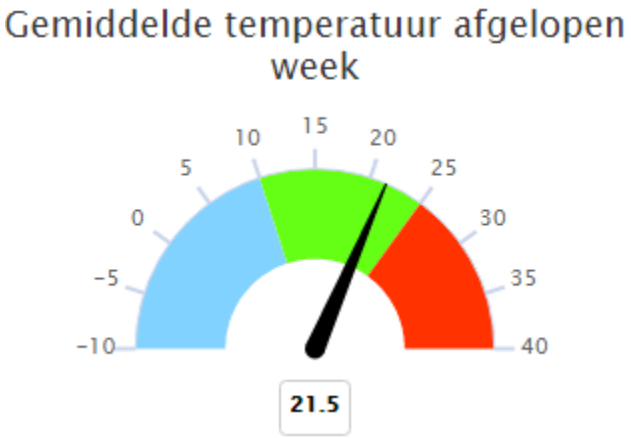
Gauge

Benodigde data:

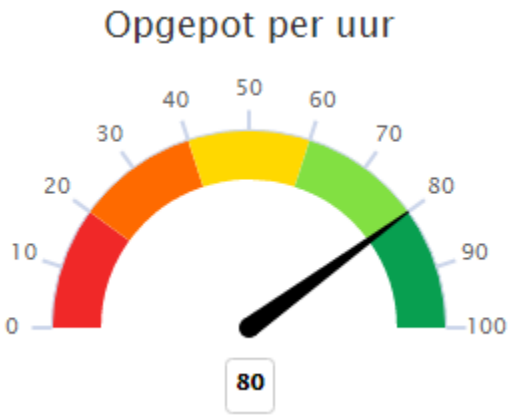
- Schaalverdeling
- Te tonen waarde
- Goede gebieden
- Foute gebieden
- Kleuren goede gebieden
- Kleuren foute gebieden

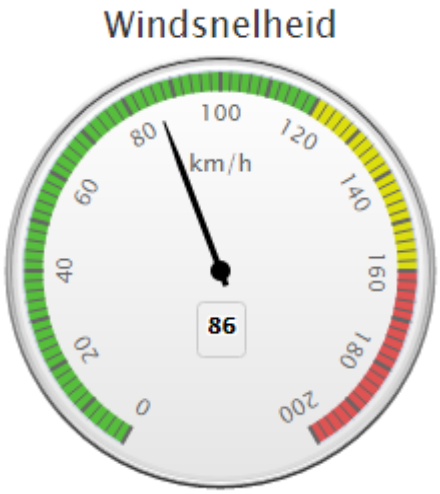
Gebruikt voor:

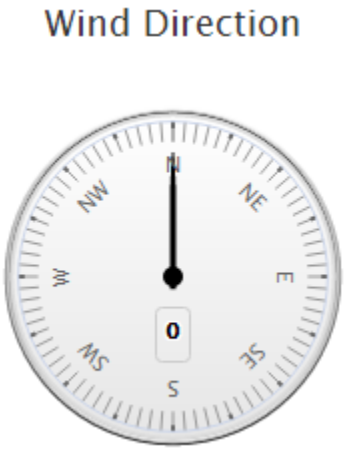
- Planten opgepot per uur
- Gemiddelde temperatuur over bepaalde periode
- Windsnelheid
- Windrichting
- Stroomproductie

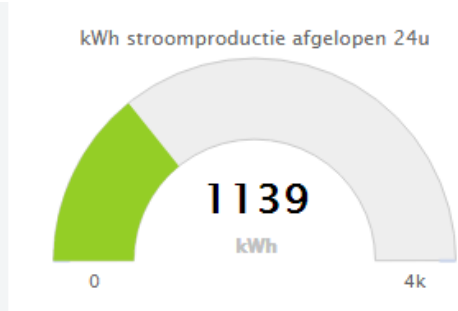


(Weergave meerdere kleuren mogelijk)









Thermometer meter:

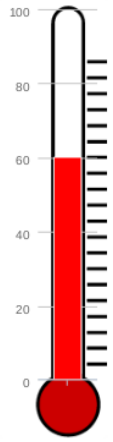
Benodigde data:

- Minimale temperatuur
- Maximale temperatuur

- Te tonen temperatuur

Gebruikt voor:

- Temperatuur buiten/ bepaalde afdeling



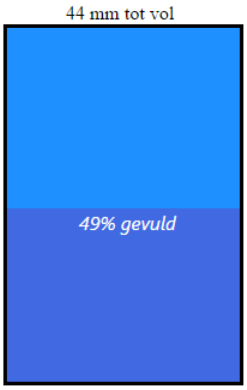
Vulling meting (Delfland)

Benodigde data:

- Actuele vulling bassin
- Totale inhoud bassin.
- Oppervlakte kas dek
- Berekening benodigde regenval tot vol

Gebruikt voor:

- Delfland project
- Vulling bassin / silo



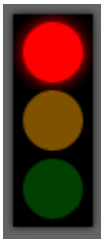
Status meter

Benodigde data:

- Waarde goed
- Waarde twijfel
- Waarde fout
- Te tonen waarde

Gebruik voor (Per item individueel stoplicht):

- Temperatuur afdeling 1 is goed,
- Temperatuur afdeling 2 vereist aandacht,
- Temperatuur afdeling 3 is niet goed



Feedback Leon Batta:

Hoi Ron,

Goed overzicht.

Vraagje bij thermometer: wat doe je met min en max waarde?

Ik zou dit verder verspreiden naar Peter, Joost en Chris en dan moeten we even beslissen waar we mee gaan beginnen.
Met vriendelijke groet,

Leon Batta

Feedback Joost Simons:

Ron

We hebben al het een en ander besproken maar dat waren meet oplossingen of mogelijkheden met de meters
zoals
Gebruik itemcode om titel en eenheid te vullen met goede suggestie

Ik zie geen status die kan wijzigen.
Bijvoorbeeld zon , zonnig, bewolkt en regen icoon al is hier geen gegeven voor waarop je dot eenvoudig kan baseren.

Vergeet in dit verhaal ook de visualisaties niet
Bv het meteo ding wat achter mij hangt kan met een standaard weergave alles bevatten.
Wat jij dan moet maken is visualisaties in de tegels.

Alle ronde vormen kun je natuurlijk als een ding samenvatten, ook de windvaan
Bij aanmaken geef je op hoeveel graden de schaal moet zijn
Min waarde en max waarde
Stap grootte of interval (ook nog iets verzinnen om de waardes langs de schaal te kunnen zetten)
Kleur stappen waarbij je altijd de max waarde invult, en dat is dan automatisch de min waarde van de volgende stap
Daarbij koppel je een kleur

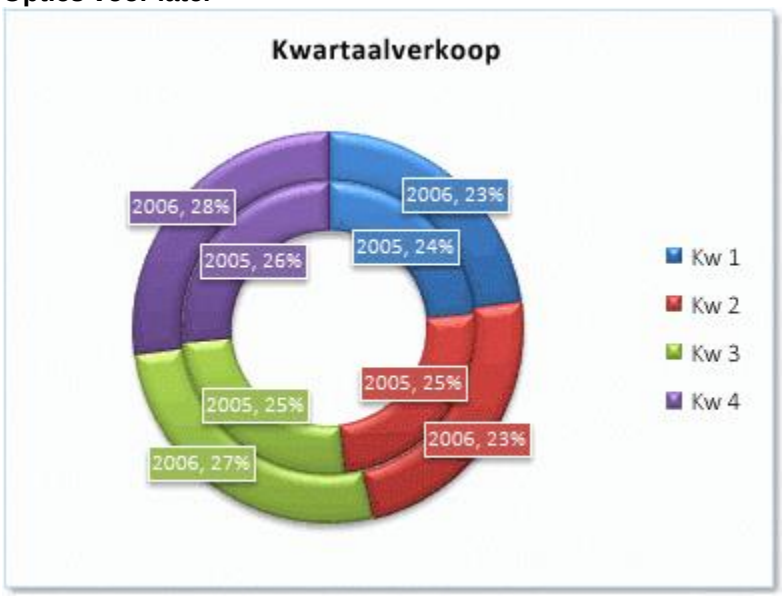
Ik zou naast een temp ook een niveau meter maken
Bv voor buffer vulling of rv. Die wil je niet als temp weergeven.
Bij beide zou ik de actuele waarde ook op de rechterzijde weergeven als getal

Ik denk dat het overal goed is om een titel te maken en eventueel een eenheid weer te geven

Het stoplicht is prima maar daar heb je standaard 3 gebieden en hoef je alleen de grenzen op te geven
Daarbij moet het ook mogelijk zijn om met een bolletje de status weer te geven.
Al zou dat ook weer met meer gebieden te maken zijn.

Voor een eerst versie
Cirkel
Temp
Niveau
Stoplicht

Opties voor later



Hiermee kun je direct zien hoe het in verschillende periode is geweest

Bijvoorbeeld arbeid deze maand en vorige maandag voeg diverse gegevens toe
Oppotten
wijderzetten
gewasverzorging
gewasbescherming
afleveren
geef periode grootte op, week, maand, 4 weken of jaar.
Daarmee is direct zichtbaar hoeveel er gewijzigd is t.o.v. de vorige periode.

Ik zou ook graag een indicatie willen zien of iets toe of afgenomen is, bv door een pijltje omhoog of omlaag.
Daarbij heb je wel een periode nodig en een soort drempel.

Wat is je doel met dit document.
Wil je het verzamelen en dan een nieuwe versie

Wil je dit bespreken? Plan dan direct een afspraak.
Dat maakt vaak veel helder
Dat geeft de termijn aan waarop men moet reageren.
Maar het geef voor mij ook aan of ik het kan toelichten of juiste heel duidelijk moet verwoorden.

Dat laatste is het nu misschien niet maar ik verwacht dat jij er wel mee verder kunt.
Met vriendelijke groet,

Joost Simons

J. Bijlage Wijzigingen met betrekking tot dashboard stijl

Lay-out wijzigingen benodigd voor dashboard:

- Zwarte thema balk verwijderen
- Dashboard titel in navigatiebalk plaatsen
- Navigatiebalk LetsGrow groen maken
- Achtergrond dashboard verwijderen
- Achtergrondkleur kiezen
- Dashboard vertalen met resources
- Automatische bewerkmodus uitschakelen
- Tegelselectie vak smaller maken
- Ruimte tussen tegels verkleinen (ruimte zo goed mogelijk benutten)
- Button kleuren wijzigen
- Tekstkleuren (navigatiebalk) wijzigen / knoppen
- Bootstrap switch wijzigen
- Navigatiebalk (bootstrap balk) smaller
- Scherm indien geen dashboard beschikbaar
- Balk met alle dashboard tegels verkleinen
- Optie om tegel te openen in nieuw scherm
- Vertalen dashboard tegel titels
- Navigatie dashboard tegel bewerk pagina intuïtiever
- Plattegrond tegel schaalbaar maken

Deze wijzigingen worden gemaakt om het dashboard de LetsGrow huisstijl te geven en zo beter overeen komt met de rest van de applicaties.

Overige wijzigingen:

- Selectie uit modules voor grafieken en overzichten

Wijzigingen bedacht en besproken met Leon

- **Iconen dashboard tegels vervangen door LetsGrow iconen**
  Voor de iconen die niet beschikbaar waren binnen LetsGrow of voor de iconen die niet bruikbaar waren, is er gezocht naar nieuwe iconen. Hierbij moest rekening gehouden worden met de copyright rechten, zodat hier in de toekomst geen problemen over kunnen ontstaan. Na de voorwaarden te hebben gelezen op de website waar de iconen van afgehaald zijn, zijn de nieuwe iconen opgenomen in het dashboard.

- **Positie hamburgermenu dashboard controleren**
  Zodra het dashboard op een mobiel apparaat werd bekeken werd het hamburger menuutje aan de rechter kant getoond. Echter werd het hamburger menuutje in de grafiek applicatie aan de linker kant getoond. Om te zorgen voor een eenduidige werking is er gekozen om het menu icoon in beide gevallen aan de linker kant te plaatsen, zodat hier geen misverstanden over kunnen ontstaan.

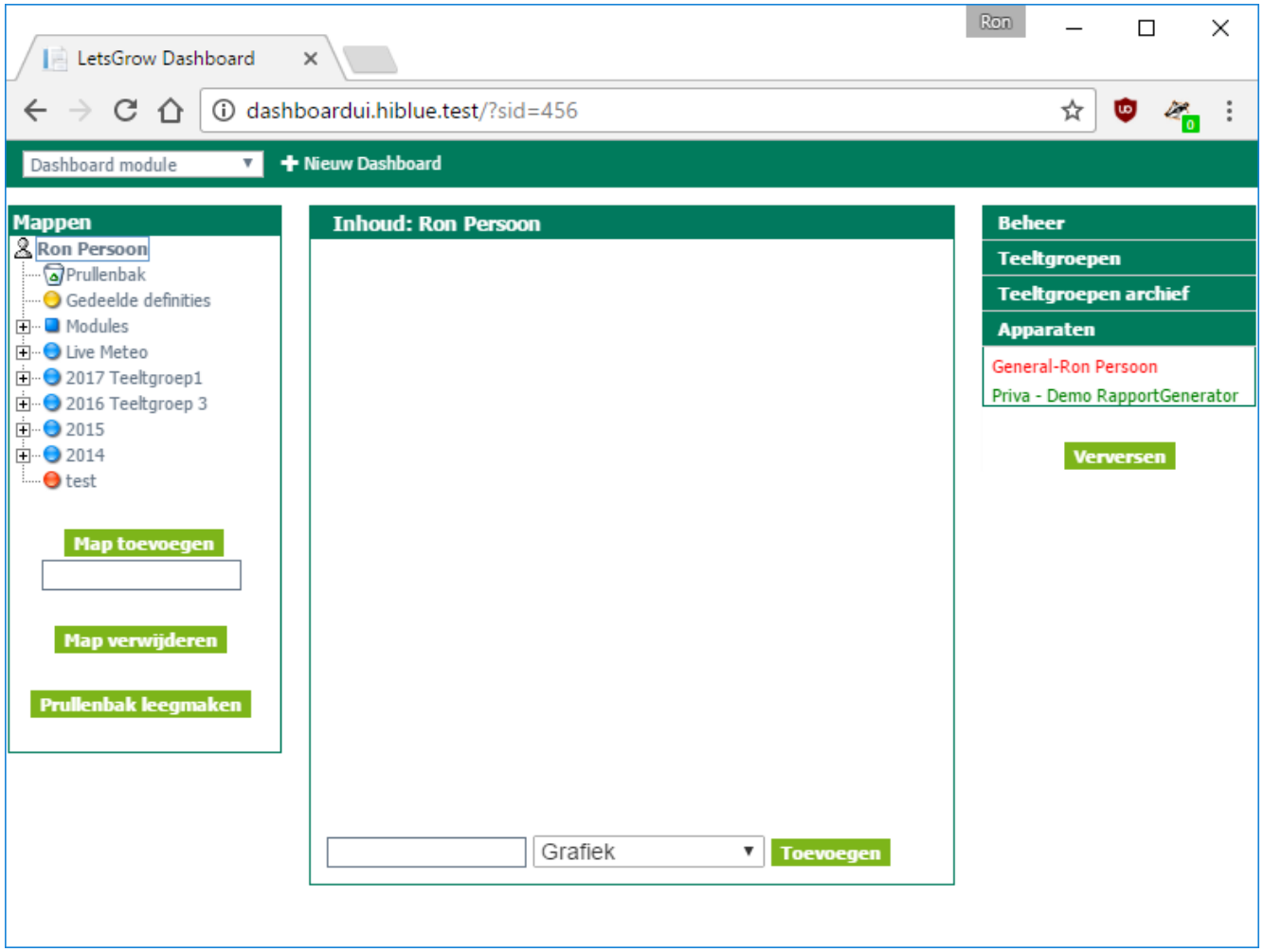
- **Dashboard selectie lijst i.p.v. knop**
  LetsGrow maakt op diverse plekken gebruik van selectielijsten. Het dashboard pakket bood echter een alternatieve methode om een dashboard te selecteren, namelijk in de vorm van een soort waaier. Aangezien deze methode wat beperkingen met zich mee bracht, vooral wanneer de klant een groot aantal dashboards heeft, is ook hier gekozen om het dashboard keuze scherm te vervangen door een keuzelijst. Op deze manier kan de klant eenvoudig wisselen tussen dashboards, is er geen limiet van het aantal dashboards meer en kan aan de huidige keuze van de keuzelijst direct gezien worden welk dashboard er op dat moment geselecteerd is.

- **Dashboard configuratie verbergen bij opstarten**
  Het dashboard startte automatisch op in de bewerk modus. Aangezien dit direct erg veel ruimte van het dashboard inneemt, is er gekozen om het dashboard op te laten starten in toonmodus, zodat de ruimte optimaal gebruikt kan worden om de informatie op van het dashboard te bekijken.

- **Bedenken implementatie oud klantmenu in dashboard**
  Voor een goede overgang van het klantmenu naar het nieuwe dashboard moest een manier bedacht worden, zodat klanten rustig kunnen wennen aan de nieuwe situatie en niet direct verplicht worden om het dashboard te gebruiken. Er is gekozen om het klantmenu in een iframe op de beginpagina van het dashboard te tonen. Hierdoor kan de klant LetsGrow op de huidige manier blijven gebruiken indien gewenst, maar is het ook erg eenvoudig om over te schakelen naar een dashboard via de keuzelijst linksboven in het scherm.

K. Bijlage Werking dashboard applicatie in afbeeldingen

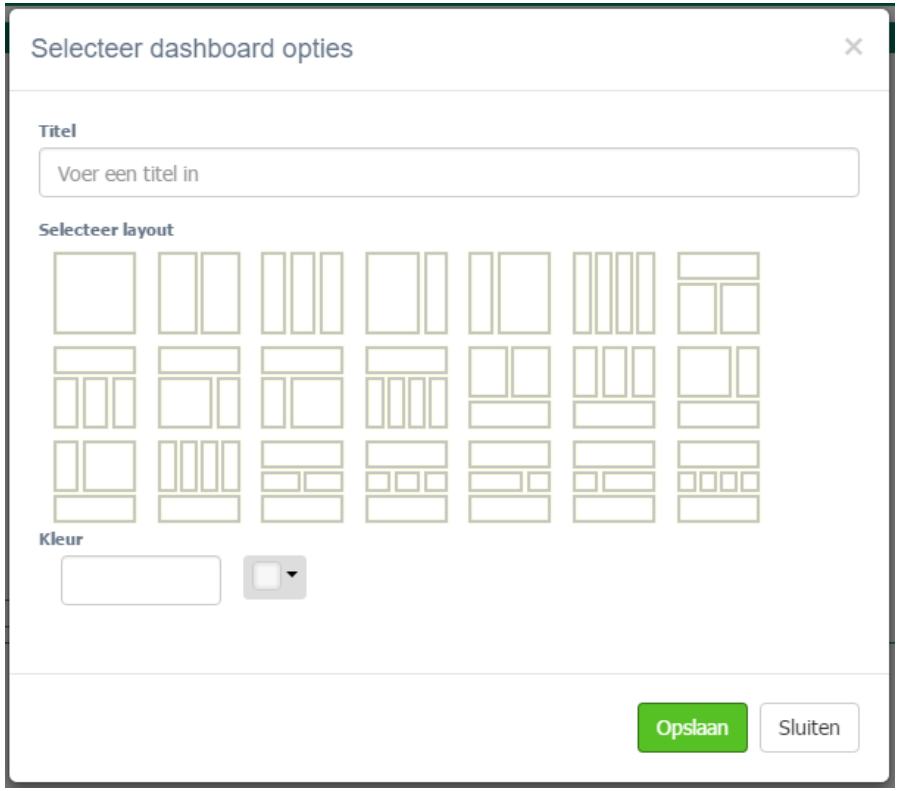
Doordat het dashboard van mijn kant gereed is, maar nog door Joost aan een acceptatietest moet worden onderworpen, zal het dashboard na de afstudeerperiode nog niet direct op de productieomgeving worden geplaatst. Hierdoor heb ik besloten, om met behulp van afbeeldingen, de flow en werking van het dashboard te beschrijven.



Afbeelding 1: Startpagina dashboard applicatie

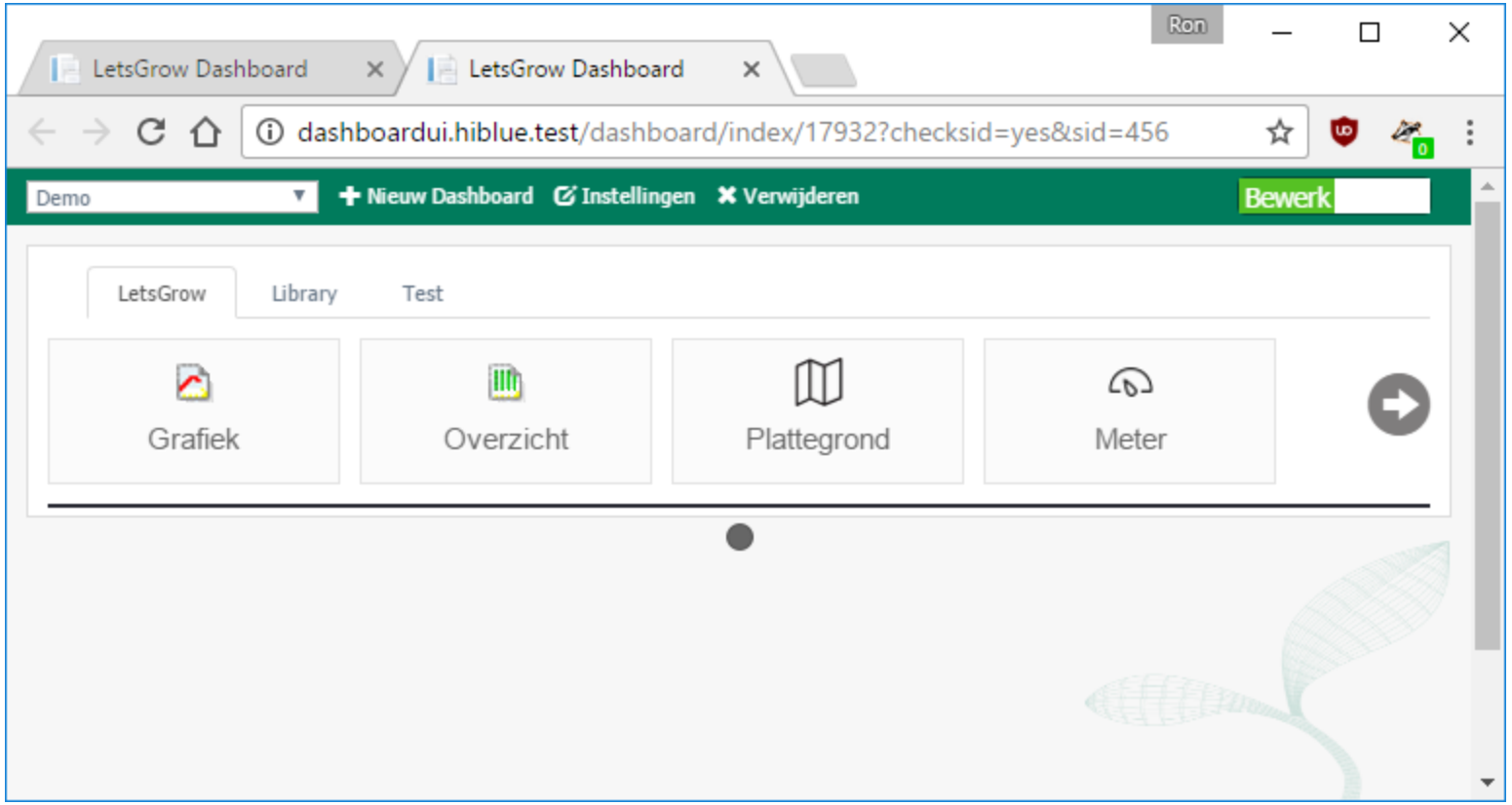
Bij het openen van het dashboard zal de gebruiker voor nu geen grote veranderingen zien. De enigste verandering is de groene balk die boven het klantmenu getoond wordt.

Zodra de gebruiker bovenin op “Nieuw dashboard” klikt verschijnt het scherm om een nieuw dashboard te configureren.



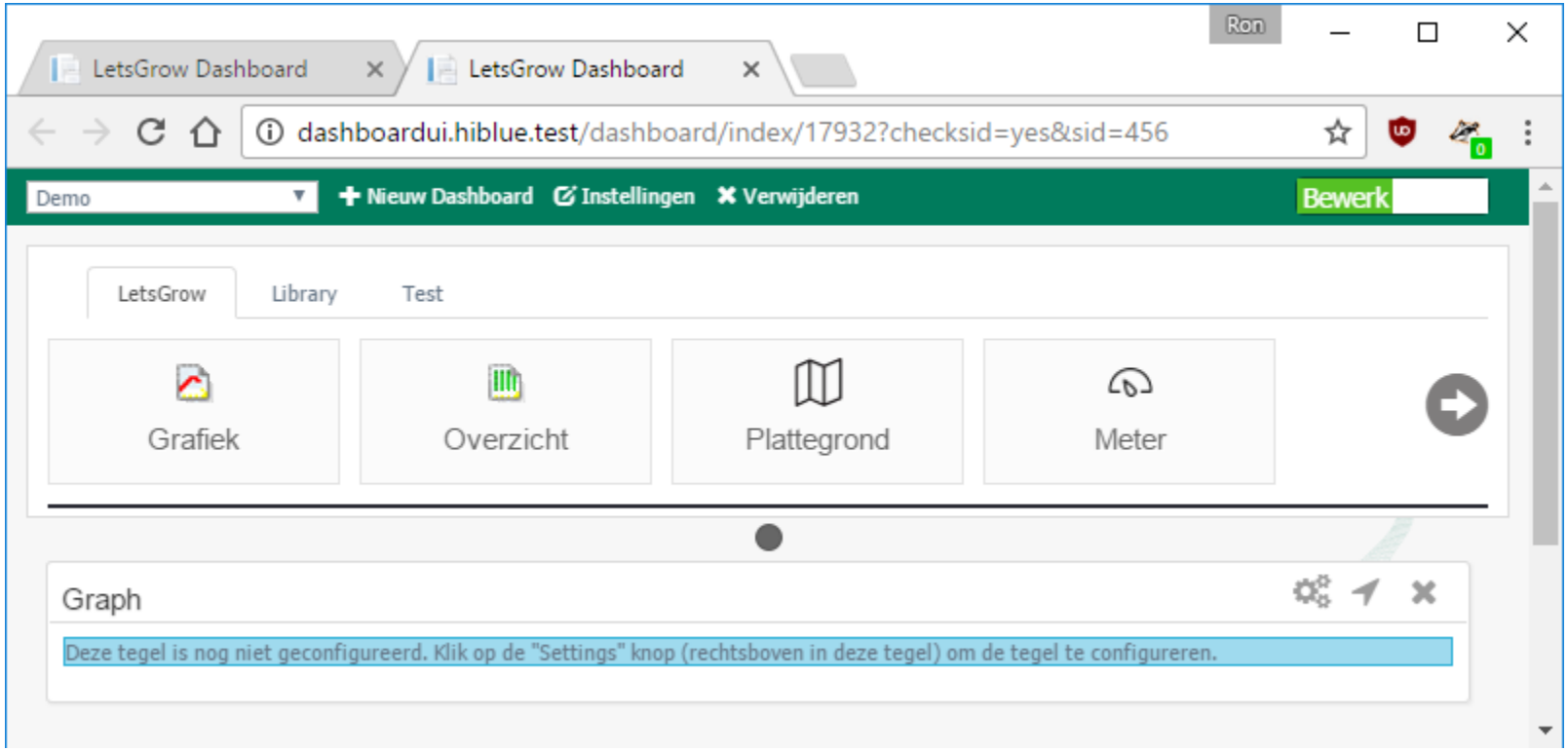
Afbeelding 2: Dashboard configureren

Nadat de titel is ingevoerd, er een lay-out is gekozen en er een kleur is gekozen klikt de gebruiker op opslaan. Er wordt nu een nieuw dashboard geopend in een nieuw tabblad. Het dashboard zal nu geopend worden in bewerk modus, zodat de gebruiker het dashboard direct kan configureren.



Afbeelding 3: Nieuw dashboard geopend

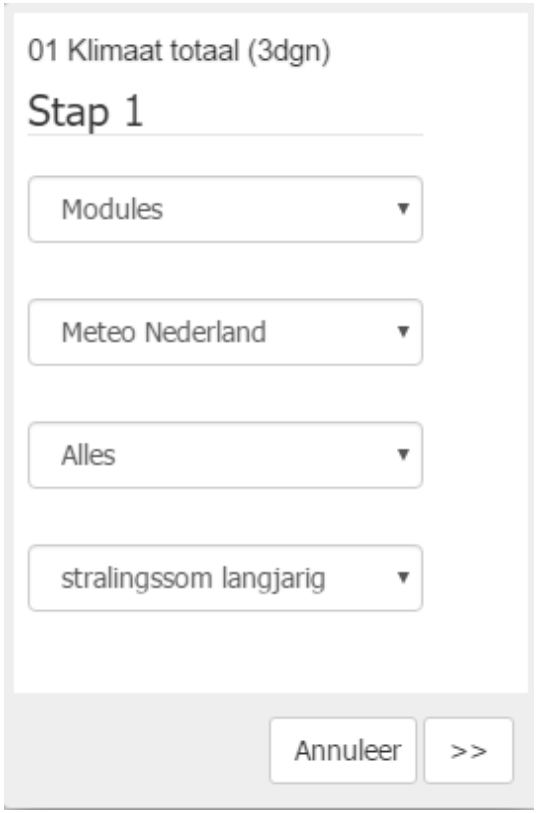
De gebruiker heeft nu de mogelijkheid om een nieuwe tegel toe te voegen aan het dashboard. Dit kan door bovenin dubbel op een van de tegels te klikken of een nieuwe tegel naar het dashboard slepen.



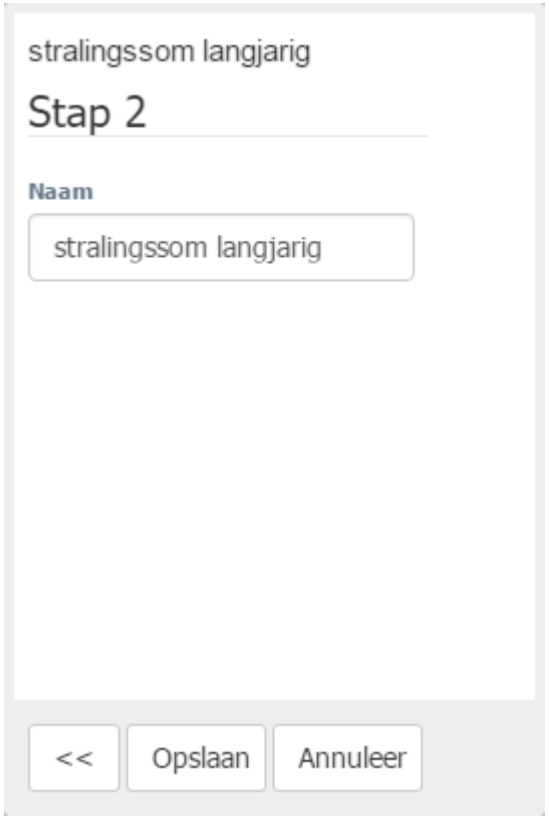
Afbeelding 4: Nieuwe dashboard tegel op het dashboard geplaatst

De nieuwe grafiek tegel wordt op het dashboard getoond en de gebruiker heeft nu de mogelijkheid om deze te configureren. Zodra de gebruiker op de “Settings” knop drukt, zal het configuratiescherm van de grafiek tegel worden geopend.

De gebruiker heeft vervolgens de mogelijkheid om uit de gewenste categorie een grafiek te selecteren. Hiervoor dient de gebruiker twee stappen te doorlopen. Stap 1 wordt gebruikt voor de selectie van de grafiek en in stap 2 wordt de naam van de grafiektegel gekozen. Standaard wordt hier de naam van de gekozen grafiek ingevuld.



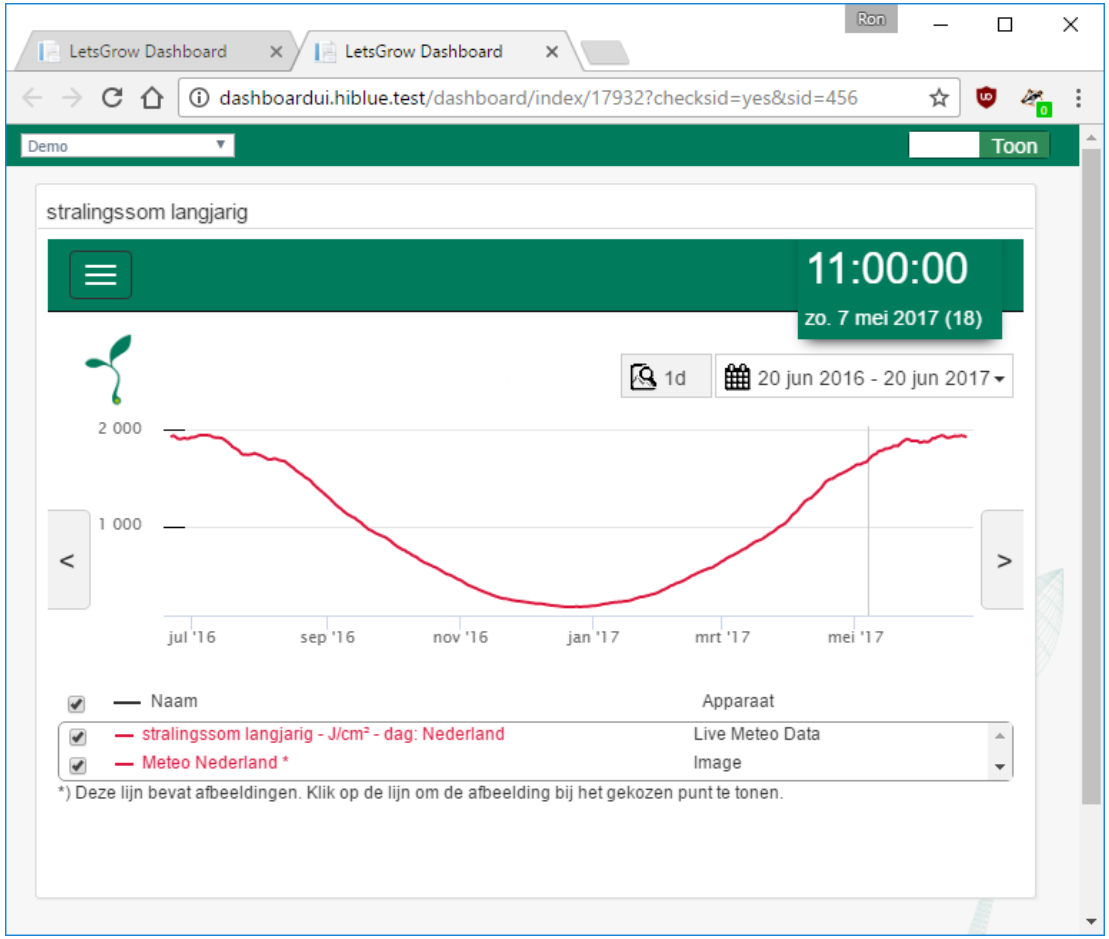
Afbeelding 5: Grafiek selecteren stap 1



Afbeelding 6: Grafiek selecteren stap 2

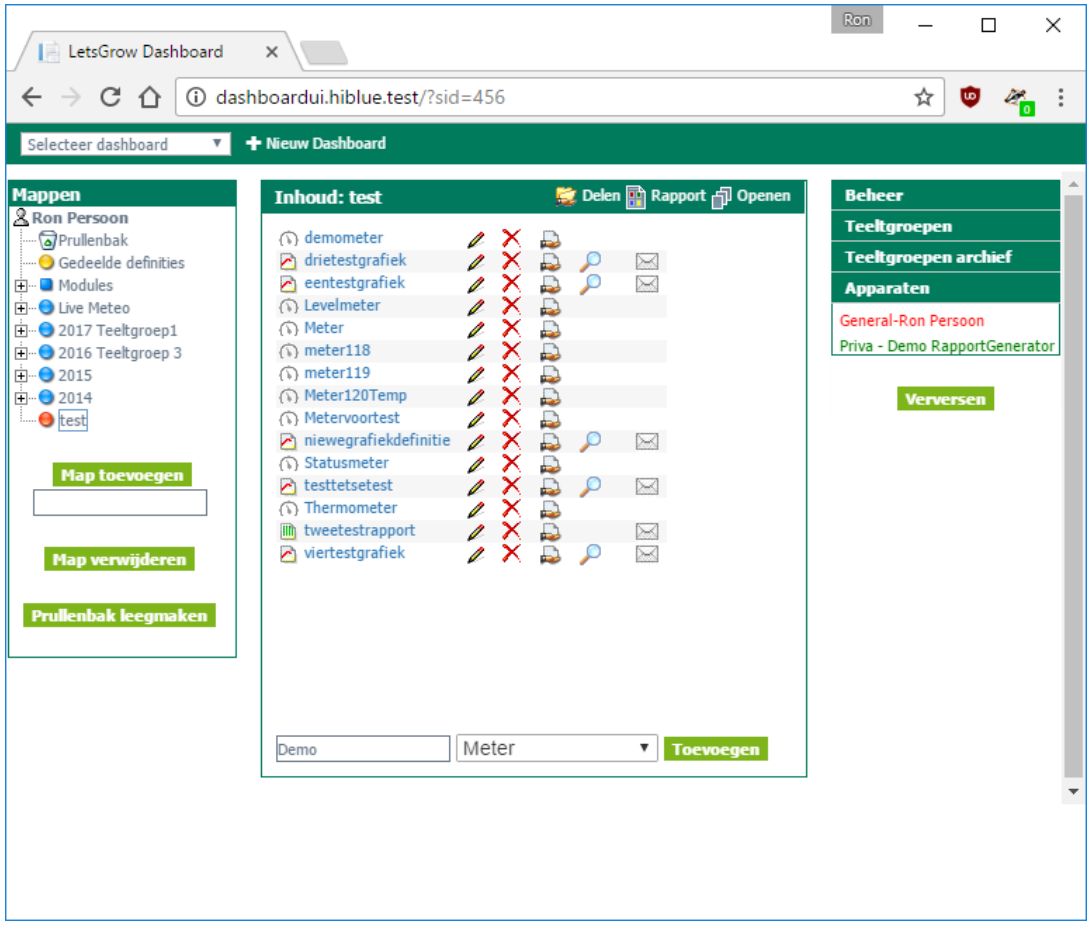
Als de gebruiker klaar is met het configureren van de grafiek klikt de gebruiker op “Opslaan”.

De gebruiker krijgt nu de geconfigureerde grafiek op het dashboard te zien.



Afbeelding 7: Grafiek geconfigureerd

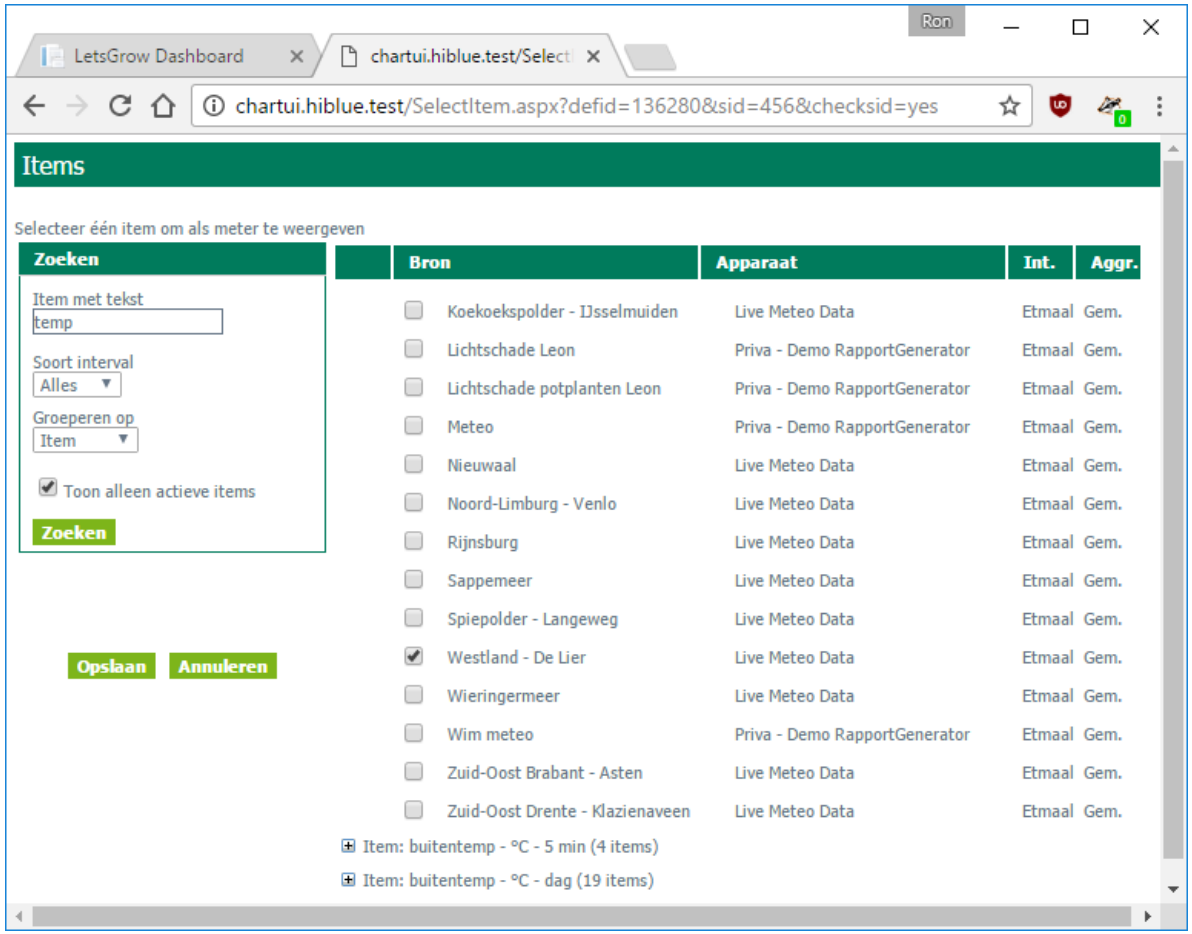
Eenzelfde procedure wordt doorlopen wanneer een gebruiker een plattegrond, rapport of meter toe wil voegen aan het dashboard. Voor er echter een nieuwe meter op het dashboard toegevoegd kan worden zal deze eerst via het klantmenu aangemaakt moeten worden. De gebruiker kan terugkeren naar het klantmenu door linksboven in de keuzelijst te kiezen voor “Klantmenu”.



Afbeelding 8: Meter aanmaken

In het voorbeeld uit afbeelding 8 is er gekozen om een nieuwe meter toe te voegen aan de “Overige definities” De gebruiker dient hiervoor onderin de naam van de meter in te voeren en te kiezen voor “Meter” uit de keuzelijst. Op het moment dat de gebruiker op opslaan heeft geklikt, wordt de meter toegevoegd aan de lijst met definities. De meters zijn te herkennen aan het meter icoontje dat ervoor staat.

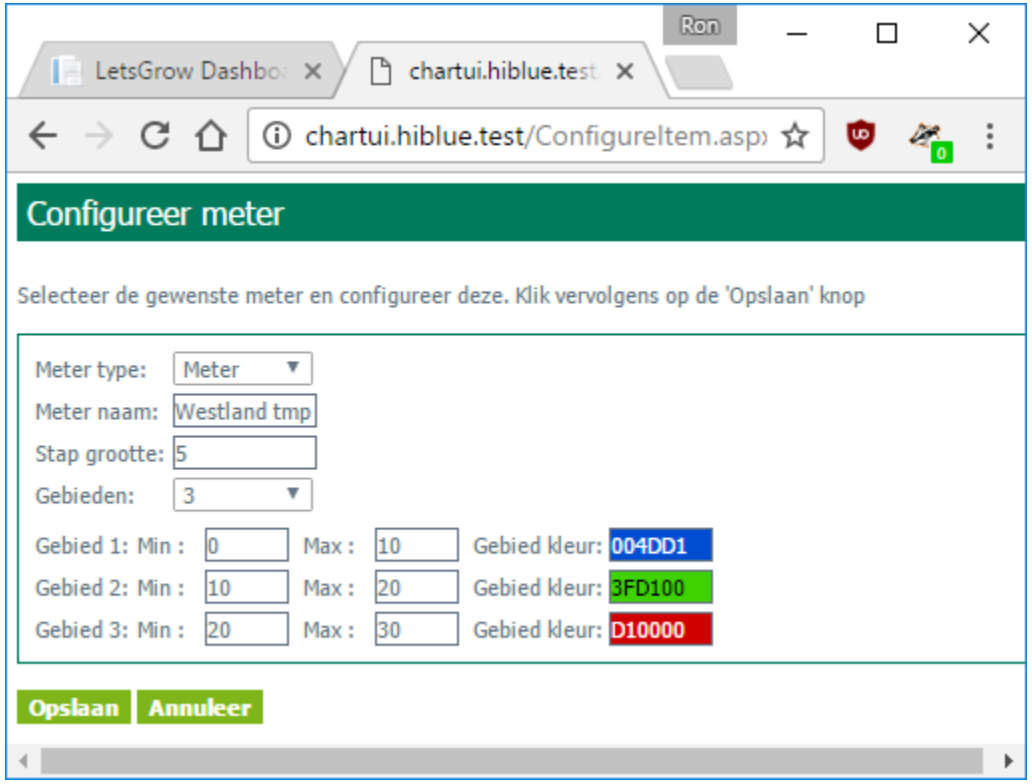
Zodra de gebruiker de nieuwe meter aanklikt, is er nog geen meter geconfigureerd. Hierdoor zal het configuratiescherm voor de meter worden getoond. In dit scherm wordt de gebruiker gevraagd om een meting te selecteren die in de meter getoond gaat worden. In het voorbeeld uit afbeelding 9 is er gekozen voor een temperatuur meting uit gemeente Westland.



Afbeelding 9: Meting selecteren

De gebruiker klikt vervolgens op “Opslaan” om naar het volgende configuratiescherm te gaan. In dit scherm kan de gebruiker het type meter kiezen en deze volledig naar wens inrichten.

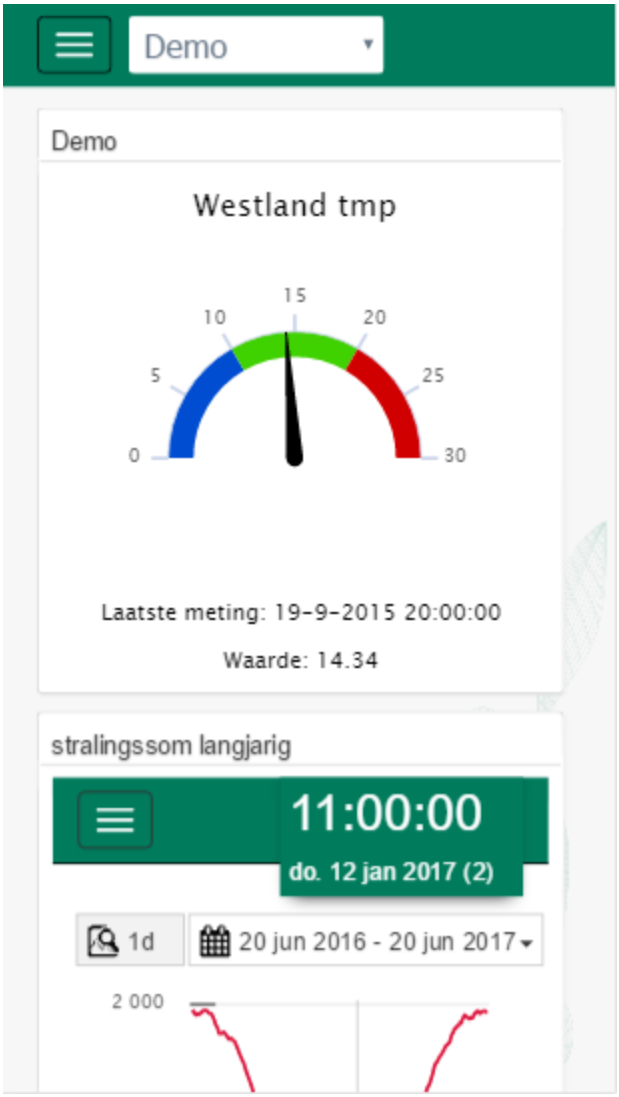
Zoals te zien is in afbeelding 10 heeft de gebruiker gekozen voor het type Meter. Deze meter kan geheel naar wens geconfigureerd worden. De gebruiker heeft hier gekozen voor 3 gebieden die koud, gemiddeld en warm representeren. Er is gekozen voor een “Stap grootte” van 5. Dit houdt in dat de range tussen het laagste getal (0 in dit voorbeeld) en het hoogste getal (30 in dit voorbeeld) wordt weergeven in stappen van 5.



Afbeelding 10: Meter configureren

Zodra de meter naar wens is geconfigureerd klikt de gebruiker op opslaan en keert de gebruiker terug naar de eerste pagina van het dashboard, namelijk het klantmenu. Nu kan deze meter ook worden toegevoegd op het dashboard. Dit zal volgens dezelfde procedure gebeuren als bij de grafiek die hierboven is beschreven.

Het resultaat van de geconfigureerde meter wordt in afbeelding 11 weergeven binnen een dashboard tegel. Op afbeelding 11 wordt het dashboard echter getoond op een mobiel apparaat. Hier is te zien dat de grafiek en het dashboard nu geoptimaliseerd zijn voor het gebruik op een mobiel apparaat.



Afbeelding 11: Geconfigureerde meter en mobiel weergave

Met deze schermafbeeldingen hoop ik voldoende duidelijk gemaakt te hebben hoe de dashboard applicatie in elkaar zit. Volgens de planning zal het dashboard uiterlijk eind januari online gaan, zodat deze gepresenteerd kan worden op de tuinbouw relatiedagen.

L. Bijlage Afstudeerplan

Afstudeerplan

Informatie afstudeerder en gastbedrijf (*structuur niet wijzigen*)

Afstudeerblok: 2016-2.1 (start uiterlijk 29 augustus 2016)
Startdatum uitvoering afstudeeropdracht: 29 augustus 2016
Inleverdatum afstudeerdossier volgens jaarrooster: 22 december 2016

Studentnummer: 13034472
Achternaam: dhr Persoon (*) *weghalen niet van toepassing*
Voorletters: R.A.H
Roepnaam: Ron
Adres: -
Postcode: -
Woonplaats: -
Telefoonnummer: -
Mobiel nummer: -
Privé emailadres: ron_persoon@hotmail.com

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

Naam studieloopbaanbegeleider: Mevrouw Wieman
Naam begeleidend examinator: Mevrouw A.M.J.J Lousberg
Naam tweede examinator: Mevrouw J.J. van der Hoek

Naam bedrijf: Lets-Grow.com B.V.
Afdeling bedrijf: Software development
Bezoekadres bedrijf: Westlandseweg 190
Postcode bezoekadres: 3131 HX
Postbusnummer: Postbus 180
Postcode postbusnummer: 3130 AC
Plaats: Vlaardingen
Telefoon bedrijf: 010 460 81 08
Telefax bedrijf: 010 460 80 00
Internetsite bedrijf: www.letsgrow.com

Achternaam opdrachtgever: dhr Hendriks (*) *weghalen niet van toepassing*
Voorletters opdrachtgever: P
Titulatuur opdrachtgever: Bsc
Functie opdrachtgever: Manager
Doorkiesnummer opdrachtgever: -
Email opdrachtgever: phe@letsgrow.com

Achternaam bedrijfsmentor: dhr Batta (*) *weghalen niet van toepassing*
Voorletters bedrijfsmentor: L
Titulatuur bedrijfsmentor: Ir
Functie bedrijfsmentor: Systeembeheer & Software development
Doorkiesnummer bedrijfsmentor:
Email bedrijfsmentor: lba@letsgrow.com

Opdrachtomschrijving *(toelichtende tekst verwijderen)*

1. Bedrijf

LetsGrow is een bedrijf dat meetgegevens registreert van meerdere tuinbouw bedrijven. Tuinbouw bedrijven kunnen bij LetsGrow een abonnement afsluiten en door middel van het systeem van LetsGrow meetgegevens inzien van andere locaties of collega tuinders.

De abonnementen worden in de vorm van modules aangeboden waarvoor de klant jaarlijks betaalt. Dit geeft de klant de mogelijkheid om zelf te bepalen welke modules wel of niet nodig zijn, zodat niet voor ongebruikte functionaliteit hoeft worden betaald.

Verder biedt LetsGrow de mogelijkheid om klanten hun data te laten delen. Hier zijn geen kosten aan verbonden en de klanten kunnen zelf bepalen met wie zij de data delen.

2. Probleemstelling

Wanneer klanten op dit moment inloggen bij LetsGrow, krijgen ze een pagina met zeer beperkte informatie te zien. Dit is momenteel niet erg klantvriendelijk in het gebruik. Het probleem is dat op deze eerste pagina nog geen meetgegevens en dergelijke worden getoond. Ook werkt het huidige systeem niet goed op mobiele apparaten.

Bovendien is het nu niet mogelijk om data uit externe bronnen te tonen, zoals twitter, het weer en eventueel de veiling.

3. Doelstelling van de afstudeeropdracht

Het ontwikkelen van een dashboard dat aan klanten getoond wordt wanneer zij inloggen op LetsGrow. Het dashboard zal responsive moeten worden, zodat deze ook gebruikt kan worden op mobiele apparaten.
Verder zal er de mogelijkheid geboden worden om data uit externe bronnen te tonen, zoals twitter, het weer en eventueel de veiling.

4. Resultaat

Er dient een dashboard te worden ontwikkeld of gekozen (met behulp van pakketselectie) waar klanten na het inloggen de belangrijkste meetgegevens kunnen zien. Ook moet er duidelijk te zien zijn waar de klanten naartoe kunnen navigeren om zo meer informatie te kunnen zien. De data wordt weergeven in verschillende stijlen, zoals overzichten (rapporten en tabellen), metertjes en foto’s.

Ook zal er de mogelijkheid zijn om data te tonen uit externe bronnen, zoals twitter, het weer en eventueel de veiling.

Het dashboard zal responsive zijn, zodat deze ook op mobiele apparaten te gebruiken is. Verder kan het dashboard per klant verschillen in verband met de verschillende modules die de klanten gebruiken.

De grafische kant van het dashboard zal, afhankelijk van de uitkomst van de pakketselectie, ook ontworpen worden. Indien er een bestaand dashboard geïmplementeerd kan worden zal hier hoogstwaarschijnlijk het ontwerp ook van aangepast worden.

De waarde voor het bedrijf is dat klanten minder vragen hoeven te stellen over de werking van LetsGrow, dat de data overzichtelijk is in te zien en dat dit alles professioneler oogt richting de klant.

Waarde is ook dat eerste navigatie intuïtiever zal worden en meer informatie laat zien in één oogopslag en wat eigentijdser oogt.

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

- Plan van aanpak 3 dagen
- Vaststellen knelpunten huidige situatie 2 dagen
- Vaststellen requirements 3 dagen
- Onderzoek werking huidige softwarepakket 2 dagen
- Onderzoek naar bestaande dashboard pakketten 5 dagen
- Onderzoek aanlevering data externe bronnen (twitter, weer veiling) 2 dagen
- Gesprekken binnen het bedrijf (requirements en de gewenste UI achterhalen, dit gesprek zal gevoerd worden met dhr Hendriks of dhr Batta) 2 dagen
- Ontwerpen mobiel dashboard 15 dagen
- Ontwerpen pc dashboard 5 dagen
- Ontwikkelen dashboard mobiel 20 dagen
- Ontwikkelen dashboard pc 5 dagen
- Testen dashboard mobiel 4 dagen
- Testen dashboard pc 2 dagen

Er zal binnen het bedrijf gewerkt worden met een variant op scrum. Er zal hoofdzakelijk met de programma's Visual studio 2015 en SQL Server gewerkt worden.
Voor het ontwikkelen van het dashboard zal er gebruik gemaakt worden van HTML, XML, ASP.NET C# en jQuery. LetsGrow maakt gebruik van een Relationele database om alle meetgegevens in op te slaan.

Het dashboard zal ontwikkeld worden volgens het mobile first principe. Dit houdt in dat het dashboard eerst ontwikkeld zal worden voor mobiele apparaten en vervolgens worden uitgewerkt, zodat deze ook goed zal werken op de pc.

6. Op te leveren (tussen)producten

- Plan van aanpak
- Volledige lijst functionele en niet functionele requirements
- Resultaat onderzoek “technology selectie of zelf ontwikkelen”
- Ontwerp mobiel
- Ontwerp pc
- Dashboard mobiel
- Dashboard pc
- Test-set en testbevindingen

7. Te demonstreren competenties en wijze waarop

1.3 Selecteren standaard Software Niveau 3
- Onderzoek naar bestaand dashboard

1.4 Uitvoeren analyse door definitie van requirements Niveau 3
- Met behulp van gesprekken met dhr Hendriks en of dhr Batta de requirements duidelijk krijgen en beschrijven

3.2 Ontwerpen systeemdeel Niveau 3
- Ontwerpen dashboard, voor pc en mobiel

3.3 Bouwen applicatie Niveau 4
- Afhankelijk van uitkomst pakketselectie het
- ontwikkelen van het dashboard / aanpassen van het dashboard pakket

M. Bijlage Beoordelingsformulier

DE HAAGSE HOGESCHOOL

Faculteit IT & Design

Delft

Den Haag

Zoetermeer

Evaluatieformulier afstuderen

In te vullen door opdrachtgever c.q. bedrijfsmentor(en)

Student: Ron Persoon

Periode: 31 augustus tot 22 december 2016

Bedrijf c.q. instelling: LetsGrow.com

Bedrijfsmentor: Leon Batta

Plaats: Vlaardingen

Datum: 21 december 2016

1. **Heeft de student zich zelf snel en goed ingewerkt in het bedrijf en de uit te voeren afstudeeropdracht?**

Ja

2. **Hoe beoordeelt u de communicatieve vaardigheden van de student (in de samenwerking met collega's, in contacten met de opdrachtgever, bij mondelinge presentaties, schriftelijke rapportages)?**

Goed

3. Hoe heeft de student tijdens het uitvoeren van de opdracht gefunctioneerd?

- **Qua verantwoordelijkheid** goed
- **Qua zelfstandigheid** voldoende
- **Qua planmatig werken** goed
- **Qua creativiteit** voldoende
- **Qua productiviteit** goed
- **Qua samenwerken met collega's** goed
- **Qua draagvlakontwikkeling** voldoende
- **Qua inspelen op bedrijfscultuur** goed
- **Qua rekening houden met de specifieke context van het bedrijf** goed
- **Qua het op gang brengen van de nodige veranderingen** voldoende

4. Hoe beoordeelt u de kennis en kunde van de student in verhouding tot wat u verwacht van een bijna afgestudeerde?

Goed

5. Hoe beoordeelt u de kwaliteit van de opgeleverde (tussen)producten?

Goed

6. **Bent u tevreden over het opgeleverde (eind)product?**

- **In hoeverre heeft u gekregen wat is afgesproken?**
  Naar aanleiding van ontwikkelingen op de markt is de afspraak bijgesteld. Die afspraak is ruim nagekomen.

- **In hoeverre voldoet het (eind)product aan uw verwachtingen?**
  goed

- **Wat is de bruikbaarheid en onderhoudbaarheid hiervan?**
  goed

- **Wat gebeurt er met het opgeleverde (eind)product?**
  Naar verwachting zal dit eind januari voor al onze klanten beschikbaar komen

- **Kunt u direct met het opgeleverde product aan de slag?**
  Voor een beperkte groep klanten wel.

7. **Zijn er nog aspecten voor u van belang die nog niet aan de orde zijn geweest?**

Als gevolg van het goede resultaat treedt Ron per 2-1-2017 in dienst bij LetsGrow.com.

8. **Bent u bereid een volgende keer weer uw medewerking te verlenen aan het beschikbaar stellen van een afstudeerplaats** (graag met toelichting)**?**

Ja, beide partijen leren hiervan en daarnaast levert het bruikbare software op.

PLAN VAN AANPAK

LetsGrow.com

Naam: Ron Persoon
Bedrijf: LetsGrow.com
Versie: 1.1
Datum: 30-08-16
Plaats: Vlaardingen

Opdrachtgever: Peter Hendriks

Begeleider: Leon Batta

Document historie

Datum	Versie	Beschrijving
31-08-2016	1.0	Eerste oplevering
01-09-2016	1.1	Verbeterde versie

Inhoud

1. Voorwoord

LetsGrow is opgericht in 2000 uit een samenwerking tussen Hoogendoorn Growth Management (HGM) en de WUR (Wageningse Universiteit Researchcentrum). Hoogendoorn is een leverancier op het gebied van klimaatregeling. In 2006 kwam de samenwerking tussen Hoogendoorn en de WUR ten einde en is LetsGrow overgenomen door Hoogendoorn.

LetsGrow biedt hun klanten, tuinders in de glastuinbouw sector, de mogelijkheid om meetgegevens van onder andere klimaatcomputers te analyseren en te vergelijken. Het gaat hier bijvoorbeeld om data zoals lichtsterkte, temperatuur, luchtvochtigheid, arbeid en energieverbruik. Klanten hebben de mogelijkheid om modules aan te schaffen om zo de meetgegevens van de klimaatcomputers te kunnen opnemen in het systeem van LetsGrow. Doordat LetsGrow de mogelijkheid biedt om data te vergelijken tussen meerdere locaties van een bedrijf of collega tuinders kunnen de klanten in hun tuinbouwbedrijf het beste klimaat realiseren en op deze manier de productie en kwaliteit verbeteren.

Het kantoor van LetsGrow is net als JB systems gevestigd in het pand van Hoogendoorn te Vlaardingen. JB Systems is ook een dochterbedrijf van Hoogendoorn en richt zich op industriële automatisering.

LetsGrow bestaat uit 8 werknemers:

Naam	Functie
Peter Hendriks	Manager, Sales
Christiaan Posthumus Meyjes	Helpdesk
Edwin Vis	Software ontwikkelaar, Systeembeheerder
Leon Batta	Software ontwikkelaar
Eray Soy	Software ontwikkelaar
Rick van den Heuvel	Software ontwikkelaar
Joost Simons	Software tester, Helpdesk
Yvonne Smits	Administratie, Helpdesk
Amar Kanhai	Helpdesk
Bart Schulte	Helpdesk

2. Aanleiding

LetsGrow heeft momenteel meer dan 700 klanten. Het gaat hier voornamelijk om Nederlandse klanten, maar ook enkele klanten in het buitenland, zoals in China en Kenia . Om het systeem ook voor buitenlandse en de wat minder technische klanten beschikbaar te maken is het noodzaak om een user interface te ontwikkelen waar iedere klant eenvoudig mee overweg kan. Ook is in de loop der jaren de vraag naar het gebruik van LetsGrow op de mobiele telefoon steeds verder toegenomen, maar het huidige systeem is hier niet op voorbereid.

2.1 Probleemstelling

Momenteel krijgen de klanten na inloggen op LetsGrow een pagina te zien met zeer beperkte informatie. Dit is momenteel niet erg overzichtelijk en klantvriendelijk in gebruik. Op dit moment worden er ook nog geen grafieken en meetgegevens getoond op pagina waar klant terecht komt na in loggen.

Doordat het huidige systeem niet is voorbereid voor het gebruik op mobiele telefoons is LetsGrow niet goed te gebruiken op mobiele apparaten. Dit is een beperkende factor in de huidige situatie, aangezien hier steeds meer vraag naar is.

Verder is er ook de vraag om op het klantenportal data uit verschillende externe bronnen te kunnen tonen, zoals twitter, het weerbericht en eventueel data van de veiling. Deze functionaliteit is namelijk in het huidige systeem niet beschikbaar.

3. Opdracht

In dit hoofdstuk wordt besproken wat het uiteindelijke doel is van het project. Verder wordt er besproken wat het uiteindelijke resultaat van het project zal zijn en er wordt besproken waar het project op afgebakend wordt.

3.1 Doelstelling

De doelstelling van de opdracht is het ontwikkelen van een overzichtelijke user interface in de vorm van een dashboard. Dit dashboard dient er voor te zorgen dat de buitenlandse en/of minder ervaren computer gebruikers goed overweg kunnen met LetsGrow.

Het dashboard zal responsive gemaakt moeten worden om deze ook bruikbaar te maken op mobiele apparaten zoals mobiele telefoons en tablets.

Ten slotte zal er ook de mogelijkheid geboden moeten worden om data uit externe bronnen te tonen. Het gaat hier om data zoals twitter, het weer en eventueel data van de veiling.

3.2 Resultaat

Er dient een overzichtelijk en aantrekkelijk dashboard te worden ontwikkeld of een bestaand dashboard pakket te worden gekozen aan de hand van pakketselectie. Op dit dashboard kunnen de klanten op een overzichtelijke manier hun belangrijkste meetgegevens zien. Tevens zal het dashboard eenvoudig in gebruik zijn, zodat de klanten eenvoudig weten waar er naar genavigeerd kan worden. Op deze manier is het voor de klant eenvoudig om alle functies van het systeem te vinden en te gebruiken.

Er zal de mogelijkheid geboden worden om data te kunnen tonen uit externe bronnen. Het gaat hier vooral om data vanuit twitter, het weer en eventueel data van de veiling.

Het ontwikkelde dashboard zal responsive zijn, zodat klanten de mogelijkheid geboden wordt om LetsGrow ook te gebruiken via hun mobiele telefoon of tablet.

Afhankelijk van de uitkomst van de pakketselectie zal ook de grafische kant van het dashboard ontworpen worden. Indien er een bestaand dashboard pakket geïmplementeerd wordt zal hier hoogstwaarschijnlijk het ontwerp ook van aangepast dienen te worden.

Het resultaat van de opdracht is een prototype van het dashboard.

3.3 Afbakening

Het project zal worden ontwikkelt via een iteratieve methode.

Om het dashboard te ontwikkelen zal er gebruikt gemaakt worden van de tools Visual Studio 2015 en Microsoft SQL Management Studio 2016.

3.4 Meerwaarde bedrijf

De meerwaarde voor het bedrijf is dat klanten minder vragen hoeven te stellen over de werking van LetsGrow. De reden hiervoor is dat de data overzichtelijk is in te zien en dat alles professioneler oogt richting de klant.

De verdere meerwaarde is dat de navigatie intuïtiever zal verlopen en het dashboard meer informatie laat zien in één oogopslag. Dit zorgt ervoor dat het systeem wat eigentijdser oogt.

4. Projectaanpak

4.1 Ontwikkelmethode

Binnen het bedrijf wordt er gebruik gemaakt van een eigen variant van Scrum. De muur van het kantoor waar ik werk is verdeeld in verschillende vakken. Dit zijn de vakken: idea/concept, todo, in progress, in test, done en impediment.

- Idea/concept: Functionaliteiten die ooit in het systeem ingebouwd kunnen gaan worden
- Todo: Functionaliteiten die in het systeem moeten komen, maar waar nog niemand aan begonnen is.
- In progress: Hier hangen de functionaliteiten waar aan gewerkt wordt.
- In Test: Hier hangen die functionaliteiten die af zijn en getest worden.
- Done: Deze functionaliteit is succesvol getest en kan gereleased worden.
- Impediment: Hier worden functionaliteiten opgehangen die niet gemaakt/afgemaakt kunnen worden vanwege een externe oorzaak.

Wanneer het voor iedereen uit komt wordt er om 10 uur ‘s ochtends een stand-up meeting gehouden. Iedereen komt dan in het kantoor staan en vertelt waar hij/zij op dat moment mee bezig is en wat hij/zij nog gaat doen. Officieel hoort een daily stand-up dagelijks te worden gehouden, maar aangezien niet altijd iedereen op kantoor aanwezig is wordt deze stand-up alleen gehouden wanneer het mogelijk is.

De reden dat er met een scrum variant wordt gewerkt is dat LetsGrow hoofdzakelijk een grote applicatie heeft. Hier worden constant nieuwe functies aan toegevoegd en tussentijds wensen van klanten ontwikkelt. Bovendien hebben de werknemers van LetsGrow niet allemaal dezelfde werkdagen. Dit zijn dan direct ook de redenen dat er niet met sprints gewerkt wordt. Doordat er tussentijds steeds opdrachten, bugs en support verzoeken tussen kunnen komen en niet alle werknemers dagelijks aanwezig zijn, is het lastig om te bepalen hoeveel tijd er nodig is om een taak af te kunnen maken.

Het te maken dashboard zal tijdens de ontwikkeling steeds verder worden uitgebreid qua functionaliteit. Het is dus vooraf niet mogelijk om een volledige lijst met requirements op te stellen. Aangezien er requirements bij kunnen komen is een agile methode een goede methodiek om deze opdracht mee uit te voeren.

Bij het ontwikkelen van het dashboard zal het wel mogelijk zijn om met sprints te werken en steeds een werkende versie op te leveren. Er zal grotendeels gewerkt worden volgens het scrum principe. Het zal echter wel gaan om een iets andere variant, aangezien de stand-up meetings niet dagelijks worden gehouden en ik de enige persoon in het scrum team ben.

4.2 Versiebeheer

Bij de ontwikkeling wordt er gebruik gemaakt van Microsoft Visual Studio 2016. Deze software biedt de mogelijkheid om gebruik te maken van de Team Foundation Server. Deze versiebeheermethode standaard is geïmplementeerd in Visual Studio en werkt dus ook uitstekend in combinatie met Visual Studio projecten.

4.3 Requirements analyse

Om requirements op een duidelijke manier te kunnen ordenen aan de hand van de prioriteiten wordt er gebruik gemaakt van MoSCoW(Clegg, 1994). MoSCow staat voor Must have, Should have, Could have en Won't have. Na het ordenen van de requirements ontstaat er een duidelijk beeld van welke requirements er als eerst gerealiseerd moeten worden.

- Must have: Deze functionaliteit moet in het eindproduct zijn opgenomen. Dit is belangrijk voor de werking van het systeem.
- Should have: Deze functionaliteit zou in het eindproduct opgenomen moeten zijn, maar de applicatie is ook bruikbaar zonder deze functionaliteit.
- Could have: Het zou mooi zijn als deze functionaliteit in het eindproduct wordt opgenomen, maar hier hoeft pas aandacht aan besteed te worden indien de overige functionaliteit is gerealiseerd.
- Won't have: Deze functionaliteit zal (nog) niet in het systeem opgenomen worden, maar zal mogelijk in latere versies worden opgenomen.

4.4 Ontwikkelstraat

LetsGrow werkt met verschillende omgevingen, namelijk de Ontwikkel, Test en Productieomgeving. De ontwikkelomgeving is sterk verbonden met de testomgeving, aangezien de ontwikkelomgeving de data gebruikt van de servers van de testomgeving. De ontwikkelomgeving draait ook niet op een aparte server, maar lokaal op de pc van de software ontwikkelaar.

Nadat de software ontwikkelaar klaar is met een functionaliteit wordt deze gereviewed en, indien de wijzigingen worden goedgekeurd, doorgevoerd naar de testomgeving. Zodra de functionaliteit hierop ook wordt goedgekeurd, zal deze worden doorgezet naar de productieomgeving. Er wordt bewust geen gebruik gemaakt van een acceptatie omgeving. Het gebruik hiervan weegt niet op tegen de extra tijd en kosten die inrichting en onderhoud van een dergelijke omgeving met zich mee brengt.

Ik heb er voor gekozen om de ontwikkelstraat van LetsGrow aan te houden, omdat dit voor het bedrijf goed werkt en deze ontwikkelstraat ook prima te gebruiken is voor mijn opdracht. Voor mijn opdracht is de acceptatieomgeving ook niet nodig, aangezien de testomgeving voldoende op de productieomgeving lijkt.

4.5 Modelleringtechniek

Tijdens dit afstudeertraject zal er gebruik gemaakt gaan worden van UML(Unified Modeling Language) (Grady Booch, James Rumbaugh en Ivar Jacobson, 1997). Deze techniek wordt op grote schaal gebruikt en aangezien ik hier al veel ervaring mee heb is dit voor mij een goede techniek. UML zal gebruikt worden voor verschillende diagrammen, zoals klassendiagrammen, sequentiediagrammen, use cases en use case tabellen.

De diagrammen zullen worden gebruikt om documentatie op te bouwen van het te ontwikkelen dashboard. Door de werking van de dashboard applicatie te documenteren wordt het eenvoudiger om hier (in de toekomst) wijzigingen in aan te brengen.

Om de diagrammen en tabellen te modelleren wordt er gebruik gemaakt van de tool Visual Paradigm. Deze tool beschikt over voldoende mogelijkheden en doordat ik deze tool op school ook vaak heb gebruikt heb ik hier al ervaring mee.

4.6 Testmethodiek

Bij LetsGrow wordt voornamelijk getest op usability en functionality. Bij het testen wordt er ingeschat hoe groot de eventuele imagoschade kan zijn, indien er nieuwe functionaliteit wordt opgeleverd waarbij er enkele bugs kunnen optreden. Dit is een bewuste keuze, aangezien er geen tijd is om de nieuwe en huidige functionaliteit steeds opnieuw te testen. Op dit punt wijkt LetsGrow dan ook af van de officiële scrum methodiek, waar er gebruik gemaakt wordt van testautomatisering en regressietesten.

Om het dashboard te kunnen testen en zo de kwaliteit vast te stellen, zal er wel gebruik gemaakt moeten gaan worden van een testmethodiek. Ik heb gekozen voor TestGoal(de Grood, 2012), aangezien ik hier al bekend mee ben en er hier veel documentatie over te vinden is. Verder is testgoal geschikt voor deze opdracht, omdat dit veel verschillende testmogelijkheden biedt met verschillende dieptes. De testen die tijdens de iteraties worden gemaakt, zullen tevens ook dienen als regressietesten om te valideren of de bestaande functionaliteit nog steeds naar behoren functioneert.

Er zal zowel getest worden via de black- als de whitebox methode. Er wordt blackbox getest, omdat er vanuit het dashboard gegevens worden opgevraagd uit verschillende applicaties van LetsGrow. De whitebox testen zullen worden opgezet voor de door mij zelf ontwikkelde delen van het dashboard.

Testgoal past binnen de ontwikkelmethodiek, aangezien iedere iteratie testen worden opgezet voor het stuk functionaliteit dat in de betreffende sprint ontwikkeld is. Hierdoor zal het detailtestplan, de testontwerpen en de testrapportage steeds verder worden aangevuld.

Om de risico’s te bepalen zal er gebruikt gemaakt worden van de testrisicoanalyse om hier vervolgens een strategie matrix mee op te stellen.

5. Risico's

Tijdens het ontwikkel traject kunnen er verschillende risico's optreden. Deze risico's kunnen leiden tot vertraging van het project. In de tabel hieronder worden een aantal risico's vermeld.

Wat	Kans	Impact	Oplossing
Ik word ziek	Midden	Klein	Een enkele dag ziek zal niet voor grote problemen zorgen, maar mocht dit te lang duren zal er contact opgenomen worden met school om te kijken naar een goede oplossing.
Mijn afstudeerbegeleider wordt ziek	Midden	Midden	Mocht dit slechts om een dag of paar dagen gaan, dan kan Edwin Vis mij ondersteunen. Mocht dit om een langere periode gaan zal dit gemeld worden op school en zal er gekeken moeten worden of iemand anders van LetsGrow de rol van begeleider over kan nemen.
Mijn kennis is niet genoeg om de opdracht te kunnen voltooien	Midden	Midden	Ik zal zelf moeten zorgen dat ik over de benodigde kennis ga beschikken door mij uitgebreid in te lezen op de benodigde materie. Mocht dit niet voldoende zijn zal er contact opgenomen worden met school.

6. Iteraties en planning

In dit hoofdstuk wordt besproken naar welke mijlpalen er wordt toegewerkt en welke producten het project uiteindelijk gaat opleveren. Ook wordt er per iteratie een planning weergeven. Aangezien er wordt gewerkt met een variant op scrum is nog niet exact bekend welke functionaliteiten het dashboard uiteindelijk zal bevatten. Om deze reden vermeld de planning alleen de onderdelen die zeker in het dashboard terug zullen komen en blijven de overige iteraties open staan voor eventuele veranderingen in requirements.

6.1 Mijlpalen

- 1. Definitieve versie Plan van Aanpak
- **2.** Eerste versie requirements lijst.
- 3. Resultaat onderzoek pakketselectie
- 4. Definitieve versie functioneel ontwerp
- 5. Definitieve versie technisch ontwerp
- 6. Definitieve versie testontwerpen
- 7. Definitieve versie testrapportage
- 8. Definitieve versie afstudeerverslag

6.2 Op te leveren producten

- Plan van aanpak
- Volledige lijst functionele en niet functionele requirements
- Resultaat onderzoek “technologie selectie of zelf ontwikkelen**”**
- Ontwerp mobiel
- Ontwerp pc
- Dashboard mobiel
- Dashboard pc
- Test-set en testbevindingen

6.3 Planning

Hieronder wordt de planning weergeven die het hele afstudeertraject omvat. Achter de iteraties zijn de belangrijkste mijlpalen gedefinieerd waar naartoe gewerkt zal worden.

Week 1 en 2: 29 Augustus t/m 9 September
In de eerste twee weken wordt het plan van aanpak opgezet, wordt er een start gemaakt met het opstellen van de requirements en wordt er onderzoek gedaan naar de werking van het huidige software pakket (Het systeem LetsGrow).
Mijlpalen:
- 1. Definitieve versie Plan van Aanpak
- **2.** Eerste versie requirements lijst.

Week 3 en 4 – Iteratie 1: 12 September t/m 23 September
In iteratie 2 wordt er onderzoek gedaan naar bestaande dashboard pakketten en wordt er een klein onderzoekje uitgevoerd naar de aanlevering van data uit externe bronnen. Verder is hier een eerste versie opgezet van het functioneel en technisch ontwerp.
Mijlpaal:
- 3. Resultaat onderzoek pakketselectie

Week 5 t/m 14 - Iteraties 2 t/m 6: 26 September t/m 2 December
In de weken 5 t/m 14 wordt het ontwikkelproces iteratief uitgevoerd. Hierbij zal steeds (indien nodig) het functioneel- en technisch ontwerp worden aangepast en er zal een deel ontwikkeld worden. Verder zal iedere iteratie het detailtestplan, de testontwerpen en het testrapportage worden uitgebreid, om zo iedere iteratie te eindigen met een werkend product. De structuur van de iteraties zal gelijk zijn, maar de daadwerkelijke implementatie van de sprintbacklog zal na iedere iteratie opnieuw bepaald worden met behulp van de productbacklog. De productbacklog is gebaseerd op de requirements en de overige activiteiten die nodig zijn om tot een werkend dashboard te komen. Deze productbacklog wordt bijgehouden op de muur in het kantoor waar ik in werk en in de bijlage “Productbacklog”.
Mijlpalen:
- 4. Definitieve versie functioneel ontwerp
- 5. Definitieve versie technisch ontwerp
- 6. Definitieve versie testontwerpen
- 7. Definitieve versie testrapportage

Week 15, 16 en 17: 5 December t/m 23 December
De laatste drie weken zullen besteed worden aan het afronden van de documentatie en het opleveren van de producten.
Mijlpalen:
- 8. Definitieve versie afstudeerverslag